An Efficient Technique for Security of Mobile Agents

A Thesis

submitted for the award of the degree of

Doctor of Philosophy

in

Computer Science and Engineering Department

Submitted by

Prabhjot Kaur
(Reg no: 951203004)

Under the Guidance of

Dr. Prashant Singh Rana
Associate Professor

Thapar Institute of Engineering and Technology, Patiala,
Punjab - 147004, India
October 2024

Certificate

I hereby certify that the work, which is being presented in the thesis, entitled "An Efficient technique for security of mobile agents" partial fulfillment of the requirements for the award of the degree of Doctor of Philosophy and submitted to the institution is an authentic record of my work carried out under the supervision of Dr. Prashant Singh Rana. I have cited the reference about the Text(s)/Figure(s)/Table(s) from where they have been taken.

The matter presented in this thesis has not been submitted either in-part or full to any other University/Institute for the award of any other degree.

(Prabhjot Kaur)
Registration No. 951203004

This is to certify that the above statements made by the candidate are correct and true to the best of my knowledge.

Verified by:

(Dr. Prashant Singh Rana)
Computer Science and Engineering Department
Thapar Institute of Engineering and Technology, Patiala, Punjab, India.

Acknowledgements

I thought this day would never arrive. Now that it has, my heart overflows with gratitude for those who have been with me throughout the ordeal. I extend my heartfelt and deep gratitude to Dr. Prashant Singh Rana, my supervisors for their exceptional guidance, unwavering support, and invaluable mentorship during my PhD journey. Their expertise, encouragement, and dedication have been instrumental in shaping the success of this research.

I am indebted to Dr. Prashant Singh Rana for their patience and encouragement during the moments of doubt and difficulty. Their mentorship and belief in my abilities have motivated me to overcome obstacles and stay focused on my research goals.

Dr. Prashant Singh Rana has been a constant source of inspiration, providing me with the freedom to explore new ideas while offering valuable feedback to refine my research. Their insightful comments and constructive criticism have consistently challenged me to strive for excellence.

I want to express my appreciation for their kindness and understanding, which made our collaboration both enjoyable and rewarding.

Thank you, Dr. Prashant Singh Rana, for being an exceptional mentor and playing an integral role in my academic journey.There advice and encouragement were always important guiding lights for when I lost my focus.

Sincere thanks must also go to Dr. Shalini Batra, Dr. Manider Singh, Dr. Parteek Bhatia, Dr. Anju Bala and Dr. Anupam Sharma for their continuous support by always providing a conducive academic environment and resources that facilitated the completion of this research. Their motivation and enthusiasm are contagious. I want to thank the members of my thesis committee: Dr. Neeraj Kumar, Dr. AK Verma and Dr. MD Singh. They generously gave their time to offer insightful comments towards improving my work. Numerous faculty members have always been very kind with their words of encouragement. Crossing paths with them while walking on campus has ever brought a smile to my face. I wish to give them my sincerest thanks. The office staff has always been available and ready for the assistance of any kind.

I am deeply grateful to my parents for their unwavering love, encouragement, and constant belief in my abilities. Their support throughout my academic pursuit has been a driving force behind my success. I also extend my heartfelt appreciation to my sib-

lings Er. Verinder Singh, Dr. Parminder Kaur, Mrs Harvinder Kaur and Dr. Yadvinder Singh who has been my rock and confidant during the challenging times of this research journey. Their encouragement, understanding, and willingness to listen have been invaluable.Irrespective of what they were dealing with on an individual level, they were always available for every conversation. I owe a lot to my parents and siblings, who encouraged and helped me at every stage of my personal and academic life and longed to see this achievement come true.

My research would have been impossible without the support of Dr. Prashant Singh Rana. My wholehearted thanks to him for sharing their domain knowledge, without which this research would have been an uphill task. Finally special thanks to Almighty for making this possible.

Abstract

In the realm of wireless sensor networks, the Mobile Agent (MA) paradigm presents significant advantages over the traditional client-server model, particularly in addressing the crucial issue of energy consumption. Optimizing the itinerary for efficient data collection and considering the detection of malicious nodes are pivotal factors. We delve into the challenges and risks of securing mobile agents within extensive and dynamic sensor networks. The security of large-scale networks is paramount, especially when harnessing mobile agents to optimize network efficiency and data processing capabilities. Securing large-scale networks is crucial for protecting sensitive data, ensuring uninterrupted operations, and upholding user trust.

This abstract delineates the security considerations associated with mobile agents in vast WSNs. Large-scale WSNs often operate in resource-limited and hostile environments, so they are vulnerable to various security threats, such as node compromise, data tampering, and unauthorized access. We assess potential vulnerabilities arising from mobile agent movement, communication, and data aggregation processes and scrutinize the impact of security breaches on network performance and reliability. To address these challenges, we survey state-of-the-art security mechanisms, including secure agent migration protocols, cryptographic methods for data protection, and trust management models for agent authentication and authorization.

In the first scheme, our research focuses on identifying and addressing attacks to prevent communication breakdown as sensor nodes become more vulnerable in dynamic environments. We use the SPIN protocol and machine learning models to classify attacks and propose an ensemble model with 95% average accuracy. K-Fold cross-validation ensures consistency.

The second scheme focuses on using the Border-Hunting Optimization-based Deep CNN (BHO-DCNN) for a mobile agent (MA)-based intrusion detection in Wireless Sensor Networks (WSN). This approach aims to accurately identify malicious activities within sensor networks. The BHO-DCNN algorithm resulted in 45 alive nodes, an end-to-end delay of 0.2572 ms, a normalized energy consumption of 0.1622 J, and a throughput of 0.3125 % for 50 nodes at a 100 % population rate.

The third scheme focuses on developing a self-configuring mobile agent-based intrusion detection system using a Hybrid Deep LSTM. In wireless networks, sensor nodes send data to cluster heads, which then transmit it to the base station. Cluster head selection

is based on a multi-objective function. The data is sent to the sink node through a mobile agent, where a Hybrid Deep LSTM classifier is used. The model will be trained to improve intrusion detection based on the self-configuring concept. Its effectiveness will be evaluated by comparing it with existing techniques using performance metrics such as energy consumption, delay, traffic overhead, and throughput.

The fourth scheme focuses on a comprehensive approach to safeguarding mobile agents from malicious code execution by utilizing a Dynamic Bloom Filter and BlowFish algorithm; it addresses the inherent security challenges in cloud networking. The proposed method aims to detect and prevent malicious code execution. The paper offers a compelling solution to enhance agent security by using Dynamic Bloom Filters, elliptical curve keys, and asymmetric BlowFish encryption. Implemented on the JADE platform, the results reaffirm the effectiveness of the Dynamic Bloom Filter and BlowFish algorithm in fortifying mobile agents against security threats.

Keywords: Wireless Sensor Networks (WSNs), Mobile Agents (MAs), SPIN Protocol, Machine Learning, Ensemble Model, Border Hunting Optimization (BHO), Deep Convolutional Neural Network (DCNN), Intrusion Detection System (IDS), LSTM, Network Simulator

Table of Contents

List of Figures

List of Tables

List of Abbreviations

ACS	Ant Colony System
ADT	Anomaly Detection Techniques
ANN	Artificial Neural Networks
ANNs	Artificial Neural Networks
ARDR	ARD Regression
Bi-LSTM	Bidirectional long short-term memory
BRR	Bayesian Ridge Regression
C3S	Concurrent Container Clusters Scheduling
CaaS	Container as a Service
CLA	Cellular Learning Automata
CM	Container Migration
CNNs	Convolutional Neural Networks
DL	Deep Learning
DLM	Dynamic Linear Models
DNN	Deep Neural Network
DVFS	Dynamic Voltage and Frequency Scaling
EN	ElasticNet
GA	Genetic Algorithm
GRU	Gated Recurrent Unit
HPC	High-Performance Computing
IF	Isolation Forest
KF	Kalmen Filtering
KNN	K-Nearest Neighbour
LightGBM	Light Gradient Boosting Machine
LR	Linear Regression
LSTM	Long Short Term Memory
LSTM-ED	Long Short-Term Memory Encoder–Decoder
MAE	Mean Absolute Error
MAPE	Mean Absolute Percentage Error
MCFP	Minimum Cost Flow Problem
MSE	Mean Square Error
MSE	Mean Square Error
MWh	Megawatts Hour
NN	Neural Network

OSVM	Open Support Vector Machine
PMs	Physical Machines
PSO	Particle Swarm Optimization
QoS	Quality of Service
RMSE	Root Mean Square Error
RNN	Recurrent Neural Network
SDWF	Self Directed Workload Forecasting Method
SLA	Service Level Agreement
STL	Seasonal Decomposition of Time Series
SVR	Support Vector Regressor
TCN	Temporal Convolutional Network
VMP	Virtual Machine Placement
VMs	Virtual Machines

List of Nomenclature

n number of sensor nodes

S_d and T_d sensor node dimensions

$\{0.5S_d, 0.5T_d\}$ sink node position

S_i and T_i direct values

S_c number of sensor nodes initiated at CH

S_c^s sensor nodes

H_M sensor nodes under the cluster

t number of cluster head

I_{st} t^{th} cluster head

N_B sink node

b_n the distance between CH and the sink node

I_d data bytes

E_{ee} electronic energy

E_t and E_a energy of transmitter

E_{rs} power amplifier in the transmitter

t_p present iteration

$\vec{C}$ and $\vec{B}$ coefficient vectors

$\vec{V_p}$ the position vector of lupus

F_f and F_b best fitness of the individual and fitness value of the Bovidae.

W_g Bovidae nearer to lupus

ϑ_b velocity of Bovidae

$(\tau + 1)$ lupus at time

$P_b(\tau)$ current position of Bovidae

ϑ_{pb} pestered Bovidae

θ_1 and θ_2 tangent values

$\vartheta_L(\tau + 1)$ the velocity of left lupus $(\tau + 1)$ lupus at time

$P_b(\tau)$ current position of Bovidae

ϑ_{pb} pestered Bovidae

θ_1 and θ_2 tangent values

$\vartheta_L(\tau + 1)$ the velocity of left lupus

Chapter 1

Introduction

Mobile agents are dynamic software programs that represent users in computer networks, seamlessly migrating between nodes to execute tasks on behalf of the user. They blend computer code and data designed to move autonomously between computers, continuing their operations seamlessly. They can roam across a network, interacting with resources and other agents while minimizing the risk of data loss. This allows for relocating complex processing functions to locations where extensive data processing is essential. Mobile Agents are also called transportable agents. They possess the capability to traverse vast networks, such as the World Wide Web, engaging with diverse hosts to gather information and fulfill user-assigned duties. The mobility of agents allows them to seamlessly move between platforms, with the home platform serving as the trusted starting point. Each platform can consist of multiple hosts and support various computational environments where agents interact. Agents can collaborate and communicate with one another, selectively sharing information while maintaining confidentiality. Security poses a primary challenge for the successful implementation of agent-based applications. They are classified into two types:

- Mobile Agents with pre-defined path: They have a static migration path.
- Mobile Agents with undefined path, i.e., Roamer: They have dynamic migration paths. The mobile agents choose their path according to the present network condition.

Mobile agents in computer networks are a revolutionary concept. They are autonomous software programs that can seamlessly move from one network node or device to another, carrying out tasks and making decisions on behalf of their users. Unlike traditional client-server models, mobile agents travel to the specific network node or device to perform their functions, reducing the load on the network and supporting disconnected operations. By employing mobile agents, handheld wireless devices can provide a reliable network for message transport over wireless nodes in the network, even in the face of low bandwidth and high risk of network errors due to high mobility. This capability can transform communication efficiency and reliability in large-scale wireless networks.

The growth of network and computing technologies has increased the demand for efficient communication paradigms for distributed applications. The traditional RPC model is not ideal due to continual connections and high network overhead. Mobile agent research has become more frequent, as mobile agents can execute code and data in agent platforms and perform distributed computation on behalf of the user. They offer superior flexibility and performance compared to existing communication paradigms. The use of mobile agents presents both benefits and challenges. While they provide increased flexibility in information retrieval, network and system management, and electronic commerce, their susceptibility to attacks hinders widespread adoption. Security is a critical concern, especially in networked environments with numerous users. Balancing security measures with usability is essential for the practical application of mobile agents.

Mobile agents can be employed in wireless handheld devices in two ways: An agent platform could be installed on the device, enabling mobile agents to run directly on it, or devices could access and use remote mobile agents running on wired networks [1].

Mobile agents represent a revolutionary computing paradigm, enabling seamless movement of applications between platforms to create a flexible distributed system. In the ever-expanding realm of internet usage, network infrastructure faces challenges like low bandwidth, high network load, and latency, which mobile agents, also known as intelligent robots, aim to alleviate. These autonomous software codes can communicate with remote terminals, Java applets, and distributed databases. Mobile agents can physically travel across networks and execute tasks on machines that support agent hosting. This unique capability allows processes to migrate between computers, split into multiple instances on different machines, and return to their origin. Unlike traditional remote procedure calls, process migration enables executable code to travel and interact with various systems and agents. In a broader sense, mobile agents act as a user's representative in a computer network, autonomously moving between nodes to perform computations on the user's behalf.

Mobility allows an agent to move or hop among agent platforms. The agent platform provides the computational environment in which an agent operates. The platform from which an agent originates is called the home platform and is usually the most trusted environment for an agent. One or more hosts may comprise an agent platform, and an agent platform may support multiple computational environments or meeting places where agents can interact. They may cooperate or communicate with other agents, making the location of some of their internal objects and methods known to other agents without giving all their information away. Security is the main issue when deploying applications based on agent technology.

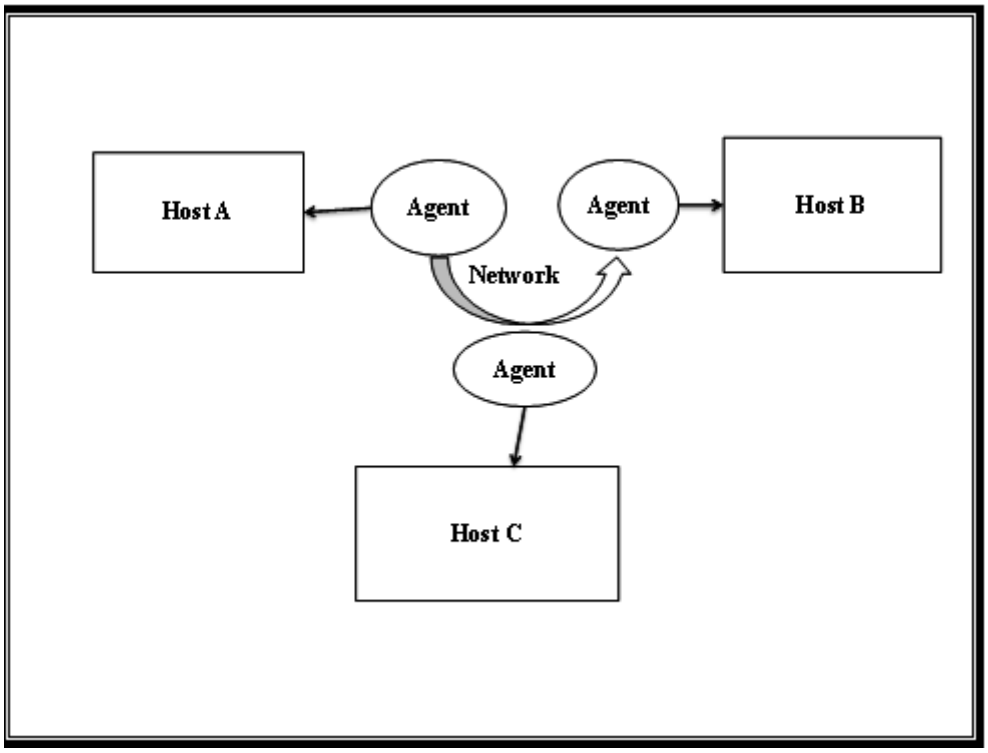


Fig 1.1: Working of Mobile Agent

The proposed methods begin with analyzing the attacks and security breaches on networks using mobile agents through Wireless Sensor Networks. Analyzing the vast data showed that security and attacks are not a single complex problem but a collection of many relatively more straightforward issues. The proposed technique considers this while analyzing the data and helps identify the critical attributes that secure the mobile and host platforms from malicious agents or platforms.

1.1 Background

In Wireless Sensor Networks (WSNs), mobile agents are crucial because they can independently move across network nodes and perform tasks like data aggregation and network management, as their many features are supported. However, mobile agents are susceptible to numerous security threats due to their dynamic nature and the sensitivity of the data they handle. Securing mobile agents involves protecting against unauthorized data access and tampering and ensuring both agents' and hosts' authenticity and integrity. This multifaceted security landscape requires robust cryptographic techniques, secure communication protocols, and stringent access control measures to uphold the effectiveness and dependability of mobile agent operations in WSNs.

Mobile agents, which autonomously migrate across sensor nodes to perform tasks, face

threats such as unauthorized access, data tampering, and malicious hosts. Ensuring secure communication, authentication, and data integrity is essential to mitigate these risks. Recent studies highlight various approaches, including cryptographic techniques and secure migration protocols, to enhance mobile agent security in WSNs [2]. The research underscores the importance of developing lightweight, efficient security solutions tailored to the resource-constrained environments of WSNs [3].

- **Mobile Agents Overview:** Mobile agents are software entities capable of autonomous movement within a network. They carry code, state, and data, enabling self-sufficiency in task execution. Their migration ability allows movement between different network nodes, including heterogeneous environments.
- **Early Research Focus:** Initial studies focused on agent mobility, security, and efficient communication protocols. Frameworks like the Java Agent Development Framework support agent execution and migration. The Aglets Communication Framework facilitates communication between agents and nodes.
- **Security Techniques:** Ensuring agent integrity, confidentiality, and authenticity during migration is critical. Techniques include encryption, digital signatures, and secure communication channels to protect agents from tampering and eavesdropping.

1.2 Challenges

Mobile agents offer flexibility and efficiency for distributed computing tasks in wireless sensor networks (WSNs). However, their security during migration poses significant challenges, requiring robust protocols and techniques to safeguard their operation in dynamic and heterogeneous environments. Despite these advancements, several challenges remain in ensuring the security and efficiency of mobile agents in WSNs:

- **Agent and Network Security During Migration:**
  - Protecting agents from attacks while they move between nodes.
  - Ensuring secure interaction with various network elements.
- **Managing Dynamic Environments:**
  - Handling the movement and interaction of agents in ever-changing network conditions.

  - Adapting to network topology changes and varying resource availability.

- **Efficient Utilization of Resources:**
  - Optimizing the use of network and computational resources.
  - Balancing the load to prevent network congestion and ensure efficient task execution.

1.3 Security threats on Mobile Agents

Mobile agents have advantages over other paradigms due to their properties of network load reduction, ability to overcome the problem of network latency, fault-tolerant capability, operation asynchronously, etc. On the other side, security risk is the main drawback of mobile agents. The most important feature of mobile agents in a network infrastructure is their mobility, through which they can autonomously and asynchronously travel in a network. Due to this feature, mobile agents are vulnerable to attacks that compromise communication security requirements like confidentiality, integrity, availability, accountability, etc.

1. **Mobile agents have two security aspects:**
   - Protecting the Host machine
   - Protecting the Mobile Agent
2. **Mobile agents and hosts are vulnerable to several threats:**
   - Agent to Host threats
   - Agent to Agent threats
   - Host to Agent threats

1.4 Limitation of Mobile Agents

There are limitations because of missing solutions to numerous issues arising in mobile agent systems concepts and design. Some of the limitations of mobile agents are:

1. A general statement for mobile agents that they are used to communicate to reduce network bandwidth consumption and performance, which is not valid in every scenario.

2. There is a lack of a systematic approach to designing mobile agents, making it challenging to develop.

3. Implementing agents is also hard work as so many unpredictable interactions are present in its journey to many places with adverse environments.

4. Mobile agents always keep secure information for authentication. Authentication is one of the essential processes for mobile agents. For this reason, the authenticating mechanism may be weak.

5. The testing and debugging of such systems are highly complex. This is because the approaches become more unpredictable.

6. Corruption of the agents is possible as the agent transfers over the various places it visits, i.e., an agent can lose information or deviate from its goal.

7. Since information might be lost, the agents cannot be trusted with secret information either. The data can be leaked on its way.

8. Mobile agents require a setup or an infrastructure to perform. A system that bases itself on mobile agents would require mobile agents to execute all the places.

9. Mobile agents, to interact, require a sort of a mechanism, a means to interact or to collaborate; there is a lack of such an approach, such a language that supports its mechanism.

10. Mobile agents remotely execute at one place and execute and go to the next; this is very much like a worm that can cause so much damage to the system if they are allowed to process.

1.5 Research Gaps

Presently, the major focus of research in the area of mobile agents is on the security of the agents. Protecting mobile agents against malicious platforms is still difficult, as no soundproof solutions are available. Although the prevalent techniques have succeeded to a certain extent, mitigation of the damage caused by malicious platforms cannot be eliminated. The mobile agent can't predict through run-time monitoring whether a platform will be malicious. Some of the major gaps that persist in the security of mobile agents from malicious hosts are:

1. **Lack of effective detection techniques:** Detection techniques such as state appraisal, reference states, and sedentary agents help detect malicious actions, but

they cannot successfully prevent such damage from malicious hosts.

2. **Existing Trust approach is the organizational solution:** The most straightforward approach consists of executing agents only in a trusted environment, i.e., hosts that are supposed to have correct behavior, so control is only based on their "reputation" and only trustworthy parties can operate hosts. The main problem is the limitation of hosts that can be considered as trusted. Furthermore, if there are no control mechanisms when a trusted host turns into malicious behavior, it is impossible to detect which host is.

3. **Lack of efficient prevention techniques:** Although various prevention techniques have received attention from multiple researchers, they have little success in protecting mobile agents against malicious platforms. Additionally, it brings to question the significance of mobility in agent-mediated applications. If the risk for agent mobility is huge under untrustworthy environments, then evaluating the necessity of mobility features for an agent-mediated application becomes essential. Avoiding mobility in untrustworthy or potentially untrustworthy environments could go a long way in reducing the attack surface for an application. Of course, such decisions about applications' architecture must be made after careful examination of various requirements such as availability, robustness, and security.

4. **Improvements desired in Black box Security technique:** Need for security architectures for creation, classification, and validation of trusted mobile agents:] The motivation behind architectures is that most of the current research and development results deal with the security of mobile agents, describe solutions only for the usage of mobile agents. These contributions assume that agents possess unique and recognizable identities, cryptographic keys, assigned assurance levels, and other security parameters. But, very few research contributions describe how to create, classify, and evaluate mobile agents before their adoption and deployment.

5. **Improvements in Security technique:** No general algorithm or approach exists for providing security.

1.6 Objectives

The broad objectives of this research work are:

1. To study, analyze and explore the already existing security techniques to protect mobile agents from malicious host and identify important security issues which need to be addressed.
2. To propose and implement a technique for the security of mobile agents from various attacks.
3. To test and validate the proposed technique on a simulated environment using various security testing parameters.

1.7 Thesis Organization

Chapter 1: Introduction

This chapter provides an overview of mobile agents in wireless sensor networks (WSNs), detailing their background, inherent challenges, security threats, limitations, research gaps, and objectives. Mobile agents are autonomous software entities capable of migrating across network nodes and executing tasks independently, offering flexibility and efficiency advantages. However, their mobility introduces significant security threats, such as tampering and unauthorized access, as well as challenges in resource management and dynamic environment adaptation. Existing literature reveals gaps in robust security mechanisms and efficient resource utilization. This research aims to address these gaps by proposing novel solutions and methodologies, thereby enhancing the security and efficiency of mobile agents in WSNs. The objectives include developing secure migration protocols, optimizing resource usage, and improving network resilience against threats.

Chapter 2: Literature Survey

The literature examines various approaches for detecting attacks and threats on mobile agents, including using available techniques and mechanisms—a detailed examination of existing research on mobile agents and their security in WSNs. Mobile agents offer a promising approach to enhancing the functionality and efficiency of Wireless Sensor Networks (WSNs). However, their integration introduces significant security challenges due to their autonomous nature and the resource-constrained environment of WSNs. This literature review examines the existing research on mobile agent security in WSNs, focusing on the primary threats, proposed solutions, and future research directions.

Chapter 3: Mobile Agents in Wireless Sensor Networks with Security Threats and Countermeasures

This chapter presents that mobile agents autonomously migrate between sensor nodes to perform tasks such as data collection and network management, and deploying mobile agents in WSNs presents several challenges, primarily related to security and resource constraints. Ensuring the secure operation of mobile agents involves protecting against threats such as unauthorized access, data tampering, and node impersonation. Robust cryptographic techniques, secure communication protocols, and effective authentication and authorization mechanisms are essential to safeguard mobile agents and sensor nodes. Their mobility and the resource-constrained nature of WSNs expose them to various security threats. These include unauthorized access, data tampering, and identity spoofing. Addressing these challenges requires robust cryptographic techniques, secure communication protocols, and stringent authentication and authorization mechanisms to protect the mobile agents and the sensor nodes they interact with for security threats and countermeasures.

Chapter 4: Mobile Agent-Based Intrusion Detection System This chapter focuses on examining the role of mobile agents in wireless sensor networks (WSNs) and addressing security challenges. Mobile agents offer advantages in WSNs, including enhanced data processing efficiency, reduced communication overhead, and improved scalability. However, deploying mobile agents introduces security concerns like node compromise and data integrity threats. Incorporating a chapter on "Using a Mobile Agent-Based Intrusion Detection System, an Ensemble Model for Automated Attack Classification in Large-Scale Wireless Sensor Networks" in a thesis requires a structured approach outlining its significance and implementation details. The chapter introduces Intrusion Detection Systems (IDS), emphasizing their crucial role in safeguarding large-scale WSNs from security threats. It delves into the concept of mobile agent-based IDS, highlighting benefits such as reduced network overhead and enhanced scalability. The focus is on detailing the ensemble model for automated attack classification and explaining the integration of multiple detection techniques to improve accuracy and reliability.

Chapter 5: Border-Hunting Optimization for Mobile Agent-Based Intrusion Detection

Integrating the "Border-Hunting Optimization for Mobile Agent-Based Intrusion Detection With Deep Convolutional Neural Network" chapter into the thesis entails a structured approach focused on a comprehensive literature review, detailed methodology, implementation specifics, performance evaluation, comparative analysis, and conclusive insights. The chapter begins with thoroughly exploring existing literature, highlighting

methodologies employing optimization techniques and deep learning, setting the stage for introducing Border-Hunting Optimization (BHO) in tandem with Deep Convolutional Neural Networks (CNNs) for intrusion detection. Methodologically, the thesis delineates the operational framework of BHO, elucidating how it optimizes mobile agent trajectories to enhance network surveillance efficiency while CNNs augment detection accuracy through robust pattern recognition. Detailed implementation accounts encompass agent deployment strategies, network configurations, and dataset specifics crucial for training and evaluating the BHO-CNN model. Rigorous performance evaluation is pivotal, rigorously assessing detection metrics such as accuracy, false favorable rates, and computational efficiency under varied intrusion scenarios. Comparative analysis juxtaposes BHO-CNN against conventional intrusion detection systems, emphasizing its efficacy and operational advantages. The discussion culminates in comprehensive insights, underscoring BHO-CNN's contributions to advancing mobile agent-based security in wireless sensor networks while identifying avenues for future research to refine further and expand its applicability.

Chapter 6:Self Configuring Mobile Agent-Based Intrusion Detection

The "Self-Configuring Mobile Agent-Based Intrusion Detection" chapter explores innovative intrusion detection systems (IDS) using mobile agents, focusing on self-configuring approaches. It begins with an introduction to mobile agent-based IDS, highlighting the limitations of traditional systems. The chapter outlines design principles, architecture, implementation details, and evaluation metrics with practical case studies. Critical analysis addresses the strengths and limitations of the self-configuring approach, paving the way for future research. The unauthorized entry into a computer system is one of the most significant risks to computer security in the modern era. Network attacks of novel types are constantly emerging as network applications expand quickly. The identification of intrusion activity that spreads across the public network is done via Intrusion Detection (ID). ID may have to handle many audit record formats. IDS, as a security tool, is now essential for spotting attacks on computer networks and resources. For mobile-agent-based ID in WSN, the proposed Dunnock Ibis Optimization LSTM model (DIO opt LSTM) has been developed for this research. To accurately identify an intrusion in a network, the DIO opt LSTM model's relevance is determined by combining its features with optimization and running it through a deep classifier. For the classifier parameters in the detection approach, which includes the searching, surrounding, group hunting, and prey attacking characteristics, the algorithm's fitness yields a better rate of convergence by utilizing the database IDS 2018 Intrusion CSVs (d1), the analysis is done based on performance metrics such throughput (TP), delay alive nodes, and NE. End-end-delay

(ED), normalized energy (NE), and TP for the 50 nodes at the 100 populate rate were all achieved with the developed DIO opt LSTM algorithm. The DIO opt LSTM method has a delay of 0.41, 0.80, and 0.46 at nodes 50, 100, and 200 at a population rate of 100. The BHO-DCNN at a population rate of 100 has NE of 0.16, 0.15, and 0.15 at nodes 50, 100, and 200. In conclusion, the developed DIO opt LSTM achieves a TP of 0.76, 0.85, and 0.89 bps at nodes 50, 100, and 200 at a population rate of 100.

Chapter 7: Enhancing Security with Dynamic Blow Filter and Blow Fish Security Protocol to Safeguard Against Malicious Mobile Agents. This chapter will explore the role of Dynamic Blow Filter and Blow Fish and its Security Protocol in safeguarding mobile agents in dynamic network environments. The chapter will begin by highlighting mobile agents' vulnerabilities and the need for robust security measures. It will then provide a comprehensive literature review, identify gaps in existing methodologies, and detail the protocol's design, implementation, and practical deployment. The chapter will also cover methodology, security test results, comparisons with existing methods, and a discussion of the implications of the findings. This thesis aims to contribute a comprehensive study of the Dynamic Blow Fish and Blow Fish algorithm and its Security Protocol, advancing knowledge and practical solutions in securing mobile agents against malicious adversaries.

Chapter 8: Conclusion and Future Directions The chapter provides a summary of the conclusions drawn from the thesis, as well as potential future directions. By highlighting the key findings in the areas of conclusions and future scope, it is evident that this thesis can make a substantial contribution to enhancing the security and efficiency of mobile agent-based intrusion detection systems in wireless sensor networks.

Chapter 2

Literature Survey

This chapter introduces the concept of mobile agents, including their working model, characteristics, features, types, limitations, and challenges. The discussion about working and available mobile agent systems will greatly contribute to further study.

2.1 Mobile Agents

Mobile agents are autonomous software entities that can migrate across network nodes, carrying their code and state. They can execute tasks, interact with network components, and exchange data with other agents and hosts. These agents are autonomous entities that can perform tasks and make decisions on behalf of their owners or users. They are designed to carry out specific functions such as data collection, processing, or communication within a network. Mobile agents are different from traditional client-server models, where the client sends a request to the server, and the server processes the request and sends the response back to the client. In the case of mobile agents, the agent itself travels to the appropriate network node or device to perform the required task.

2.2 Features of Mobile Agents

1. Mobility: With the help of the mobility feature, mobile agents migrate from node to node, and communication processes can be performed in a wireless and fixed network.
2. Autonomy: Mobile agents execute autonomously on behalf of some other process.
3. Communication: Mobile agents can communicate with another mobile agent, servers (i.e., either mobile or fixed), and other clients.
4. Learning: One of the most exciting features of mobile agents is their learning ability, i.e., they change their behavior according to their past experience.
5. Interoperability: Mobile agents have the property to operate on different platforms or over different clients and adapt to environmental changes.

6. Persistence: Mobile agents do not need to establish continuous connections for the execution of programs, i.e., it deals with disconnected operations.

2.3 Benefits of Mobile Agents

- Reduction in network load.
- Ability to overcome the problem of latency.
- Capability of encapsulating protocols.
- Dynamic adaptation.
- Ability to execute asynchronously.
- Fault tolerance.
- Mobile agents are heterogeneous in nature

2.4 Applications of Mobile Agents

Mobile agent can be used in many commercial applications like

- Data collection or information retrieval.
- E-Commerce
- Parallel Computing
- Mobile Computing
- Network Management

This technology offers several benefits, including efficient resource utilization, reduced network latency, flexibility, adaptability, fault tolerance, and security. Shashank Srivastava et al. The mobile agent technique was a software-oriented agent system that offers to move between different locations. Developing an agent-based system, the system itself consists of a mobility framework. The framework consists of all agent models, including the navigation model.[4] Chia-Hui Wei et al. describe The field of agents as having numerous applications, methodologies, and thoughts, which has made it one of the more dynamic research areas in past years. The agent model was a newer extension in distributed computing models.[5] C. Zhang et al. proposed a significant approach of mobile agents in networks in distributed computing and task offloading. Mobile agents

can move to different nodes to perform computations locally, minimizing network traffic and reducing the load on central[6]

Nimbalkar et al. introduced a development of load balancing by migrating the agents on different platforms. This provided improved work in assassinating the movement in distributed environments, which allowed dynamic processing from host to host to any point. A migration algorithm was introduced to migrate the processes dynamically to a host having low load using the load of CPU as parameter.[7]

Dada E. G.et al. Mobile Agent is an alternate way to compose a dignified distributed system. The mobile agent advent extends the applet approach by moving code and information settings starting with one host and then onto the next. The agents moving at one location move with their information to another host, and at that host, execution is extended. A provisional discussion is done and allocation according to the execution of the JADE and Aglet mobile agents characteristics and also focuses on JADE and Aglet architecture as a suitable framework for distributed system.[8]

Sharma et al. The mobile agent paradigm is one such science that has several applications where it can be beneficial to identify a few areas. The places the mobile retailers have practicably deployed are database search, distributed framework, and e-commerce. It can produce very proper results in confined resources or in poor surroundings where bandwidth and memory are giant constraints. However, it is still not broadly normal due to its protection issues. This work seeks to explain the targets and properties of the mobile marketers in the currently used architecture and platform of the mobile world from the presently used strategy RPC(Remote Procedure Calling) and new approach RP (Remote Programming) of the mobile network, we can distinguish the quite a number used unused and new features of mobile agents.[9]

2.5 Multi Agents Model

A multi-agent model is a computational or conceptual framework involving multiple autonomous entities and agents interacting and collaborating to achieve specific goals or solve complex problems. Each agent possesses its own goals, knowledge, and capabilities and can make decisions and take actions independently based on its internal state and interactions with other agents. The concept of multi-agent models has been widely studied and applied in various fields. For example, in the domain of robotics, multi-agent models have been used to simulate and study the coordination and cooperation of multiple robots in tasks such as exploration, search and rescue, and swarm robotics. In the field of economics, multi-agent models have been employed to simulate market dynam-

ics, analyze the behavior of individual agents in complex economic systems, and predict market outcomes [10]

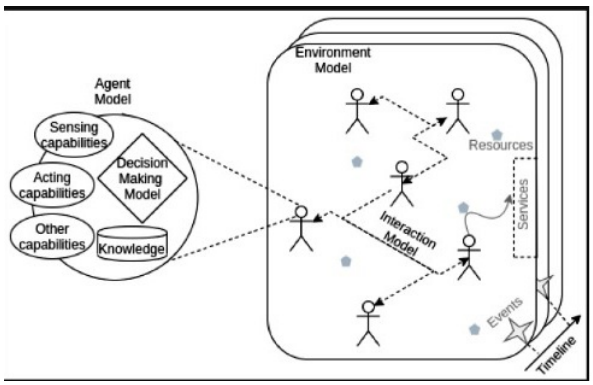


Fig 2.1: Process of Multi Agent System

In the scenario of multiple autonomous agents, the multi-agent model concept will be used, such as traffic lights, vehicles, and pedestrians interact and collaborate to optimize traffic flow and ensure efficient transportation. Each traffic light agent has its own local goals, such as minimizing congestion or maximizing throughput, and decides when to change signal phases based on local traffic conditions and predefined rules. Vehicle agents, on the other hand, navigate through the road network and make decisions on route selection, speed, and lane changes to reach their destinations efficiently. The interactions between these agents can be modeled using communication protocols, negotiation mechanisms, and coordination strategies. For example, traffic lights can communicate with neighboring lights to synchronize their signal timings, allowing for smoother traffic flow along a corridor.

2.6 Characteristics of Mobile Agent

Mobile agents possess several characteristics that distinguish them from traditional software entities. Here are some key characteristics of mobile agents:

1. **Autonomy** Mobile agents are autonomous software entities capable of making decisions and performing tasks independently. They can execute their code and perform operations without constant interaction with a central server or host.

2. **Mobility** One of the defining features of mobile agents is their ability to migrate or move across network nodes. They can travel from one host to another, carrying their code, state, and execution context. Mobility allows agents to execute tasks on different nodes and leverage local resources.

3. **Code and State Mobility** Mobile agents encapsulate their code, state, and execution context, which can be transferred from one host to another during migration. This characteristic enables agents to execute seamlessly on different hosts without losing their internal state.

4. **Asynchronous Execution** Mobile agents operate asynchronously, meaning they can execute tasks independently from their original host or the central server. They can continue their execution even without direct network connectivity or when the original host is offline.

5. **Communication and Interaction** Mobile agents can communicate and interact with other agents, host nodes, or servers in the network. They can exchange messages, transmit data, and coordinate actions with other entities, enabling collaboration and distributed processing.

6. **Intelligent Behavior** Mobile agents can exhibit intelligent behavior by adapting their actions based on local information and context. They can make decisions, process data, and adjust their behavior dynamically, providing flexibility and adaptability in different network environments.

7. **Resource Awareness** Mobile agents know the resources available on host nodes. They can utilize local resources efficiently, such as processing power, memory, and storage, to optimize their execution and minimize network traffic.

8. **Security Considerations** Mobile agents require security mechanisms to protect against unauthorized access, code injection, or tampering during migration. They may employ encryption, authentication, and access control measures to ensure secure operation and protect sensitive data.

2.7 Available Mobile Agent Models

These characteristics make mobile agents well-suited for various applications, including distributed computing, network management, data aggregation, and intelligent decision-making in dynamic network environments. Mobile agents' specific characteristics and behavior can vary based on the implementation, programming model, and requirements of the system or application in which they are deployed.

Several mobile agent systems are available to facilitate the development and deployment of mobile agents in various network environments. Here are a few examples of well-known mobile agent systems:

1. **Rover Toolkit (1990)** The Rover Toolkit, developed by the University of Virginia, provided a platform for creating and executing mobile agents in distributed systems. It offers a flexible and extensible framework for building mobile agent systems and supports agent migration, communication, and resource management features.

2. **Aglets (1997)** Aglets is a popular mobile agent system developed by IBM. It provides a Java-based platform for creating and executing mobile agents. Aglets offer features such as mobility support, communication facilities, and security mechanisms.

3. **SEAL(Secure Environment for Agent-based Old Economy)(2000)** SEAL was developed by the University of Minnesota. It focused on providing security features for mobile agent systems, such as secure communication and code protection. SEAL (Simple and Efficient Agent Language) is a mobile agent system emphasizing simplicity and efficiency. It provides a lightweight platform for mobile agent development and execution, supporting mobility, communication, and security.

4. **MOLE (Mobile Objects, Language, and Environment) (2000)** MOLE, developed by the University of Twente in the Netherlands, was deployed in the early 2000s. It aimed to provide a flexible and extensible environment for mobile agents and their execution in distributed systems. It offers a scalable and flexible platform for building mobile agent systems, including agent migration, communication, and fault tolerance features.

5. **JADE (Java Agent DEvelopment Framework) (2001)** JADE, developed by Telecom Italia and released as an open-source platform, is a popular mobile agent framework written in Java. It enables the development and execution of mobile agents in distributed systems. It provides a comprehensive framework for developing intelligent agent applications, including support for mobility, communication, and agent coordination.

6. **Grasshopper(2003)** Developed by Fujitsu Laboratories, Grasshopper is a mobile agent system designed for distributed computing environments. It supports the development and execution of mobile agents using a lightweight architecture. Mobile agents are written in various programming languages and provide migration, communication, and fault-tolerance features.

7. **Voyager(2004)** Voyager, developed by Agent is Software, is a mobile agent system that enables the creation, deployment, and execution of mobile agents in distributed networks. It provides a framework for building agent-based applications.

8. **Concordia (2009)** Developed by Nortel Networks, Concordia is a mobile agent platform that allows the development and deployment of mobile agents in distributed environments. It supports the execution of mobile agents on different nodes and facilitates distributed computing.

These are just a few examples of mobile agent systems available in the research and development community. Each system has its own set of features, programming models, and targeted application domains. The choice of a mobile agent system depends on the specific requirements, programming language preferences, and architectural considerations of the intended application.

2.8 Working of Mobile Agents

The mobile agent consists of the program code and the program execution state (the current values of variables, the next instruction to be executed, etc.). Initially, a mobile agent resides on a computer called the home machine, which is dispatched to execute on a remote computer called a mobile agent host (a mobile agent host is also called a mobile agent platform or mobile agent server). When a mobile agent is dispatched, the entire code of the mobile agent and the execution state of the mobile agent are transferred to the host.

The host provides a suitable execution environment for the mobile agent to execute. The mobile agent uses the host's resources (CPU, memory, etc.) to perform its task. After completing its task on the host, the mobile agent migrates to another computer. Since the state information is also transferred to the host, mobile agents can resume the code execution from where they left off in the previous host instead of restarting the execution from the beginning. This continues until the mobile agent returns to its home machine after executing the last device in its itinerary.

2.8.1 Life Cycle of Mobile Agent

Mobile agents work in networks by migrating or moving across network nodes to perform tasks and execute operations. Here are the phases of how mobile agents work in networks:

1. **Agent Creation** Mobile agents are created by a host or a central server and are equipped with their code, state, and execution context. The creation process involves defining the agent's purpose, behavior, and initial instructions.

2. **Agent Deployment** Once created, the host or server deploys mobile agents in the

network. They are typically deployed on specific agent hosts or nodes capable of executing the agent's tasks.

3. **Agent Activation** After deployment, the mobile agent is activated to initiate its execution. The agent starts running its code and begins its mission or task.

4. **Agent Migration** Mobile agents can migrate or move across network nodes during their execution. Migration allows agents to travel from one host to another, carrying their code, state, and execution context. Migration can be initiated by the agent itself or by external triggers, such as predefined conditions or instructions from the host.

5. **Agent Execution** Once a mobile agent reaches a new host, it resumes its execution. The agent interacts with the local environment, accesses resources, and performs computations or actions based on its programmed behavior. It may collect data, perform data processing, make decisions, or coordinate activities with other agents or entities in the network.

6. **Agent Communication** Mobile agents communicate with other network agents, hosts, or servers. They can exchange messages, transmit data, or coordinate actions with other agents or entities. Communication can be achieved through predefined communication protocols or interfaces.

7. **Agent Task Completion** Mobile agents continue executing until they complete their assigned task or mission. Once the task is accomplished, the agent may return to the original host or terminate its execution, depending on the specific requirements and objectives.

8. **Agent Termination** Mobile agents can terminate their execution voluntarily or as the host or server instructed. Termination may occur when the agent has completed its task, encountered an error, or reached a predefined termination condition.

Throughout the process, mobile agents leverage the resources and capabilities of the host nodes they migrate to, allowing for distributed and decentralized execution of tasks in the network. Mobile agents' specific implementation and behavior can vary depending on the application, network architecture, and programming model used. Mobile agent technology provides flexibility, autonomy, and mobility for executing tasks in networked environments.

2.8.2 Components of Mobile Agents Life Cycle

Three components, code, state, and attributes, complete the cycle of a mobile agent. Every mobile agent carried these three components for their identification as well as for the task:

- **Code**- the program (in a suitable language) that defines the agent's behavior.
- **State** – the agent's internal variables, etc., which enables it to resume its activities after moving to another host.
- **Attributes** – information describing the agent, its origin and owner, its movement history, resource requirements, authentication key, etc. Part of this may be accessible to the agent itself. Still, the agent must not be able to modify the attributes.

2.9 Limitations of Mobile Agents Model

There are limitations because of missing solutions to numerous issues in mobile agent systems concepts and design. Some of the limitations of mobile agents are discussed below:-

1. A general statement for mobile agents that they are used to communicate to reduce network bandwidth consumption and performance, which is not true in every scenario.
2. A Lack of a systematic approach to designing mobile agents makes it difficult to develop.
3. Implementing agents is also hard work as so many unpredictable interactions are present in its journey to many places with adverse environments.
4. Mobile agents always keep secure information for authentication. Authentication is one of the important processes for mobile agents. For this reason, the authenticating mechanism may be weak.
5. The testing and debugging of such systems are extremely complex. This is due to the fact that the approaches become more unpredictable.
6. Corruption of the agents is possible as the agent transfers over the various places it visits, i.e., an agent can lose its information or actually deviate from its goal.
7. Since information might be lost, the agents cannot be trusted with secret information either. The information can be leaked on its way.

8. Mobile agents require a setup or an infrastructure to perform. A system that bases itself on mobile agents would require mobile agents to execute all the places.

9. Mobile agents, to interact, require a sort of a mechanism, a means to interact or to collaborate; there is a lack of such an approach, such a language that supports its mechanism.

10. Mobile agents remotely execute at one place and execute and go to the next; this is very much like a worm that can cause so much damage to the system if they are allowed to process.

2.10 Challenges in Mobile Agent Technology

The use of mobile agent technology in distributed systems and Wireless Sensor Networks (WSNs) presents security challenges due to the decentralized and dynamic nature of these environments. Mobile agents are at risk of interception, modification, or destruction by malicious entities as they traverse various nodes. Limited resources in WSNs make it hard to implement robust encryption and authentication mechanisms, leaving agents vulnerable to attacks. Developing lightweight, energy-efficient security protocols and adaptive intrusion detection systems is vital to address these challenges in distributed systems and WSNs.

1. **Security and Trust**
   Ensuring the security of mobile agents during migration and protecting against unauthorized code execution or tampering remains a significant challenge.[3]. When mobile agents move between systems, the host system must trust the agent and ensure it doesn't execute malicious activities. Verifying the integrity and authenticity of mobile agents poses a challenge, as they can potentially carry malicious code or perform unauthorized actions.

2. **Agent Coordination and Synchronization**
   Coordinating the activities of multiple mobile agents and synchronizing their actions to achieve desired objectives can be complex, especially in large-scale systems [11].

3. **Network Heterogeneity and Interoperability**
   Mobile agent systems need to handle diverse network environments, including different operating systems, hardware platforms, and communication protocols. Ensuring interoperability and compatibility across heterogeneous networks poses a challenge [3]. Mobile agents heavily rely on network connectivity to communicate with remote systems and exchange data. However, mobile devices often operate in dynamic and

unpredictable network environments, leading to intermittent connectivity or high latency. These network limitations can impact the responsiveness and reliability of mobile agents.

4. **Load Balancing and Resource Management**
   Efficient load balancing and resource allocation in mobile agent systems can be challenging due to the dynamic nature of agent migration and varying resource availability on host nodes [12].

5. **Resource Constraints**
   Mobile devices typically have limited computational power, battery life, and storage capacity. These constraints can restrict the functionality and performance of mobile agents, especially when executing complex tasks or interacting with resource-intensive applications.

2.11 Summary

This chapter serves as a comprehensive summary of the crucial points discussed, highlighting the significance of mobile agents in distributed computing environments. It emphasizes the benefits of mobile agents, such as reduced network traffic and improved scalability, while also addressing the challenges and limitations that must be overcome. The summary effectively underscores the current state of research and development in mobile agent technology, emphasizing the imperative need for ongoing innovation to conquer existing challenges.

Chapter 3

Overview of Mobile Agents in Wireless Sensor Networks

This chapter explores the utilization of mobile agents in Wireless Sensor Networks (WSNs). Mobile agents, which are self-governing software entities capable of traversing networks, offer substantial benefits in administering and enhancing sensor networks. It addresses fundamental concepts, real-world applications, constraints, and security concerns associated with mobile agents in WSNs.

The world of mobile and wireless computing presents unique challenges due to the diverse array of devices, user mobility, and constantly changing network conditions and execution context. Introducing mobile agent-based middleware to support mobile computing requires expanding the platform architecture, typically organized in layered services. This chapter delves into MA-based mobile computing middleware and an MA-enabled platform on wireless hand-held devices, with a specific focus on MA technology in mobile and wireless computing. Mobile agent-based middleware offers a sophisticated infrastructure that integrates support protocols, mechanisms, and tools to enable communication and coordination among mobile entities. Many mobile applications can greatly benefit from the use of mobile agents, which are also well-suited for advanced applications such as workflow management. The chapter overviews the latest advancements in mobile and wireless computing middleware.

In WSNs, Mobile Agents (MAs) are software abstractions performing information-rich data collection and autonomous data processing while dynamically migrating between network nodes so that data is exchanged between participant nodes data collection and autonomous data processing while dynamically migrating between network nodes so that data is exchanged between participant nodes[13].

Mobile agents in wireless sensor networks (WSNs) have been an active research area. Mobile agents are intelligent entities that can move autonomously through the sensor nodes in a network, collecting and processing data and executing tasks on behalf of the network. They can be deployed to improve the efficiency, reliability, and scalability of WSNs by enabling dynamic data aggregation, data fusion, energy optimization, and fault

tolerance.

Wireless Sensor Networks (WSNs) are emerging as powerful platforms for distributed embedded computing supporting various high-impact applications. However, programming WSNs is a complex task that requires suitable paradigms and technologies capable of supporting the specific characteristics of such networks, which uniquely integrate distributed sensing, computation, and communication. Mobile agents are a distributed computing paradigm based on code mobility that has already demonstrated high effectiveness and efficiency in IP-based, highly dynamic distributed environments. Mobile agents can provide more benefits in the context of WSNs than in conventional distributed environments. In this paper, we first discuss using mobile agents as an effective enabling technology for WSNs and then describe a lightweight framework for supporting mobile agents in WSNs.

Emerging mobile agent technology with the WSNs can reduce the network transformed overcome the network latency, and is an eligible substitute for Client/Server model. Wireless sensors work competently for a flexible connection subsystem. The client/Server model adopted in traditional intelligent maintenance, which requires transmitting vast data via a network, is unsuitable in wireless sensor networks with limited bandwidth and unstable connection. In distributed computing, the mobile agent model is the mainstream technology. With autonomy, communication, mobility, and role, the mobile agent model is more suitable for large-scale, resources-restrained WSN to deal with big data.

Agent-based technology can provide an efficient solution for the energy optimization problems of WSN. Mobile agents in wireless sensor networks (WSNs) have been an active research area. Mobile agents are intelligent entities that can move autonomously through the sensor nodes in a network, collecting and processing data and executing tasks on behalf of the network. They can be deployed to improve the efficiency, reliability, and scalability of WSNs by enabling dynamic data aggregation, data fusion, energy optimization, and fault tolerance.

Wireless Sensor Networks (WSNs) are emerging as powerful platforms for distributed embedded computing supporting various high-impact applications. However, programming WSNs is a complex task that requires suitable paradigms and technologies capable of supporting the specific characteristics of such networks, which uniquely integrate distributed sensing, computation, and communication. Mobile agents are a distributed computing paradigm based on code mobility that has already demonstrated high effectiveness and efficiency in IP-based, highly dynamic distributed environments. Mobile agents can provide more benefits in the context of WSNs than in conventional distributed environments. We will use mobile agents as an effective enabling technology for WSNs and then describe

a lightweight framework for supporting mobile agents in WSNs.

3.1 Key Concepts and Applications of Mobile Agents in WSNs

This section outlines the fundamental ideas behind mobile agents and their integration into WSNs. It highlights:

1. **Dynamic Data Aggregation** Mobile agents can move through the sensor nodes, intelligently aggregating data from various nodes and reducing redundant transmissions. That helps in optimizing energy consumption and bandwidth utilization.
2. **Data Fusion** Mobile agents can perform data fusion tasks by collecting data from multiple nodes, combining it, and making decisions based on the fused data. That can improve accuracy and efficiency in target tracking or environment monitoring applications.
3. **Energy Optimization** By intelligently planning their movement and data collection routes, mobile agents can help conserve energy in individual sensor nodes, prolonging the overall network lifetime.
4. **Fault Tolerance** Mobile agents can assist in detecting and isolating faulty nodes by periodically checking their status or employing specific fault detection algorithms.
5. **Network Management and Reconfiguration** Mobile agents can reconfigure the network topology, optimize communication paths, and perform self-organization tasks.
6. **Security** Mobile agents can enforce security measures, such as intrusion detection and prevention, by analyzing data locally and reporting any suspicious activities.

Further, it promotes the use of mobile agents as an enabling technology for programming WSNs. The specific characteristics of mobile agents can provide more benefits in the context of WSNs than in conventional distributed environments. A mobile agent framework has been defined to experiment with mobile agents in WSNs, which provides an event-driven component-based architecture for supporting agent execution, communication, and migration. Mobile agents are programmed according to a hybrid yet effective programming language, which integrates three major WSN programming paradigms (event-driven, state-based, and mobile agent-based programming) along with the emerging role-based programming. A programming example of mobile agents at the middleware layer (data collection) has also been presented and discussed to exemplify the proposed

framework, which is currently being developed by using the innovative Sun SPOT technology [14] which allows for easy programming of the sensor nodes in Java. In particular, both the agent server and the mobile agents are implemented as Isolates (applications represented as isolated threaded objects), which can be easily paused, migrated to a different sensor node, and restarted.

The Mobile Agent model has efficient data accumulation ability in Wireless Sensor Networks. An unnamed Vehicle is a computing entity that can collect precedence information sensed, operating individually and getting a series and sequence of aims on behalf of users[15]. It performs data processing one by one while moving from node to node. Its features include reactivity, independence, goal-oriented, mobility, adaptability, and communication skill, including Lifetime Improvement using Mobile Agents in Wireless Sensor Networks.

3.2 Uses of Mobile Agents in Wireless Sensor Networks

Mobile agents have various applications in wireless sensor networks (SNs) due to their ability to move autonomously and execute tasks. Here are some critical applications of mobile agents in WSNs:

1. **Limitations:** Constraints related to resource usage and operational complexities. Challenges and Considerations: Broader issues such as mobility management and performance optimization.
2. **Itinerary Planning:** Techniques for effective planning of mobile agent movements and tasks.
3. **Security Issues:** Examination of potential security threats and the requirements for securing mobile agents.
4. **Security Threats and Requirements:** Detailed analysis of specific threats and necessary security measures. Malicious Hosts: Issues related to malicious nodes that may attempt to compromise the network and strategies to counteract them.
5. **Environment Exploration:** In applications like environmental monitoring or disaster response, mobile agents can be deployed to explore and gather data from hard-to-reach or hazardous locations.
6. **Collaborative Data Processing:** Mobile agents can facilitate collaborative data processing among sensor nodes. They can collect data from multiple nodes, process

it locally, and then distribute the results to relevant nodes.

7. **Task Offloading:** Mobile agents can offload computation-intensive tasks from resource-constrained sensor nodes to more powerful base stations or cloud servers, reducing the burden on individual nodes.

8. **Multi-Hop Communication:** Mobile agents can serve as data carriers between distant or disconnected sensor nodes, enabling multi-hop communication and extending the network's communication range.

Using mobile agents in WSNs offers several advantages, including improved efficiency, scalability, fault tolerance, and adaptability. However, their implementation requires careful consideration of factors like agent mobility patterns, energy consumption, communication overhead, and security considerations. Mobile agents offer advantages and disadvantages in the context of Wireless Sensor networks (WSNs). On the positive side, mobile agents bring significant benefits to WSNs. Their ability to move autonomously allows for dynamic data aggregation and fusion, reducing redundant transmissions and optimizing energy consumption in the network. By intelligently planning their routes, mobile agents can extend the overall network lifetime and improve fault tolerance by Mobile agents and isolating faulty nodes.

Moreover, network management and reconfiguration tasks, such as optimizing communication paths and adjusting transmission power. Mobile agents can also enhance security by actively monitoring the network for anomalies and reporting suspicious activities, thus improving intrusion detection and prevention. Additionally, they enable collaborative data processing and task offloading, allowing for efficient data handling in resource-constrained sensor nodes.

However, mobile agents also come with certain drawbacks. The deployment of mobile agents introduces additional complexity to the network, leading to increased communication overhead and potential latency issues. Coordinating the movement of agents and ensuring their safety and reliability can be challenging, especially in large-scale WSNs. Moreover, mobile agents may consume significant energy during their movement, potentially offsetting some energy-saving benefits they provide through data aggregation.

Security is another concern, as mobile agents can become attack targets, potentially compromising the network's integrity and confidentiality. Ensuring the trustworthiness of mobile agents is vital to prevent malicious agents from compromising the system. Furthermore, designing efficient and robust mobile agent-based algorithms requires careful consideration of various factors, such as agent mobility patterns, data collection strategies, and task scheduling.

Mobile agents bring numerous advantages to Wireless Sensor Networks, including improved data aggregation, energy optimization, fault tolerance, and security. However, their implementation should be carefully managed to address communication overhead, energy consumption, coordination, and security challenges. Despite the drawbacks, mobile agents hold great potential to enhance the efficiency and reliability of WSNs and continue to be an area of active research and development.

3.3 Limitations of Mobile Agents in WSNs

Mobile Agents in Wireless Sensor Networks (WSNs) offer several advantages, but they also come with certain limitations that must be considered during their deployment and implementation.

1. **Communication Overhead:** The deployment of mobile agents introduces additional communication overhead in the network. Agents need to communicate with sensor nodes to exchange data and instructions, which can lead to increased traffic and latency.

2. **Energy Consumption:** While mobile agents can optimize energy consumption through data aggregation and offloading task, their own movement requires energy. Depending on the mobility patterns and the movement frequency, mobile agents may consume a significant amount of energy during their traversal through the network.

3. **Scalability:** As the size of the WSN grows, the scalability of mobile agents becomes a concern. Coordinating the movement of agents and managing communication with many sensor nodes can become challenging and may lead to inefficiencies.

4. **Agent Coordination and Synchronization:** Proper coordination and synchronization among multiple mobile agents are essential to avoid conflicts, ensure data consistency, and prevent collisions when accessing shared resources. Achieving efficient coordination can be complex, especially in large-scale WSNs.

5. **Security and Trust:** The mobility of agents introduces security risks. Mobile agents can be susceptible to attacks during their movement, leading to potential data tampering, unauthorized access, or even hijacking of agents by malicious entities. Ensuring the trustworthiness of mobile agents is crucial to maintaining the overall network security.

6. **Mobility-Induced Data Staleness:** Implementing mobile agents may require additional infrastructure and support, such as agent management systems, hosts,

and migration protocols. Setting up and maintaining this infrastructure can be resource-intensive and complex.

7. **Programming and algorithm Complexity:** Designing efficient and robust mobile agent-based algorithms for specific applications can be challenging. Developing sophisticated agent behaviors and strategies may require significant programming effort and the host's expertise.

8. **Network Partitioning:** In the case of network partitioning, where parts of the WSN become disconnected, mobile agents may face challenges in traversing these partitions, leading to data loss or inefficiencies in data collection. Addressing above limitations requires careful consideration of factors such as agent mobility patterns, energy management, security mechanisms, communication protocols, and agent coordination strategies and maximize the benefits of mobile agent deployment in WSNs.. Despite these challenges, mobile agents remain a promising approach to improving the efficiency and adaptability of WSNs in various applications.

3.4 Challenges and Considerations using Mobile Agents in WSNs

Using mobile agents in wireless sensor networks can be a good option, depending on the specific application and network requirements. It offers several advancements that can enhance the overall performance and efficiency of the network. However, it also comes with challenges and considerations that must be addressed.

1. **Communication Overhead:** The deployment of mobile agents introduces an additional agent's communication overhead, which may affect the overall network performance and latency.

2. **Energy Consumption:** While mobile agents can optimize energy usage, their movement consumes energy, and proper energy management is essential to avoid negating the benefits of aggregation.

3. **Security :** Security is a critical concern in mobile agent deployment. Mobile agents can become targets of attacks or be compromised, potentially leading to security breaches in the network.

4. **Coordination and Synchronization:** Coordinating the movement of mobile agents and managing data consistency can be complex, especially in large-scale WSNs.

5. **Section mark Scalability:** The scalability of mobile agent-based systems may be challenging as the network size increases.
6. **Programming Complexity:** Developing efficient and robust mobile agent-based algorithms requires expertise and programming efforts.
7. **Network Partitioning:** Mobile agents may face challenges traversing network partitions, affecting data collection and distribution.

In wireless sensor networks (WSNs), where MA migrates among sensor nodes (SNs) for data fusion as compared to the traditional client/server paradigm, where each SN in the network sends its collected data to the sink, Mobile agent (MA) presents a good alternative regarding energy consumption, response time, and network lifetime. In this paradigm, the most critical property of MA is itinerary planning; it has always been an issue and an NP-hard problem.MA itinerary planning techniques can be classified into three categories: static itinerary planning, dynamic itinerary planning, and hybrid itinerary planning. The benefits and shortcomings of different MA itinerary planning approaches are presented. Furthermore, we implement, simulate and compare the most prominent itinerary planning algorithms upon a common parameter space, making realistic agents-level assumptions[16]

Security in large-scale wireless sensor networks (WSNs) is of paramount importance, especially when mobile agents are employed to enhance network efficiency and data processing capabilities. We explore the unique challenges and risks associated with securing mobile agents in the context of vast and dynamic sensor networks. Large-scale WSNs often operate in resource-constrained and hostile environments, making them susceptible to various security threats, including node compromise, data tampering, and unauthorized access.

We analyze the potential vulnerabilities arising from mobile agent movement communication and data aggregation processes and discuss the implications of security breaches on network performance and reliability. To address these challenges, we review state-of-the-art security mechanisms, including secure agent migration protocols, cryptography techniques for data protection, and trust management models for agent authentication and authorization. Additionally, we examine the trade-offs between security requirements and resource constraints in large-scale WSNs and propose context-aware security strategies to strike a balance between protection and energy efficiency.

This abstract aims to provide an overview of the key security considerations in using mobile agents in large-scale wireless sensor networks. It serves as a guide for researchers and practitioners in developing robust and scalable security solutions to safeguard mobile

agent operations in expansive WSN deployments.

Security is critical when executable code is transferred across a network, and security alone is a significant reason for the lack of use of mobile agents. Security is an important issue for mobile agents because the agents are roaming in the network and can be used as malicious objects to access private and confidential information and resources, sometimes causing serious problems. The lack of security has been highlighted mainly because of the lack of approaches to so many things that have to be secured about a mobile agent system [17].

Malicious or badly written code could wreak havoc when unleashed upon an unsuspecting host, and agents themselves need protection against hostile hosts that would seek to dissect or modify those [18].

There is no magic solution that will solve all the security problems of mobile agents, but precautions can be taken to minimize risk.

When an agent leaves for a new host, extreme care must be taken to prevent unauthorized modification or analysis of the agent. Agents may carry with them confidential or sensitive information and logic, which shouldn't be accessible to the agent host.

Since mobile agents can travel to another system in a network and can execute there by consuming remote host resources, they are open to several attacks and abuses, i.e., they amplify the threats of abuse and misuse due to their mobility and execution on different platforms.

Mobile agent systems provide a computing infrastructure upon which mobile agents belonging to different and potentially untrusted users can execute. The communication medium is inherently insecure, and the different agents and agent systems may have conflicting objectives. In this scenario, a variety of attacks can be conceived. Unauthorized users may eavesdrop on network traffic and observe agents in action, or worse, an active intruder may modify the code, data, or state of an agent in transit. Agents may attack the agent platform (host), supporting their execution to gain unauthorized access to resources.

3.5 Mobile Agents Itinerary Planning Techniques in WSNs

It involves strategies to optimize the movement of mobile agents within the network while considering factors such as energy consumption, data collection efficiency, network

coverage, and communication overhead. Here are some commonly used itinerary planning techniques for mobile agents in WSNs:

1. **Greedy Algorithms**
   Greedy algorithms focus on making locally optimal decisions at each step to achieve a global optimization. These algorithms are often used in simple itinerary planning, where agents move from one sensor node to the next based on certain criteria such as energy level, data availability, or distance.

2. **Ant Colony Optimization (ACO)**
   ACO is inspired by the foraging behavior of ants. Agents deposit pheromones on their paths, and other agents use these pheromone trails to decide their routes. ACO-based itinerary planning can help agents find paths that are efficient in terms of energy and data collection.

3. **Particle Swarm Optimization (PSO)**
   PSO mimics the social behavior of birds flocking or fish schooling. In the context of mobile agents, PSO can be used to optimize agent movement patterns by adjusting their velocities and positions based on their own experiences and the experiences of other agents.

4. **Reinforcement Learning**
   Reinforcement learning involves agents learning from their actions and the outcomes of those actions. Agents in WSNs can use reinforcement learning to adapt their movement patterns based on the quality of data collected, energy consumption, and other relevant metrics.

5. **Genetic Algorithms (GA)**
   GA is an optimization technique that uses principles of natural selection and genetics. Agents' movement plans can be encoded as chromosomes, and genetic operations such as selection, crossover, and mutation can be applied to find optimal or near optimal routes.

6. **Fuzzy Logic**
   Fuzzy logic can be used to define rules and decision-making strategies for agent movement. Fuzzy rules can consider factors like sensor data quality, energy levels, and communication costs to guide the agents' paths.

7. **Dynamic Programming**
   Dynamic programming can be employed to find the optimal sequence of movements for agents by considering all possible paths and evaluating their costs. This

technique can be computationally expensive but can guarantee optimal solutions.

8. **Multi-Objective Optimization**
   This technique considers multiple objectives such as energy consumption, data collection efficiency, and network coverage simultaneously. Agents aim to find Pareto-optimal solutions that balance these objectives.

9. **Heuristic Methods**
   Heuristic methods use domain-specific knowledge and rules to guide the agents' movement. These methods are often simple and quick to implement, making them suitable for real-time itinerary planning.

10. **Hybrid Approaches**
    Hybrid approaches combine multiple techniques to capitalize on their strengths. For instance, a hybrid approach might use genetic algorithms for coarse-grained path optimization and then use local search techniques for fine-tuning.

It's important to note that the choice of itinerary planning technique depends on the specific goals, constraints, and characteristics of the WSN application. Additionally, due to the dynamic nature of WSNs, techniques that can adapt to changing conditions and uncertainties are often preferred.

Recent research has underscored the vital importance of implementing robust security measures for mobile agents, given their increasing role in diverse computing applications. The dynamic and autonomous nature of mobile agents introduces unique challenges to ensure their safe operation. One prevalent approach is the adoption of encryption techniques to secure data transmission and agent mobility.

Additionally, multi-factor authentication methods are being explored to verify the identity of agents and prevent unauthorized access. Intrusion detection systems, leveraging machine learning algorithms, are gaining traction to identify and mitigate potential threats to mobile agents while in transit or during execution. Research is also delving into secure code execution environments, leveraging containerization and sandboxing to limit an agent's access to system resources. As the landscape of mobile agent applications continues to expand, a holistic security framework encompassing encryption, authentication, intrusion detection, and controlled execution environments is essential to mitigate risks and foster the trustworthy deployment of mobile agents.

3.6 Security Issues

Security issues associated with deploying mobile agents in Wireless Sensor Networks (WSNs) are a critical concern that demands careful attention. The nature of mobile agents, which autonomously move through network nodes to execute tasks, introduces vulnerabilities that can compromise the integrity, confidentiality, and overall functionality of the network. One primary concern is the potential for code tampering, where malicious agents may alter their behavior or inject malicious code during migration. This can lead to unauthorized data access, disruptions in data aggregation, and even the propagation of malware throughout the network.

Additionally, the mobility of agents increases the risk of unauthorized access to sensitive data as agents traverse nodes with varying levels of security. Communication channels used by agents can be exploited for eavesdropping, potentially exposing critical information.

Another substantial threat is the possibility of malicious agent attacks, where adversaries deploy agents with harmful intentions, such as data exfiltration, denial of service, or node compromise. These agents can undermine the network's operation, compromise data integrity, and consume valuable resources. Furthermore, the inherent mobility of agents can lead to data inconsistencies, especially in scenarios where agents aggregate data from multiple nodes, potentially leading to inaccurate decisions and analyses. Network partitioning can also occur during agent migration, posing challenges in maintaining communication and data integrity across disconnected segments of the network.

Mitigating these security issues requires the implementation of robust mechanisms. Secure code mobility techniques can help ensure the integrity of agents during migration. Encryption and authentication protocols can safeguard agent communication and data exchange. Trust management models play a pivotal role in establishing the authenticity of agents and nodes, allowing for secure interaction. Access control measures can restrict unauthorized agents from accessing sensitive resources. Intrusion detection and prevention systems can aid in identifying and mitigating malicious agent attacks.

In conclusion, the integration of mobile agents in WSNs brings forth a range of security challenges stemming from their mobility, communication, and potential for malicious intent. Addressing these issues is crucial to maintaining the reliability and security of the network. Effective solutions involve a combination of secure migration protocols, cryptographic techniques, trust-based models, and vigilant monitoring to ensure the safe and efficient operation of mobile agents in WSNs.

To be more precise, the security concerns revolve around:

1. **Authentication of the user:** Who is responsible for this script? Who is accountable for any charges it runs up?

2. **Authorization of the user:** Can this user execute this script? Which operations on which objects is this user allowed to invoke? What resources can this script consume?

3. **Virus propagation:** Can agents corrupt the underlying system or processes that they come in contact with?

4. **Trojan horses:** Since Telescript does inter-process communication by passing objects around, and the only way to "use" an object is to invoke an operation on it, how does one know whether or not an arbitrary class contains malicious code?

3.7 Security Threats

Mobile agents in Wireless Sensor Networks (WSNs) face various security threats due to their autonomous mobility and interaction with network nodes.

1. **Code Tampering and Modification:** Malicious agents or attackers can tamper with the code of mobile agents during migration, altering their intended behavior or injecting malicious code. This can lead to unauthorized data access, disruption of data aggregation, or the propagation of malware within the network.

2. **Unauthorized Data Access:** Mobile agents may inadvertently or maliciously access data on sensor nodes without proper authorization. This can result in privacy breaches and unauthorized exposure of sensitive information.

3. **Data Manipulation:** Attackers can modify or manipulate data collected or processed by mobile agents. This can lead to false conclusions, inaccurate analyses, and decisions based on manipulated data.

4. **Eavesdropping and Data Interception:** The communication channels used by mobile agents to interact with nodes can be vulnerable to eavesdropping. Attackers can intercept sensitive data, compromising its confidentiality.

5. **Malicious Agent Attacks:** Adversaries may deploy malicious agents that aim to disrupt the network's operation. These agents can engage in denial-of-service attacks, consume network resources, or even compromise nodes.

6. **Node Compromise:** Mobile agents might migrate to compromised nodes, leading to potential node compromise or takeover. This can provide attackers with a platform to launch further attacks within the network.

7. **Man-in-the-Middle Attacks:** Attackers can position themselves between the mobile agents and target nodes, intercepting and altering communication between them. This enables unauthorized access or data manipulation.

8. **Network Partitioning:** During agent migration, the network might partition due to communication failures or physical barriers. This can lead to data loss, incomplete operations, and difficulties in agent coordination.

9. **Resource Exhaustion:** Mobile agents can be designed to exhaust the resources of the nodes they visit, leading to the depletion of energy, memory, or processing power. This can disrupt the normal operation of nodes.

10. **Location Privacy Concerns:** The movement of mobile agents can potentially expose the location information of nodes. This can be exploited by attackers to track or identify sensitive locations.

11. **Replay Attacks:** Attackers can record and replay actions performed by mobile agents. This can result in duplicate or unintended operations being carried out in the network.

12. **Trustworthiness Issues:** Assessing the trustworthiness of mobile agents and the nodes they interact with can be challenging. An agent might unknowingly interact with compromised or untrusted nodes. Addressing the above security threats requires a combination of measures, including secure code mobility techniques, encryption for communication, access control mechanisms, intrusion detection systems, and trusted computing concepts. Proper authentication, authorization, and monitoring mechanisms are also crucial to ensure the integrity, confidentiality, and availability of data and operations involving mobile agents in WSNs.

3.7.1 Security Requirements of Mobile Agents

The users of networked computer systems have some security requirements: confidentiality, integrity, availability, anonymity, and accountability. The users of agent and mobile agent frameworks also have these same security requirements. This section provides a brief overview of these security requirements and how they apply to agent frameworks. Mobile agents should be protected while they are in transit from one host to another host. Communication between agents and users, as well as between agents themselves, should

also be protected. All four security requirements are required to provide secure agents and platforms to communicate. An agent and platform must keep sensitive information confidential, integrity, availability, anonymity, and accountability

1. **Confidentiality**

   Any private data stored on a platform or from an agent must remain confidential. Agent frameworks must be able to ensure that their intra and inter-platform communications remain confidential. Eavesdroppers can gather information about an agent's activities from the content of the messages exchanged and from the message flow from one agent to another. Monitoring the message flow may allow other agents to infer helpful information without having access to the actual message content.

2. **Integrity**

   The agent platform must protect agents from unauthorized modification of their code, state, and data and ensure that only authorized agents or processes carry out any modification of shared data. The agent itself cannot prevent a malicious agent platform from tampering with its code, state, or data, but the agent can take measures to detect this tampering. The secure operation of mobile agent systems also depends on the integrity of the local and remote agent platforms.

   A malicious host can easily compromise the integrity of a mobile agent visiting a remote agent platform. The trend toward releasing open-source platforms and operating systems makes it easier for unscrupulous administrators or organizations to make unauthorized modifications to the agent platform and the underlying framework. The agent's user and developer may need to learn of the behavior of this modified framework and the effects it will have on the agent's code, state, and data that are under the platform's complete control.

3. **Accountability**

   Accountability requires maintaining an audit log of security-relevant events and listing each event and the agent or process responsible for that event. Each cycle, human user, or agent on a platform must be held accountable for their actions.

   For each process, the human user or agent must be uniquely identified, authenticated, and audited to be held accountable. Examples of actions for which they must be held accountable include access to an object, such as a file, or administrative changes to a platform security mechanism. Security-relevant events are defined in the platform security policy. They should include, at a minimum, the following:

- User/agent name
- Time of the event
- Type of event
- Success or failure of the event
- Audit logs must be protected from unauthorized access and modification.

4. **Availability**

   The agent platform must ensure the availability of data and services to local and remote agents. The agent platform must be able to provide controlled concurrency, support for simultaneous access, deadlock management, and exclusive access as required. Shared data must be available in a usable form, capacity must be available to meet service needs, and provisions for the fair allocation of resources and timeliness of service must be made. The agent platform must be able to detect and recover from system software and hardware failures. While the platform can provide some level of fault tolerance and recovery, agents may be required to assume responsibility for their fault recovery.

   The agent platform may need to balance an agent's need for privacy with the platform's need to hold the agent accountable for its actions. The platform may be able to keep the agent's identity secret from other agents and maintain a form of reversible anonymity where it can determine the agent's identity if necessary and legal. Anonymity in human communities may help foster certain types of commerce or promote the freedom of expression by eliminating the fear of retribution for voicing unpopular viewpoints.

3.7.2 The Problem of Malicious Hosts

In the context of mobile agents in Wireless Sensor Networks (WSNs), the problem of malicious hosts refers to the major security challenge posed by compromised or malicious nodes that the mobile agent interacts with during its movement through the network. These malicious hosts can manipulate the agent's behavior, alter its data, or attempt to extract sensitive information, thereby compromising the integrity, confidentiality, and overall functionality of the agent and the network. This issue can have significant implications for the security and reliability of the WSN. Here are some key aspects of the problem:.

1. **Data Tampering and Manipulation:** Malicious hosts can modify the data car-

ried by the mobile agent, leading to inaccuracies in the collected information. This can result in incorrect decisions, faulty data aggregation, and unreliable network operation.

2. **Unauthorized Data Access:** Compromised hosts might gain unauthorized access to the data stored within the mobile agent. This can lead to data breaches and compromise the privacy of sensitive information.

3. **Code Injection:** Malicious hosts can inject malicious code into the mobile agent as it passes through their territory. This injected code could propagate malware throughout the network, potentially affecting other nodes and agents.

4. **Repudiation of Actions:** Malicious hosts can alter the actions and interactions of the mobile agent, making it difficult to establish the true origin or intent of the agent's operations. This can create challenges in tracking and attributing malicious actions.

5. **Resource Depletion:** Malicious hosts may deliberately consume the resources of the mobile agent, such as its energy or memory, causing it to fail prematurely and disrupting its intended functions.

6. **Sensitive Information Leakage:** Compromised hosts might attempt to extract sensitive data from the mobile agent, potentially leading to the exposure of confidential information or sensitive algorithms. Addressing the problem of malicious hosts in mobile agents during WSNs requires a combination of security measures.

7. **Authentication and Authorization:** Implement strong authentication mechanisms to ensure that the agent interacts only with trusted hosts. Authorization mechanisms can restrict the operations the agent can perform on a host.

8. **Secure Communication:** Employ secure communication protocols to protect interactions between the mobile agent and hosts. Encryption can prevent eavesdropping and data manipulation.

9. **Code Integrity:** Use techniques like code signing or checksum verification to ensure the integrity of the mobile agent's code.

10. **Trust Management:** Utilize trust-based models to assess the trustworthiness of hosts. Monitoring and evaluating hosts' behavior can help identify potential threats.

11. **Isolation:** Consider isolating the execution of mobile agents from the underlying host environment using virtualization or sandboxing techniques, preventing direct access to host resources.

The problem of malicious hosts underscores the importance of robust security mechanisms to protect mobile agents as they traverse through the WSN. By addressing this challenge, the network can maintain the integrity of data, operations, and communication even in the presence of compromised or malicious nodes. To achieve a protection level that is comparable to the one of agents that run on non-malicious or trusted hosts, the following attacks are identified:-

- Spying out code
- Spying out data
- Spying out control flow
- Manipulation of code
- Manipulation of data
- Manipulation of control flow
- Incorrect execution of code
- Masquerading of the host
- Denial of execution
- Spying out interaction with other agents
- Manipulation of interaction with other agents
- Returning wrong results of system calls issued by the agent

3.7.3 Counter Measure

Many conventional security techniques used in contemporary distributed applications (e.g., client-server) also have utility as countermeasures within the mobile agent paradigm. Moreover, there are a number of extensions to conventional techniques and techniques devised specifically for controlling mobile code and executable content (e.g., Java applets) that are applicable to mobile agent security (Jansen). After reviewing these countermeasures, mobile agent computing allows for both prevention and detection mechanisms techniques. These techniques can be used to protect agents as well as platforms, separately from those used to protect the agents that run on them. These countermeasures have been implemented or suggested to reduce the vulnerability of the mobile agent against malicious hosts. As with security countermeasures, Prevention mechanisms aim to protect the mobile agent so that it becomes difficult, or at least very expensive, to

attack the agent. The detection mechanisms perform (regular) checks to discover possible security breaches. The Table 3.1 indicates the countermeasures that are useful in detection and which are more appropriate for prevention.

Table 3.1: Categorization of Mobile Agent Countermeasures [19].

SN	Countermeasures	Category
1	Partial result encapsulation	Detection
2	Mutual itinerary recording	Detection
3	Replication and voting	Detection
4	Execution tracing	Detection
5	Digital Signatures	Detection
6	State Appraisal	Detection
7	Migration Paths	Detection
8	Replication and Voting	Detection
9	Execution Monitoring	Detection
10	Reference States	Detection
11	Sedentary Agents	Detection
12	Environmental key generation	Prevention
13	Computing with encrypted function	Prevention
14	Code Obfuscation	Prevention
15	Computing with Encrypted Functions	Prevention

3.8 Summary

This Chapter presents a detailed exploration of mobile agents in WSNs, highlighting their key roles and applications in enhancing network performance and functionality. By addressing both the advantages and challenges associated with mobile agents, the chapter provides a balanced view of their impact on wireless sensor networks. Through the discussion of practical uses, limitations, and security considerations, this chapter aims to offer insights into the effective deployment and management of mobile agents in modern sensor networks.

Table 3.2: Security Threats and its Countermeasures

SN	Security Threats	Countermeasure	Ref
1	**Malicious Code Injection** Mobile agents may carry malicious code or payloads that can compromise the host system or network. This code injection can lead to unauthorized access, data leakage, or disruption of services.	Code validation and verification techniques can be employed to ensure that the mobile agent's code is safe and free from malicious elements. Techniques such as code signing, integrity checking, and sand boxing can be utilized to mitigate the risks of code injection.	[20]
2	**Unauthorized Access and Privilege Escalation** Mobile agents may attempt to access or modify sensitive data or resources without authorization. They may also try to escalate their privileges to gain higher access levels than intended.	Access control mechanisms should be implemented to enforce proper authentication, authorization, and privilege management. Techniques such as encryption, robust authentication protocols, and secure communication channels can be employed to protect against unauthorized access and privilege escalation	[21]
3	**Network Eavesdropping and Tampering** Mobile agents may be vulnerable to eavesdropping attacks, where attackers intercept and analyze the communication between agents and hosts. Additionally, attackers may tamper with the agent's messages or data, leading to information leakage or manipulation.	Secure communication protocols such as SSL/TLS can be utilized to encrypt the communication between mobile agents and hosts, ensuring confidentiality and integrity. Digital signatures and message authentication codes can be employed to detect tampering attempts.	[22]
4	**Denial-of-Service (DoS) Attacks** Mobile agents can be targeted with DoS attacks to disrupt their execution or prevent them from completing tasks. Attackers may overload the agent's resources, exploit vulnerabilities, or exhaust network bandwidth.	Implementing robust intrusion detection and prevention systems can help identify and mitigate DoS attacks. Load balancing techniques, resource monitoring, and rate-limiting mechanisms can be utilized to mitigate the impact of DoS attacks on mobile agents.	[23]

Chapter 4

Mobile Agent-Based Intrusion Detection System

This chapter explores Mobile Agent-Based Intrusion Detection Systems (IDS), focusing on how mobile agents enhance the security and effectiveness of intrusion detection in various network environments. It provides a detailed overview of IDS concepts, the role of mobile agents, and the implementation and evaluation of an automated attack classification model.

An Intrusion Detection System (IDS) is a cyber security solution designed to monitor network traffic, system activities, and user behaviors in order to identify and respond to unauthorized or malicious activities. Its primary purpose is to detect and alert administrators about potential security breaches, attacks, or policy violations within a computer system or network. The preservation of data integrity, network security, and system confidentiality is of paramount importance. As organizations increasingly rely on technology for their operations, the potential for security breaches and unauthorized access rises.

This is where Intrusion Detection Systems (IDS) step in as a crucial line of defense. An IDS serves as a vigilant sentinel, tirelessly monitoring networks, systems, and applications to detect any signs of unauthorized or malicious activities. An IDS can swiftly identify potential security breaches, ranging from sophisticated cyber attacks to inadvertent policy violations, by analyzing patterns, behaviors, and anomalies within the network traffic or system logs. This proactive approach allows organizations to promptly respond to threats, mitigating potential damages and safeguarding sensitive information.

Whether it's a network-based IDS scanning data packets for telltale signs of intrusion or a host-based IDS scrutinizing system activities for unauthorized access attempts, IDS technology plays a pivotal role in maintaining the integrity and security of digital ecosystems. Its ability to provide real-time alerts, facilitate incident investigations, and contribute to compliance efforts makes it an indispensable tool in the arsenal of modern cyber security strategies. As the cyber landscape evolves, the role of IDS remains ever-relevant, enabling organizations to stay one step ahead of potential threats and ensuring the robustness of

their digital infrastructure.

4.1 Intrusion Detection System

An Intrusion Detection System (IDS) is a vital tool that continually monitors and analyzes network or system activities for any signs of malicious activities or policy violations. The main objective of an IDS is to swiftly detect and alert potential security breaches, intrusions, or any other unwanted activities in real-time or near real-time. Here are the primary types and functions of IDS:

1. **Type of IDS**
    - **Network-based Intrusion Detection System (NIDS):** NIDS monitors network traffic, analyzing data packets in real time to identify suspicious patterns or anomalies. It looks for signs of known attack signatures or behaviors that deviate from normal network activity. NIDS can be placed at key points within the network to capture and analyze traffic.
    - **Host-based Intrusion Detection System (HIDS):** HIDS operates at the individual host level, monitoring activities on a specific system or device. It examines system logs, file integrity, and other host-specific information to detect any unusual behavior or unauthorized access.
2. **Functions of IDS:** The functioning of an IDS involves these key steps:
    - **Monitoring** The IDS constantly collects and analyzes data from network traffic or host activities.
    - **Detection** The collected data is compared against known attack signatures, predefined rules, or behavioral patterns. If any match is found, the system triggers an alert.
    - **Alerting** When a potential intrusion is detected, the IDS generates alerts or notifications, which can be sent to system administrators, security personnel, or other designated parties.
    - **Response** Depending on the setup, administrators can take actions based on the alerts. These actions might include further investigation, isolating affected systems, and implementing countermeasures.

Intrusion detection systems are crucial in enhancing an organization's overall cyber security posture by detecting unauthorized activities in a timely manner. They help iden-

tify various types of threats, such as malware infections, unauthorized access attempts, brute-force attacks, and more. IDS also aids in understanding attack trends and patterns, enabling organizations to implement proactive measures to prevent future incidents.

It's important to note that while IDS is a powerful tool for detecting potential intrusions, it may also generate false positives (indicating an attack that didn't actually occur) or false negatives (missing a genuine attack). Therefore, IDS is often used with other cyber security measures, such as firewalls, antivirus software, and Intrusion Prevention Systems (IPS), to protect against cyber threats.

4.2 An Ensemble Model for Automated Attack Classification

The most recent technological development is the convergence of mobile agent technology for high-level inference and surveillance in wireless sensor networks. Connecting the two technologies advances new fields and is becoming more important in communication. The sensor nodes become more open to malicious attacks as they move from a static to a dynamic environment that changes quickly. This research looks into the attacks that cause communication to break down and sounds the alert for the necessary response. One of the most helpful data-centric routing protocols in WSNs is the SPIN protocol. A mechanism has been developed to collect enough information from the Network Simulator (NS-2.35) on the necessary properties to spot unusual behavior on sensor nodes. This paper provides a novel method for categorizing attacks based on wasted energy using several criteria.

Additionally, common classifiers with machine learning models build a performance matrix according to the type of attack. An ensemble model with a 95 % average accuracy is proposed to improve the performance of the ensemble classifiers. Additionally, K Fold cross-validation was done to make sure the proposed ensemble model was consistent.

Research into wireless sensor networks has exploded recently due to the widespread interest in the enticing technologies they employ, which enable and support the next generation of ubiquitous and pervasive computing scenarios. Large numbers of self-sufficient sensor nodes are spread across an area of interest and programmed to carry out certain tasks.

The current mobile agent-based Wireless Sensor Networks (WSN) implementations are considered an incremental evolution in energy efficiency and data delivery in a region using sensor nodes. These sensor nodes interact wirelessly and do not require shared

infrastructure or centralized control. Nodes may collaborate by forwarding or relaying packets to one another, potentially involving numerous intermediate relay nodes. Because of the distributed nature of learning and decision-making in Wireless Area Networks, this study pioneered the use of mobile agents to implement intrusion detection.

As demonstrated in Figure 4.1, sensor agents can conduct local detection utilizing attack attributes available inside the limited sensing zone.

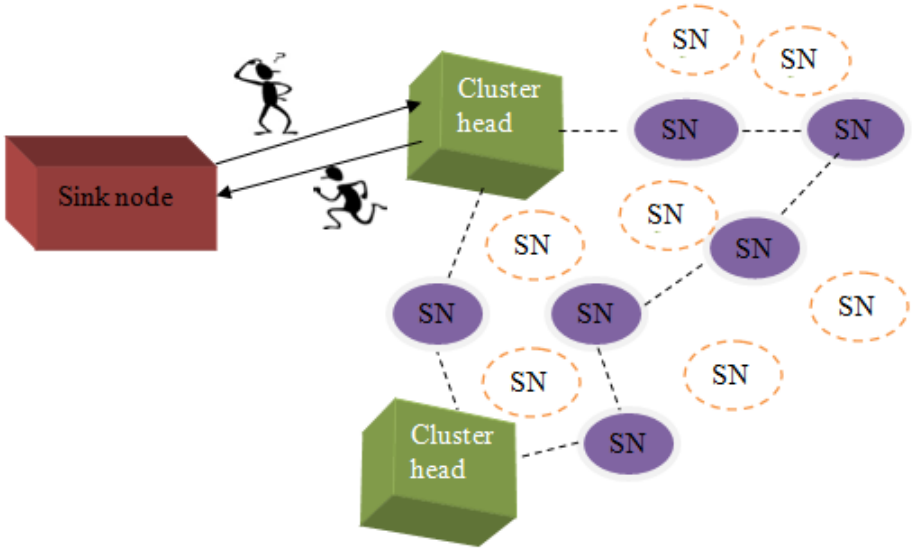


Fig 4.1: Structure of WSN

A method for collecting data on network parameters using modeling and simulation using Network Simulator-2 (NS-2.35). A WSN has many mobile agents that operate as autonomous sensor nodes and are placed in different clusters throughout different regions to execute work independently toward a stated objective while interacting with other agents and their surroundings. WSN is being implemented for task-specific monitoring and agent security.

Typically, these networks are made up of sensor nodes that are fixedly placed at predetermined positions across the environment and continuously monitor one or more sensors to look for any unusual activity. The majority of security networks do not gather data, which is a key distinction between security and environmental WSNs. The WSN must acknowledge any nodes that are disabled or stop functioning as a security breach that needs to be notified. In a security WSN, nodes spend the majority of their energy trying to achieve the strict latency criteria necessary to sound an alarm if and when security violations occur.

If a security violation is found, a report must be created and sent immediately to the sink or base station. The performance of the application is significantly impacted by the

latency of these transmissions from the nodes to the network sink. In order to improve performance and attack identification with ensemble models, mobile agent technology still has a way to go toward reaching the ideal behavior. This can be accomplished with the use of machine learning models. Mobile agent technology has the ability to develop a foothold and increase its reach, allowing for the gradual adoption and usage of cutting-edge technology to improve accuracy.

To become an effective enabling technology on Wireless Sensor Networks (WSNs), mobile agents have qualities such as robust itinerary and order with a distributed computing paradigm based on code mobility. In this research, we build a unique dataset to describe four distinct types of DoS attacks, as well as baseline behavior [24]An approach is placed for developing an Intrusion Detection System (IDS) for autonomous mobile agents. Every node in the network counts as the computing node, and mobile agents migrate and collaboratively perform attack detection [25].

The IDS must be able to detect the greatest number of security threats possible and be consistent with the characteristics of WSNs. Test-Data is the name of the created dataset.

Further, random parameters like network lifetime, consumed energy, throughput, and Packet Delivery Ratio (PDR) determine the dependability of generating attacks on a network. This work considers consumed energy as a vital dependability enhancement feature. The contribution of work is as follows:

1. Create a model of the proposed IDS using a Network Simulator (NS-2.35), simulate it, and then analyze the simulated data to build the dataset, which will be used to detect attacks based on various factors in WSNs.

2. To optimize the generated dataset with data pre-processing.

3. Implement and perform prediction on an ensemble model.

The novelty of this work is predicting the behavior of mobile agents on WSN, which is a learning model that can be used to identify the attacks before allowing for further communication, opening a new avenue for the research fraternity. This entire paper is organized as follows: Section II describes the related work as background information. Section III illustrated the modeling and simulation. The proposed model is presented in Section IV. The model evaluation with hyperparameter is in Section V. Section VI discusses the result analysis and comparison with the available model. The conclusion and future scope are discussed in Section VII.

4.3 Background and Related work

This Section presents an overview of the literature survey in four phases. The first phase deals with the routing algorithm for data evaluation followed by the mobile agent IDS on Wireless Sensor networks, simulation of data on the basis of a performance parameter of nodes for attack identification, and comparison with no attack parameter to design an ensemble model using machine learning model concludes with the comparison of performance with available machine learning models.

4.3.1 SPIN Protocol Overview

An energy-constrained wireless sensor network considers Sensor Protocols for Information through Negotiation (SPIN), a negotiation-based data-centric routing protocol sensors in an energy-constrained wireless sensor network [26].

The protocol starts when a SPIN node obtains new data to share. It broadcasts an ADV message containing meta-data. If a neighbor node is interested in data, a REQ message is sent for the DATA. The neighbor sensor node then repeats this process to its neighbors, resulting in the entire sensor area getting a copy [27]. SPIN is a data-centric routing protocol. A meta-data is used, which is exchanged among the sensor nodes. A node that sensed some new data sends an advertisement message containing the Metadata to all its neighbors [28]. In SPIN, three types of messages are used to communicate :

- ADV message.
- REQ message.
- DATA message.

SPIN assumes that the Base Station (BS) is fixed and located far from sensor nodes. Figure 4.2 shows the structure of nodes in a SPIN routing protocol.

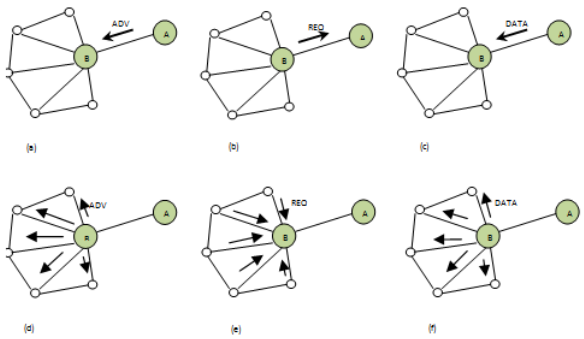


Fig 4.2: Structure of SPIN Protocol

4.3.2 Mobile Agents in WSNs

Mobile agents for efficient data dissemination in WSNs, the occurrence of certain events will alert sensors to collect data and send them to sink node with the strategies on the sensor network have proposed [29]. The development of WSN application based on the mobile agent paradigm and how WSN has posed its effectiveness in a highly distributed environment [30].

A mobile agent paradigm proposed to reduce and aggregate data in a planar sensor network as agents reduce the communication cost, especially over low bandwidth links [31].

A mobile agent-based middleware using a publish mechanism in wireless sensor networks and the execution time and energy consumption results show that the proposed model is more effective during agent transmission [32].

A model was proposed and compared with the traditional cryptography mechanism to identify the weak security in the wireless environment. Further, use mobile agents in the model to provide dependable Internet services delivery to users; this will guarantee secure authentication in wireless networks and examine the feasibility of the solution with a model of WSN [33].

Using mobile agents as enabling technology for Wireless Sensor Networks via encapsulate protocols and dynamically deploy them to overcome standardization or upgrading security issues, created and migrated into the sensor nodes of the target WSN zone [34].

A middleware is presented to support remote-level communication in different tuples, integrating mobile agents into the multicasting wireless sensor networks. It enhanced the mobile agents for supporting multiple concurrent applications to support dynamic changes and support large-scale networks for increasing scalability and heterogeneity [35].

4.4 Mobile Agents based IDS in WSNs

A Mobile Agent-based Intrusion Detection System was proposed using a scheme to perform two intrusion detection levels using the minimum possible network resources. The proposed idea enhances network lifetime by reducing the workload on Cluster Head (CH), and it also provides enhanced security in WSN [36].

The concept of a lightweight analysis, energy-efficient system that uses mobile agents on intrusions-based detection for the energy consumption on the sensor nodes. A linear

regression model is applied to predict energy consumption and deployed results under the denial-of-service attacks. Consider flooding can be detected with high accuracy while keeping the number of false positives very low.

Consider the importance of the security of networks. Analyses an autonomous mobile agent-based intrusion detection architecture to address the security in wireless body area networks to highlight the unique benefits of the proposed architecture [37]. As discussed, the detection techniques for attacks used in WSNs to deploy for supervisory control and data acquisition systems [38].

4.5 Machine Learning Model for WSNs

Autonomous mobile agents use a machine learning model to perform detection to detect various security attacks. Simulation results show that the proposed system performs efficiently with high detection accuracy and low energy consumption. A hierarchical and distributed approach to seeing various security attacks in these networks used machine learning classifiers on the mobile agents to detect attacks accurately.

Further investigation and comparison of machine learning classifiers were made, and the most suitable one was chosen to implement IDS. The system was assessed for detection accuracy and energy consumption and achieved credible results with different combinations of adversarial sophistication. The concept of congestion in WSN is well discussed by [39]. The main reason identified is the increased DF rate, which generates enormous traffic on the network. A machine learning model and an intrusion detection system were used to deal with the denial of service attack on the WSN. The performance parameters considered are PDR, energy consumption, detection rate, and delay. The simulation was deployed in NS-2 using the AODV routing protocol [40].

A detailed review from 2002 to 2013 of machine learning techniques used in WSN presented in the discussion states that many research challenges in WSN have been tackled with the help of machine learning techniques. Further, it is concluded that the performance parameters of WSN need to be optimized further for attaining a dependable WSN [41]. Consider an intrusion detection system (IDS) based on the randomized and normalized deployment of nodes on Wireless Sensor Networks (WSNs). Analyses the machine learning techniques to enhance network dependability under flooding attacks. This work found that machine learning models play a significant role in predicting the data flow. The experiments on the simulated dataset underline the part of the machine learning model for data flow prediction on the normalized dataset. These models perform differently according to the size of training and testing dataset partitions [42].

4.6 Assumption and Mathematical Model

This proposed ensemble model uses a mobile agent-based IDS architecture in wireless sensor networks to generate a data set of WSN Test Case data. Figure 4.3 presents the network model of WSN using Mobile Agent-Based IDSs to ensure security in a network.

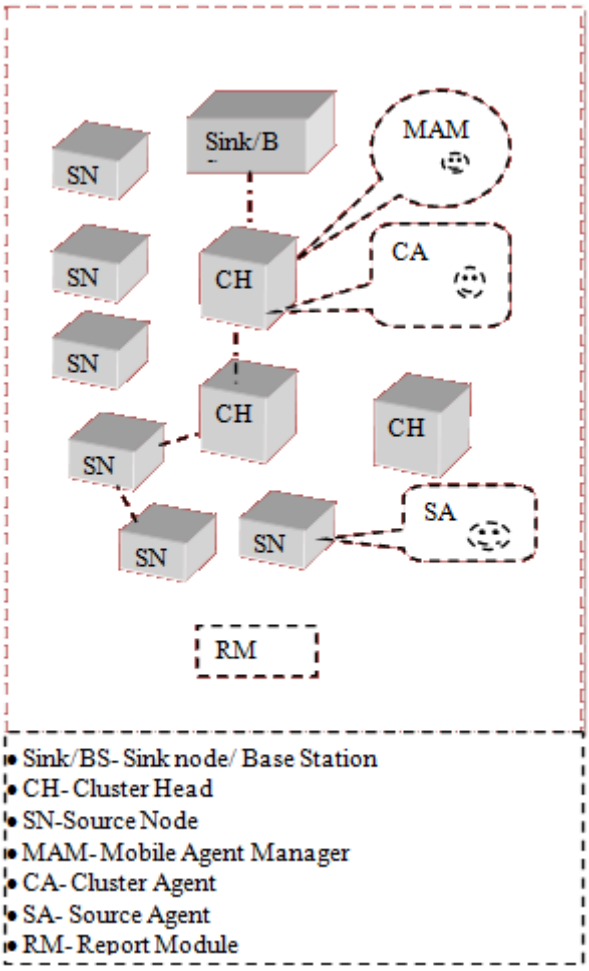


Fig 4.3: WSN with Mobile Agent Based IDS

In this framework, the proposed network consists of wireless sensor nodes such as base nodes, cluster heads, and sensor nodes that monitor, collect, and relay the data to the provided storage. As discussed earlier, an anomaly on the node can occur at any point in the form of an attack. An effective IDS is required as distributed in the network to prevent such attacks. This proposed model uses multiple mobile agent-based intrusion detection systems for wireless sensor networks. Mobile agents-based IDS can detect attacks using features available in the network using network simulation (NS-2.35).

The attack identification technique is designed to cope with nodes based on parameters as

mentioned in the model and generate data named WSN-test case based on consumed energy by nodes to identify Denial of Services (DoS) attacks using machine learning models. Mobile agents-based IDS works on most of IDS's standard components, such as monitoring, analysis, detection, logging, alarming, and prevention. The agent model is designed to be flexible, scalable, platform-independent, and dynamic. The proposed method uses mobile agents and multiple base stations deployed in the WSN to randomize and deploy secured nodes. Consider a Base Station BS consisting of S of sensor nodes s_i, s_{i+1}, . . . s_N, where the total number of sensor nodes is N in the network. The mobile agent-based IDS contains different agents, such as Mobile Agent Supervisor (MAS), Cluster Head Agent (CA), and Sensor Node (SN). A Report Module (RM) is responsible for keeping track of all agents in the network.

The Mobile Agent Manager (MAM) is responsible for releasing and storing updating sensor and cluster agents in the network and acts as a central hub in the network. Sensor and Cluster agents are used to detect anomalies on the respective nodes and perform global detection on a cluster and local detection on a sensor node as suspicious activities. A routing protocol is essential to transmitting the data from source to target nodes and efficiently disseminating information in the network. SPIN (Sensor Protocol for Information via Negotiation) routing algorithm is used in the proposed model from several types of routing algorithms available in WSN. SPIN is a data-centric protocol that dissemination data in the network through negotiation in an energy-constrained network. SPIN uses data negotiation techniques and resource adaptive algorithms to deliver the data in the network.

4.6.1 Mathematical Model

A mathematical analysis has been conducted to ensure the correctness of the generated dataset called the WSN-Test Case. It is calculated by observing the number of messages sent by nodes and received by nodes via mobile agent-based IDS on a specific transmission cycle with the parameters of data that further generate attacks. For the dataset, a mobile agent-based IDS is installed on the sink node (base station) to check the network's behavior and the working abnormality of nodes. It has a life cycle of three phases: collection, detection, and response phase for deployed sensor nodes or both ends. The collection phase gathered data from global as well as local nodes. The collected data was simulated on a network simulator (NS-2.35) and analyzed on the basis of performance parameters. Table 4.4 conducted to all phases further compared to the result of simulation in case of a normal situation where no DoS attack occurred. The terms used in the proposed model are listed as follows:

N: number of sensor nodes in WSN.

i: sensor node i.

SA: number of sensor/ Sensor agents.

CH: No of Cluster Heads.

CA: number of cluster head/cluster head Agent.

ADV-PM: Advertisement-Packet Message Sent.

ADV-RCVD: Advertisement-Packet Message Received.

REQ-PR: Request-Packet Received.

DP: Drop Packet.

PDR: Packet Delivery Ratio.

DF: Data Flow.

PT: Protocol Type.

Base: Central hub/ Sink node.

RM: Report Module.

MAM: Mobile Agent Manager.

TD-SENT: number of TDMA schedules sent by CAS.

TD-RCVD: number of TDMA schedules received by sensor nodes.

There are some basic concepts and assumptions to deploy the above terms in a mathematical model and compare the simulation results with the case of a normal situation when there is no DoS attack.

Each cluster head is responsible for local detection within a clique of each sensor for multiple spawning sensors. Each Mobile agent traverses the nodes within its set itinerary and performs local detection in each node by aggregating the logs accumulated over a period of time.

1. **Advertisement Message:** Calculate the number of advertisement messages sent by sensor nodes and received by CHs in a specific round and the maximum ADV-PM sent within the particular round in CA. The equation used for maximum request via equation 4.1. The procedure used SPIN routing algorithm, SA broadcasts an advertisement message in each round is supposed to the rest of the nodes.

$$REQ - PRis(N-1) * CH \tag{4.1}$$

2. **Request Message:** The number of join request messages is sent by sensor nodes and received by H to associate with them. In the cluster setup phase of SPIN, the maximum REQ-PR sent equals PDR, which is N-CH because once each sensor node has decided which cluster it will belong to, H will receive the message. In the data transmission phase, at the end of each round, BS receives the sensed data received by CHs from the sensor nodes according to their time slot assigned by the schedule; it combines them into a packet and sends them to the CA from CM.

3. **Data message:** Calculate the amount of actual data delivered to the target node at the end of each round. Once the CH receives the sensed data from the sensor node, it negotiates the data and sends it to the target node. Throughout the round, every packet received by nodes is sent to the target node after the data has been negotiated.

Table 4.2 shows the comparison between the mathematical and simulation results to confirm the correctness of the simulation of the collected data. Equations were applied to obtain the mathematical results, while the simulation results were obtained from Network Simulator (NS-2.35).

4.7 WSN Test Case Dataset Workflow Description

To generate the dataset and collect the data from the sent and received packets in WSN, a monitoring service is performed by a mobile agent-based IDS. To identify the possible attacks on the generated dataset, it is necessary to collect such data that help detect, classify, and prevent attacks. Each sensor node will monitor its neighbor node. This Section proposes an ensemble model based on the collected data on a Denial of Services attack. So, multiple machine learning models are used to predict nodes under attack or normal.

4.7.1 Assumption for Proposed Model

The main idea behind the proposed model is based on DoS attacks that have been identified, like Flooding attacks, Blackhole attacks, Wormhole, and Grayhole attacks. The seriousness of the DoS attack originates from the fact that sensor node deployment in

harsh environments is difficult to control. A system is constructed using a machine learning perspective on collected, trained data from network simulation with the following assumptions to solve the problem.

1. The wireless links are active and dynamic.
2. The sensors are densely deployed and can work asynchronously because of mobile agents.
3. It provides a single security checkpoint, i.e., a mobile agent for effective WSN.
4. Mobile agents are used to segregate the faulty sensor nodes at the initial state from normal sensor nodes and improve the efficiency of the network.
5. Perform attacks by dropping packets to identify attack types like drop all packets, randomly drop packets, or selectively drop packets.

The proposed model generates a dataset using the following rules:

1. Mobile agent-based IDS works as a monitoring service to build a specialized dataset and collect the data from the sending and receiving packets in WSN using mobile agent migrating algorithm Figure 4.4.
2. The sensor node transmits a packet and checks the consumed energy in the previous round on the basis of distance, received signal strength, current energy of node, number of messages sent, received, and drop to sensor agent. An attack is identified on the basis of consumed energy by the node during connectivity (Refer Table 4.1).
3. In order to gather the required data, Network Simulator (NS-2.35) is used as per the simulation parameter as mentioned in Table 4.3.
4. The classification of attacks generated on the above parameter and feature have been grouped for attack identification as in Table 4.1.

Table 4.1: Sample of Wireless Sensor Network dataset [24]

Time	Is _CH	who CH	Dist _To_CH	ADV _S	ADV _R	JOIN _S	JOIN _R	SCH _S	SCH _R	Rank	DATA _S	DATA _R	Data _To_BS	dist_CH _To_BS	send _code	Consumed Energy	Attack type
50	1	101000	0	1	0	0	25	1	0	0	0	1200	48	130.08535	0	2.4694	Normal
50	0	101044	75.32345	0	4	1	0	0	1	2	38	0	0	0	4	0.06957	Normal
50	0	101010	46.95453	0	4	1	0	0	1	19	41	0	0	0	3	0.06898	Normal
50	0	101044	64.85231	0	4	1	0	0	1	16	38	0	0	0	4	0.06673	Normal
50	0	101010	4.83341	0	4	1	0	0	1	25	41	0	0	0	3	0.06534	Normal
50	0	101010	31.91198	0	4	1	0	0	1	18	41	0	0	0	3	0.06717	Normal
50	0	101044	24.34167	0	4	1	0	0	1	5	38	0	0	0	4	0.06214	Normal
50	0	101010	26.75033	0	4	1	0	0	1	21	41	0	0	0	3	0.06662	Normal
50	0	101044	63.66485	0	4	1	0	0	1	17	38	0	0	0	4	0.06649	Normal
103	1	102041	0	1	3	0	93	1	0	0	0	1302	0	0	0	0.00726	Blackhole
153	1	103003	0	1	1	0	72	1	0	0	0	1296	0	0	0	0.00745	Blackhole
403	1	108078	0	1	3	0	73	1	0	0	0	1314	0	0	0	0.00725	Blackhole
553	1	111029	0	1	5	0	36	1	0	0	0	1260	0	0	0	0.00722	Blackhole
603	1	112040	0	1	5	0	48	1	0	0	0	1296	0	0	0	0.00726	Blackhole
703	1	114005	0	1	3	0	38	1	0	0	0	1254	0	0	0	0.00732	Blackhole
703	1	114065	0	1	3	0	40	1	0	0	0	1280	0	0	0	0.0073	Blackhole
53	1	101096	0	7	0	0	90	1	0	0	0	1350	15	121.69498	0	2.25865	Flooding
103	1	102001	0	6	14	0	51	1	0	0	0	150	0	0	0	0.00743	Flooding
103	1	102034	0	4	16	0	8	1	0	0	0	888	111	165.46205	0	2.35811	Flooding
103	1	102069	0	8	12	0	31	1	0	0	0	1240	40	93.93772	0	2.20716	Flooding
153	1	103009	0	6	16	0	20	1	0	0	0	1140	57	159.31297	0	4.19113	Flooding
153	1	103073	0	4	18	0	18	1	0	0	0	310	0	0	0	1.50971	Flooding
153	1	103088	0	6	16	0	11	1	0	0	0	990	90	88.47785	0	3.5653	Flooding
153	1	103096	0	4	18	0	40	1	0	0	0	1248	32	121.69498	0	3.86013	Flooding
203	1	104073	0	3	9	0	44	1	0	0	0	1276	29	136.03878	0	4.59168	Flooding
153	1	103003	0	1	4	0	22	1	0	0	0	1166	29	85.19787	0	2.06959	Grayhole
153	1	103043	0	1	4	0	47	1	0	0	0	1269	14	145.08942	0	1.88023	Grayhole
253	1	105005	0	1	9	0	47	1	0	0	0	1170	7	137.59248	0	0.92063	Grayhole
303	1	106079	0	1	3	0	75	1	0	0	0	1350	7	108.34705	0	1.64035	Grayhole
353	1	107033	0	1	3	0	71	1	0	0	0	1349	9	162.5505	0	2.03296	Grayhole
503	1	110024	0	1	9	0	35	1	0	0	0	1200	15	113.27654	0	2.0577	Grayhole
403	1	1088001	0	1	2	0	36	36	0	0	0	0	0	0	0	0.00723	Scheduling
403	1	108069	0	1	2	0	27	27	0	0	0	0	0	0	0	0.00722	Scheduling

Table 4.2: Comparison of Mathematical and Simulation Results

Round	No. of clusters	ADV_PM_SENT		ADV_PM_RCVD		REQ_PR_SENT		REQ_PR_RCVD		BS Receives	
		Math	Sim	Math	Sim	Math	Sim	Math	Sim	Math	Sim
1	4	4	4	293	293	92	92	92	92	234	234
2	2	2	2	150	150	94	94	94	94	45	45
3	4	4	4	250	250	88	88	88	88	165	165
2	5	5	5	237	237	67	67	67	67	176	176
5	6	6	6	334	334	45	45	45	45	298	298
6	7	7	7	398	398	99	99	99	99	345	345
7	2	2	2	132	132	45	45	45	45	98	98
8	3	3	3	238	238	67	67	67	67	652	652
9	4	4	4	345	345	34	34	34	34	334	334
10	6	6	6	694	694	65	65	65	65	198	198

Sim = Simulation

Table 4.3: NS-2.35 Simulation Parameter

SN	Parameter	Value
1	Channel type	Wireless
2	Number of nodes	200 nodes
3	Number of Clusters	10
4	Network area	200mX200m
5	Base Station location	(30,185)
6	Size of data packet	500 bytes
7	Size of packet header	20 Bytes
8	Max transmission range	200m
9	Routing Protocol	SPIN
10	MAC Protocol	CSMA/TDMA
11	Simulation Time	3600s
12	Initial Energy(in joule)	5, 50
13	Attacker intensities	10%, 30%, 50%

Table 4.4: NS-2.35 Simulation Parameters for Proposed Model

SN	Parameter	Value
1	Network Area	200m × 200m
2	Number of target source	5
3	Data rate at MAC layer	2 Mbps
4	Max packet in queue	150
5	Data Flow	50-100 Mbps
6	MA_ Fusion Factor(α)	1
7	MA_Reduction Ratio(γ)	10%
8	Local execution time at target node (τ)	50ms
9	Size of Processing Code	0.4k bytes

4.7.2 Proposed Workflow Description

This Section makes use of trained and un-trained datasets. The un-trained dataset contains the network performance parameter where the randomly deployed network nodes

```
begin
     agent arrives in region
      authenticate agent in MAM
   if authenticate(MAM) = true; /* check the authentication of the arrived agent
from specified source or not  */
  enter into region after the information from Sink node
if( ThisNode=ClusterNode)
MAM migrate the agent and assign as CA on cluster node;
      elseif(ThisNode= SourceNode)
MAM  migrate the agent via CA on source node and assign as SA.
MA collected sensed data; among the nodes from Sink node, select the target nodes
and assign to agents as Nextnode;
Set NextNode in the MA packet and MA migrate towards the target node
   If ( NextNode= ClusterNode) /*if next node from target node is cluster node enter
in CA;
   enter CH              /* MAM pass node to Cluster Agent */
   begin
  authenticate (node);
  if authenticate(node)=true; /* check the authentication of the node*/
  begin
  create RO;              /* Create ReadOnly file of node*/
 enter into Sensor node if( NextNode= SourceNode) /* as per target node*/
 elseif(ThisNode=LastNode) MA collected the sensed data; and migrate back to
MAM;
 goto  CA ;
elseif( NextNode= SourceNode)   /* CA pass node to Source Agent*/
enter SA                      /*node come on source node via SA  */
 begin
authenticate (node);
if authenticate(node)=true; /* check the authentication of the node*/
begin
create RO;              /* Create ReadOnly file of node*/
enter into source node if NextNode= SourceNode)
elseif(ThisNode=LastNode) MA collected the sensed data; and migrate back to SA;
goto CA;
else
goto MAM;
task() complete   /* agent task is completed*/
create AO;          /*Create AppendOnly file of  agent*/
authenticate (agent) from MA  /* Check the authentication of agent*/
 if true, then goto next host
    else
 agent attack is found;
  end;   else    goto Sink node;
        end;   else
goto CA; /* go to cluster Agent*/
goto SA   ;/*go to Source Agent*/
end;
```

Fig 4.4: Algorithm for Network Action on Agent Arrival

are. In comparison, the trained dataset considers performance parameters after applying the Waikato Environment for Knowledge Analysis (WEKA). It contains a lot of algorithms for data pre-processing, clustering, classification, association rules, regression, and visualization. Toolbox technique of node deployment for simulation experiments to evaluate the proposed dataset [24]. The workflow of the presented model is broken into phases. The first phase evaluates the untrained dataset, and the second phase evaluates the trained dataset from the machine learning learning algorithm to evaluate the best-suited model. A generic flow is presented in Figure 4.5.

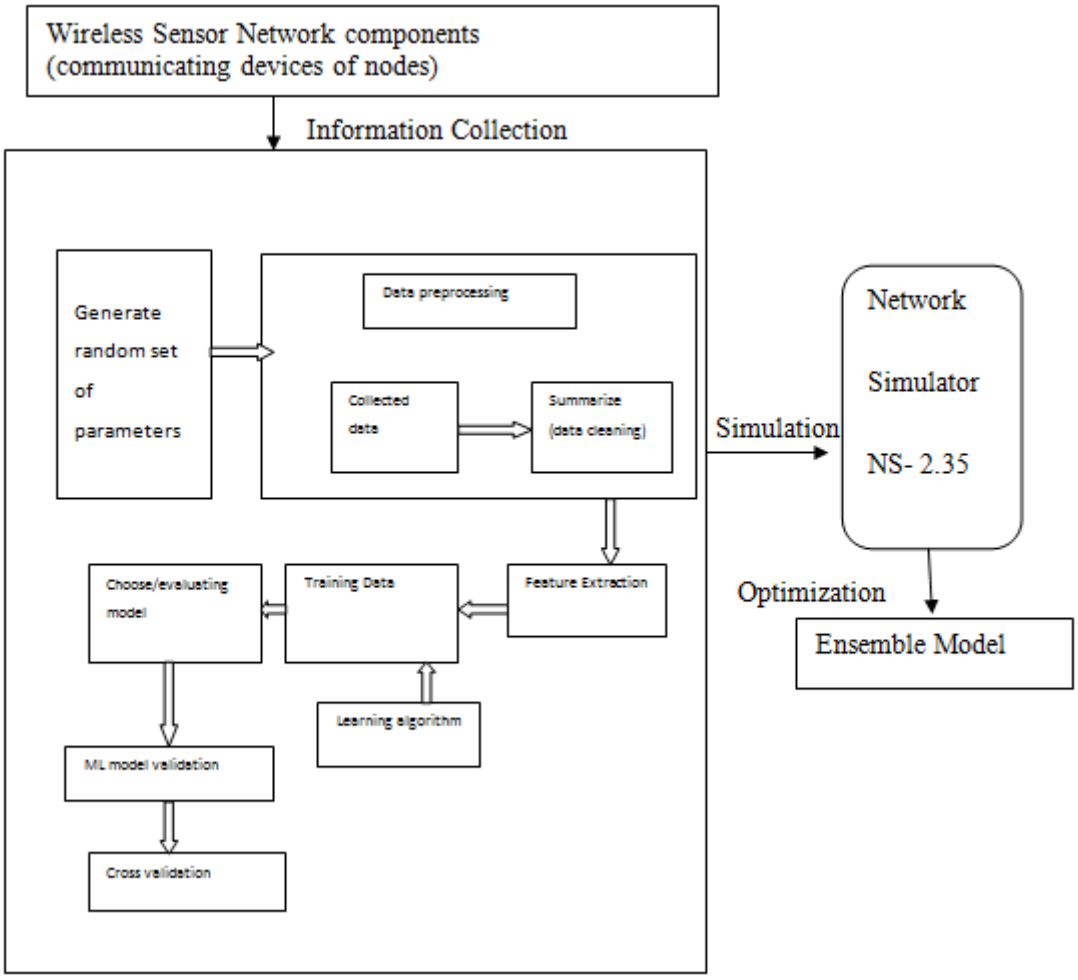


Fig 4.5: Proposed Structure for Ensemble Model

The performance parameter computed for this deployment is analyzed by attack type on a node on the basis of consumed energy criteria, which is an essential metric for dependability evaluation. The coordinates of attacks are normalized using the feature extraction technique to get the trained data. A subset of the WSN dataset is taken to extract the feature from all the attack cases that are classified correctly. The performance metric of features on the machine learning model is used and compared with the trained

data. After testing out various combinations of the model, an ensemble model has been proposed. It is validated that the proposed ensemble model performs better than the individual models. Various techniques like feature selection and cross-validation have been applied to achieve maximum accuracy in the prediction of an attack. The workflow of the proposed ensemble model can be broken down into four phases which is present in Figure 4.6.

- **Phase I**:
  The first phase includes identifying the five most common Denial of Services (DoS) attacks, namely Blackhole, Greyhole, flooding, scheduling, and normal (without attack) from the WSN Dataset. Features are then extracted from the attack files.

- **Phase II**:
  Various machine learning algorithms are trained on the training set. Hyperparameter tuning of these individual algorithms is done to achieve the best results on the test set. Table 4.1 gives a description of the various machine learning models along with their tuned hyperparameters.

- **Phase III**:
  After running rigorous iterations of various combinations of the machine learning models, a comparison of the proposed ensemble model was obtained based on evaluation parameters like Accuracy, AUC, Recall, Prec., F1, Kappa MCC as discussed in Section IV, which consisted of an ensemble of five machine learning models named (Random Forest, Gradient Boost, Linear SVM, Poly SVM, Logistic Regression).

- **Phase IV**:
  The performance of various models is generated, and the proposed ensemble model is better in comparison to other models (Refer Table 4.5. Further K-fold cross-validation is done to check the consistency of the ensemble model.

4.8 Model Evaluation

The performance of the ensemble model is evaluated using various parameters such as accuracy, AUC, precision, recall, F1, Kappa, and MCC. The results of all models on the same parameters are compiled in Table 4.5, which shows that the ensemble model performs better than other models. The k-fold cross-validation method is used to attempt to maximize the dataset for training and testing a model. It is beneficial for assessing model performance, as it provides a range of accuracy scores across (somewhat) different data sets. Further, K-fold cross-validation has been performed to check the consistency

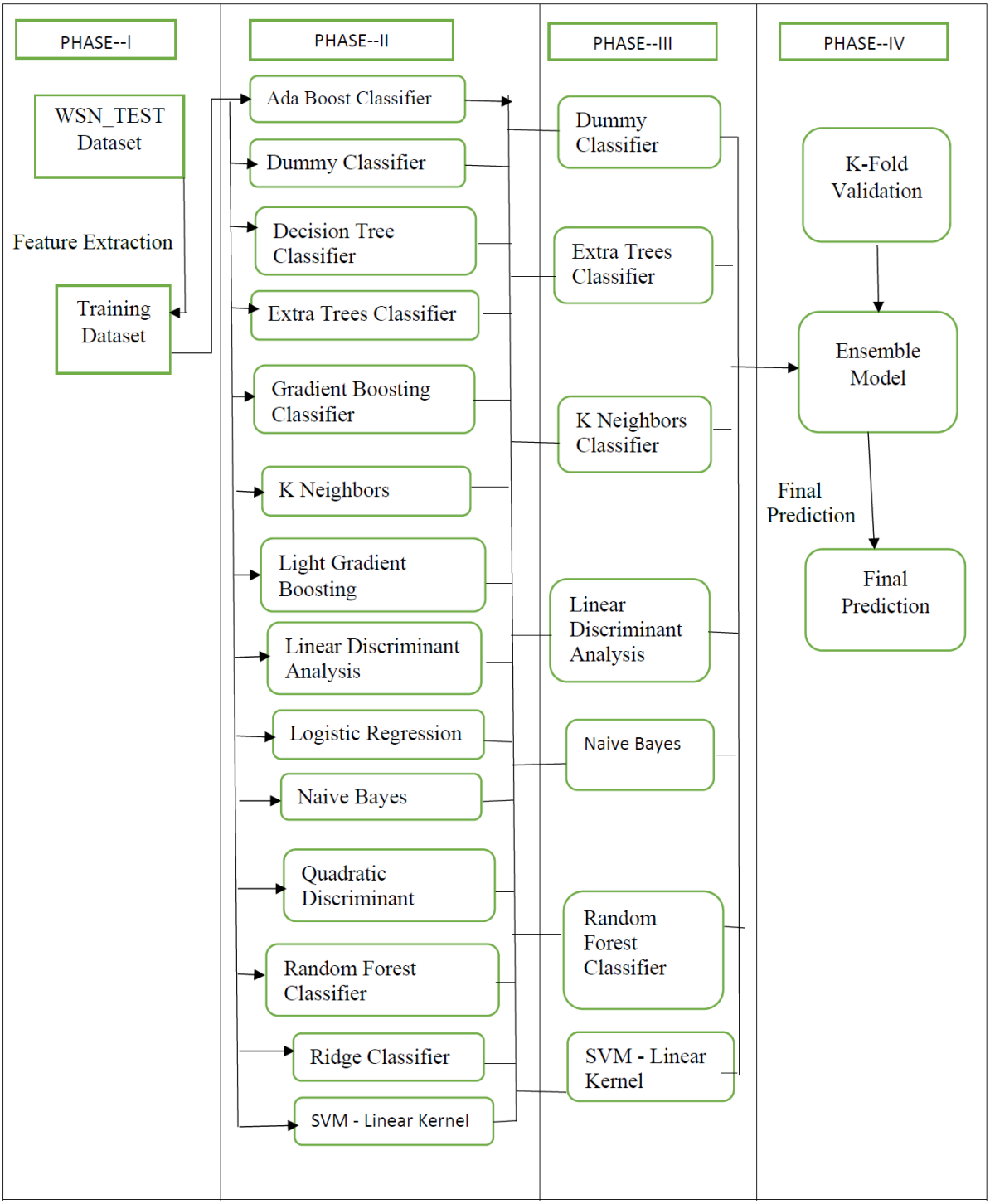


Fig 4.6: Workflow of Proposed Ensemble Model

of results and test the robustness of the model.

4.8.1 Model Evaluation Parameters

Model evaluation parameters are used to calculate the Accuracy, Precision, and Recall of various mentioned machine learning models to classify models.

1. **Precision** Precision is the fraction of relevant instances among the retrieved instances. It helps to calculate the fraction of relevant data (true positives) among all of the data predicted for a particular class. To compute the data, use equation 4.2.

$$Precision = TP/(TP + FP) \tag{4.2}$$

2. **Recall** Recall has defined the fraction of examples to predict the class, which is concerned with models that truly belong in the class. It is the fraction of relevant instances that have been retrieved over the total number of relevant instances. Recall is computed as equation 4.3.

$$Recall = TP/(TP + FN) \tag{4.3}$$

3. **F1-Score** F-1 Score is the harmonic mean of Precision and Recall. The contribution of both is to get a higher and better F1 score. Due to the product in the numerator, the final F1 score goes down significantly if one goes low. So a model does well in F1 score if the positive predicted are actually positives (Precision) and doesn't miss out on positives and predicts them negative (Recall).To compute F-1 Score equation 4.4.

$$F - 1Score = 2PrecisionRecall/(Precision + Recall) \tag{4.4}$$

4. **Accuracy** Accuracy calculates the %age of correct predictions for the test data. It can be calculated easily by dividing the number of accurate predictions by the number of total predictions. It measures the correctness of the classifier. Accuracy is divided by the correct prediction with all predication, computed as equation 4.5.

$$Accuracy = (TP + TN)/TotalData \tag{4.5}$$

4.8.2 K-Fold Cross Validation

K-Fold cross-validation estimates the accuracy of a classifier, which is important not only for predicting its future accuracy but also for choosing a classifier from a given set (model selection) or combining classifiers. To ensure that the proposed ensemble model is consistent with low bias and low variance, repeated K-fold cross-validation is performed. In this present work, K-fold cross-validation is repeated five times. The results obtained are plotted against accuracy as shown in Figure 4.7 where the lines are overlapping, which signifies the robust proposed ensemble model. The final average Accuracy obtained is 95.25%.

Table 4.5: Comparison of Machine Learning Models

SN	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC
1	Random Forest Classifier	0.9201	0.9201	0.9201	0.9201	0.9201	0.9201	0.9201
2	Light Gradient Boosting Machine	0.94	0.94	0.94	0.94	0.94	0.94	0.94
3	Decision Tree Classifier	0.9764	0.9751	0.9731	0.9926	0.9644	0.9653	0.9955
4	Gradient Boosting Classifier	0.9664	0.95	0.97	0.9973	0.9661	0.9553	0.9954
5	Extra Trees Classifier	0.9792	0.96	0.9618	0.9922	0.9796	0.9627	0.9862
6	Ridge Classifier	0.9675	0.9321	0.9607	0.9764	0.9682	0.9568	0.9583
7	Linear Discriminant Analysis	0.9639	0.9756	0.9582	0.974	0.9649	0.952	0.9537
8	Naive Bayes	0.9097	0.9756	0.9321	0.937	0.915	0.8817	0.8865
9	Logistic Regression	0.8448	0.9777	0.8006	0.8546	0.8386	0.7923	0.797
10	K Neighbors Classifier	0.6858	0.8951	0.5662	0.6745	0.6682	0.5736	0.5818
11	Ada Boost Classifier	0.6278	0.7896	0.4333	0.4714	0.517	0.4657	0.5643
12	Dummy Classifier	0.3214	0.5	0.2	0.1035	0.1565	0	0
13	SVM - Linear Kernel	0.2643	0	0.2044	0.0917	0.1254	0.0086	0.0146
14	Quadratic Discriminant Analysis	0.0687	0	0.2	0.0048	0.009	0	0
15	Ensemble Model	0.9894	0.9812	0.9791	0.9976	0.9896	0.9857	0.9966

4.9 Result Analysis, Comparison and Discussion

Table 4.5 lists the machine learning models trained on the dataset along with their hyperparameters. Feature extraction is done, and the dataset is split into two parts- the training dataset (80% of the total dataset) and the testing dataset (20% of the total dataset). The models are trained on the training dataset and are further tested on the testing dataset. The proposed ensemble model is a combination of five models. The models are evaluated on various parameters, as mentioned in Section V. The model should be cross-validated. If the resultant Accuracy after various runs is consistent in all the runs, then the trained models are not overfitted. Overfitting models are well-obtained data and learn too much. The Accuracy is validated by applying 10-fold cross-validation five times. Figure 4.7 shows the results after using K-fold cross-validation. The proposed ensemble model achieves higher accuracy.

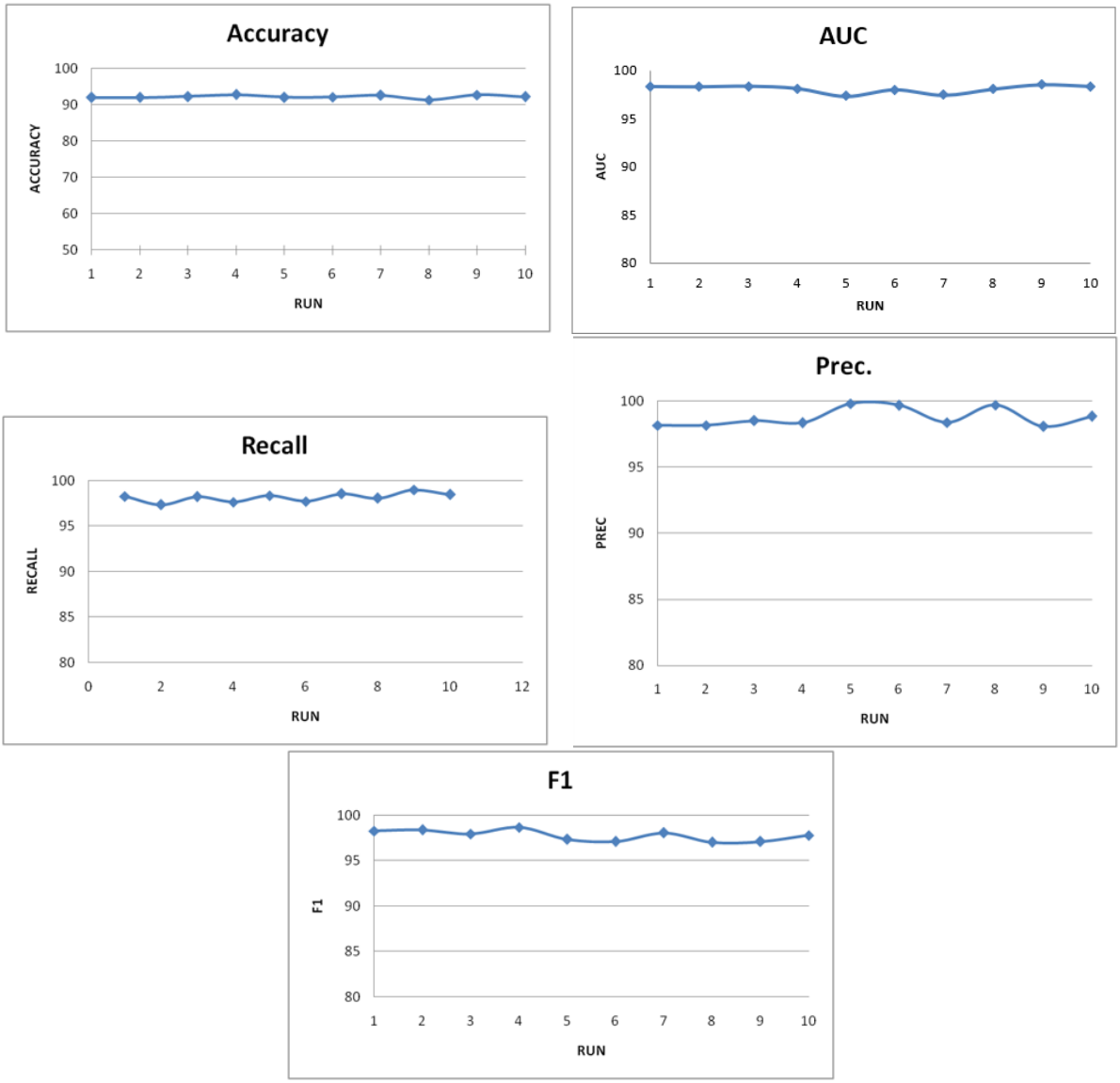


Fig 4.7: K-Fold Cross Validation

4.10 Summary

The aim of this ensemble model is to train a machine learning model to classify the attacks using a mobile agent-based intrusion detection system. To identify the attacks, an ensemble model normalized and un-normalized the data on the basis of consumed energy by the node. The four attacks, Blackhole, Grayhole, Flooding, and Scheduling attack, are identified for further data preprocessing classification and for the normal nodes when there is no attack. Further, classification methods are used to train the dataset to tune the hyperparameters.

Compare the machine learning models with an ensemble model to predict the attack on a

node. Identify the miscellaneous activities on the network scenario. The machine learning model performed differently according to the training dataset size. The recommended ensemble model achieved high classification accuracy when compared to other models to predict attacks.

To analyze the robustness of the ensemble model, the K-fold validation was used to determine the consistency of the ensemble model. The results conclude that the proposed ensemble model is capable of identifying the attack on the node. This research emphasizes the importance of security in the early research; we cannot waste time checking all the learning and classification models. Without such an ensemble model, we cannot identify the inherited vulnerabilities on these networks.

Chapter 5

Border-Hunting Optimization for Mobile Agent-Based Intrusion Detection With Deep Convolutional Neural Network

This Chapter delves into the advanced techniques of enhancing intrusion detection systems using mobile agents, specifically focusing on Border-Hunting Optimization combined with Deep Convolutional Neural Networks (DCNNs). It discuss about the explores innovative approaches to improving the accuracy and efficiency of mobile agent-based intrusion detection systems (IDS) within Wireless Sensor Networks (WSNs). further begins with a foundational introduction to the subject, setting the stage for the detailed exploration of various methodologies and models employed in the optimization and detection process. It provides a comprehensive overview of the current state-of-the-art techniques, including the integration of deep learning with mobile agents for more robust intrusion detection.

The sensor nodes, which are available in the Wireless Sensor Networks (WSN), are equipped with sensing abilities and communication. Several domains require the sensor nodes to be arranged in aggressive surroundings, in which observing malicious activities within the sensor network. Therefore, the present research proposes the border-hunting optimization-based deep CNN (BHO-DCNN) for mobile agent (MA)-based intrusion detection in WSN. The importance of the research relies on the BHO-DCNN model for identifying the intrusion available in the network, which is established by integrating the optimization with its features through a deep classifier for precise detection. The algorithm follows the communal hierarchy, surrounding, group hunting, and prey attacking, which provides an enhanced rate of convergence in the method of detection. The analysis is achieved through the IDS 2018 Intrusion CSV database, depending on the performance, such as delay, alive nodes, normalized energy, and throughput. The obtained number of alive nodes through the developed BHO-DCNN algorithm is 45, end-to-end delay is 0.2572 ms, normalized energy is 0.1622 J, as well as throughput is 0.3125 % for nodes 50 at the populate rate of 100 respectively.

5.1 Introduction

Nowadays, the Internet is considered an essential component of everyday life, and the fast development in technology related to computer networking makes it easy to be comfortable in organizations, social communities, as well as in business [43]. Concurrently, several kinds of threats to network safety are rising owing to the usual growth of several techniques related to vulnerability, as well as attacks [44]. Generally, the intrusion detection system (IDS) is intended to avoid intrusions as well as to prevent the programs, which categorize the intrusions as intrinsic as well as extrinsic in the computer networks of an association, as well as initiate the alarm when the safety intrusion us included in the network of association [45].

Several securities are available in wired networks, such as secured gateways and firewalls, to identify the attackers [46]. The previous web-based data processing systems are fed to several attacks, resulting in considerable losses of several forms of damage [47]. Generally, a mobile node is referred a malicious node when it undergoes the features like consumption in bandwidth, drained battery, packet delay, message tampering, breakage of links, capturing session, information stealing, drop packets, as well as false routing [7].

Consequently, the information on the malicious node identification is a vital feature present in the network [46]. Several schemes are established to identify and block the attacks that arise from the network, and the scheme of Mortgage Detection (MDS) is considered a significant scheme since it withstands external attacks very efficiently [47]. In addition, MDS provides a security fence to overcome the physical attack rises in the internet on the computer system [48].

In the research industry, applications with MA are considered a present topic, in which MA is the structure of both the information and software. Since MAs are available in such networks, there is no requirement for appropriate servers [46]. MA located itself in a particular region to gather the data and distribute the data to every node available in the network. MA is considered a code of the software, which transfers between the sensor nodes (SN) to gather information in the sensor network [49] that is situated closer to one another for creating extremely irrelevant information while sensing the atmosphere [47].

Malicious users attack information very easily, but it is tough to attack the machine of computation, which transfers data instead of transferring it to the machine through mobile agents. MA senses the surroundings and adapts to change them dynamically. MA is considered a particular kind of software entity that transfers between sensor nodes to

collect information. When the redundancy subsists in the information, then it eradicates mechanically, likewise, the volume of information, which is transferred to the sink is minimized through the MA [46].

Several systems of intrusion detection have been established by several researchers to sense intruders with numerous approaches. The faulty scheme, depending on three-phase negotiation, is achieved to detect the intrusion, where the consumption of energy is minimized since only one agent is transferred between the detected CHs node of malicious and the information from the trusted CHs [45]. Due to a few challenges, the previous methods are unable to perform the accuracy of sufficient detection [50]. The traffic in data transmission of the sensor node is superior to the network's ability, but the bandwidth of the link is limited in MA [51][52], as well as several MAs gather the data from every node of the sensor network. Thus, the overhead of the net traffic is acquired owing to the communication between agents as well as high energy consumption [53].

MAs are located in a particular region to gather the information and distribute it to every node present in the system [54]. When the MA is missed or transferred out of service like an unauthorized region, then the gathered information will be lost and lead to a delay in time as well as resource wastage [46].

Generally, the accuracy of the classifier does not decline significantly, and between every possible structure, the primary class distribution is about near to the scheduled allocation of the class, when the ideals are probable towards the chosen structures [45]. The system of ID scans and observes the network traffic to determine the activities of malicious. In addition, the occurrence of IoT devices is interlinked with a Wi-Fi network to form a compound as well as high-dimensional information, which leads to difficulty in detection. Besides, the MAs visit every sensor node as well as gather information from them [43], so the delay is high as well as the size of data upsurges while collecting information from node to node as well as depleting the energy due to migration.

Finally, the reliability gets minimized owing to the huge quantity of gathered data [53]. The foremost challenge is that the accuracy of the classifier does not decline extensively, and the primary allocation of the class between features will be near to the scheduled allocation of the class, which is close whenever the principles are similar to the selected features. Owing to the occurrence of a huge quantity of data, it is complex to learn, even if the classifier is proficient in most classification troubles, till the unnecessary structures are barred from the objective purpose [3]. Due to the communication between agents, the overhead in network traffic arises and increases energy consumption. Since the MAs visit every sensor node to gather the information, and then the delay is increased, and energy will be depleted due to migration. Thus, the reliability also gets minimized due

to the collection of a large quantity of information [53].

The present research focuses on developing the model of Border-Hunting Optimization (BHO) based DCNN to detect intrusion present in WSN. The BHO is intended to combine the features of encircling, hunting, and assembly, pestering, as well as observing both the Grey-Wolf optimization [55] and Border-Collie Optimization [56]. The major role of the present research includes:

- **Border-Hunting Optimization** The BHO is considered as the hybridized optimization of Lupus and collie, which is intended arithmetically with the features of lupus, the communal hierarchy, surrounding, and group hunting, but it lags in the prey attacking phase and falls in local optima hence the behaviors of collie have merged arithmetically like assembly, pestering and observing are merged with lupus. In the pestering phase, the velocity is also increased and the arithmetic compilation of these two optimization has an efficiency to attain the best optimal solution and high convergence rate that finds the optimal cluster head.
- **Border-Hunting Optimization-based Deep CNN (BHO-DCNN)** The intrusion in WSN is detected through the BHO-DCNN classifier, which highlights the improved performance through the Mobile Agent selection as well as the clustering with the developed optimization.

5.2 Literature Survey

Malicious mode detection in mobile with the MA in a cluster depending wireless sensor networks (WSNs) is discussed in [53], in which the defective scheme depending on three-phase negotiation is achieved, and the agents of multi-mobile are employed to accumulate the information from every sensor nodes after authentication. However, it is ineffective to confirm every node present in the network due to mobility and high consumption of energy as well as delay, so this can be resolved through assembling sensor nodes as clusters, and a single MA achieves confirmation only with every CH instead of confirming every SN. Thus, energy consumption is low, whereas time consumption is high.

An IDS for Internet of Medical Things (IoMT) is described in [54], in which the novel MA-based IDS is intended to protect the system of linked medicinal devices that is independent and utilizes machine learning as well as the regression algorithms to identify the system level intrusions and irregularity available in the sensor information. Here, the calculation time is low, whereas the expected accuracy is not obtained due to the pixel matrix. The particular energy consumption of the detection system is to be minimized. It

is also important to observe other algorithms, like state graphs, to examine the change in sensor data. An effective, secure detection as well as the prevention of malevolent nodes using the system of surprise check with the trusted MAs in Mobile Ad-hoc networks (MAN) is developed in [46], which utilizes three stages such as lightweight surprise check manager, cluster pattern, as well as MA. Initially, the manager identifies the malevolent nodes depending on the rate of the forwarded node and acknowledgment node, as well as the node's location and safe broadcasting.

Subsequently, the clusters are created depending on the strength of the expected signal, and the elected CH is based on the residual energy. Finally, in the MA stage, the scheme of the secret key is employed to validate the authentication of the MA. The network overload and energy utilization are minimized, and the efficiency of the network is improved. However, this system does not validate the several kinds of attacks present in the MAN with the highly developed cryptography algorithms for safety improvement.

The approach based on energy efficiency, as well as the secured MA, is discussed in [44], in which the framework based on fingerprint is developed to calculate the node authentication. The CH selection is done depending on the cluster formation as well as the density impact factor to get a better duration of the system. Intelligent prevention of the intrusion of gathering features as well as classification algorithms is explained in [48], which are based on several algorithms like neuron-genetic, intelligent software agents, etc, to detect and prevent the intrusion of the network to deliver the internet safety as well as enhance the value of service. This system minimizes the delay to enhance the effectiveness of routing and obtain the expected accuracy.

The method of Deep Learning (DL) with CNN for detecting the intrusion is effectively depicted in [50], in which a novel algorithm of feature selection termed conditional random field as well as coefficient of linear correlation is to choose the most contributed structures as well as to categorize with the prior convolution neural network, which initiates the conditional random field for choosing two variables. The use of CNN is helpful in enhancing the accuracy of detection performance. However, the speed of data communication is to be enhanced. The data is fused with the intelligent data-gathering schema in WSN [57].

The data fusion is executed with the help of NN, which enhances the network's working efficiency. Depending on the node score, the data fusion is done, which assists in selecting the cluster head. Since this method can define the undefined data this leads to fitting issues. To overcome the hotspot problem, a power collection in sensor information system was developed in [58]. During the transmission process, the transmission expense was reduced by determining the communication distance. The dying nodes are safeguarded

by concerning the threshold value. The energy was consumed using the mobile sink technique where the threshold used caused computation expenses. A traffic partitioning method was developed that allocates the packet traffic and allows the lightest traffic flow [59].

The connections were split to neglect the false TCP connections with the assistance of the initial and established connection table. The DTP used can create vulnerabilities in the network. The features of spatial-temporal were extracted using the CNN; the features of spatial-temporal were controlled using the gated scheme in [60]. The multiple gated spatial CNN predicts the traffic flow to a huge area. The weight fusion method is not capable of fusing a large number of extracted features. The data security and protection of blockchain technology are examined in [6] The decentralized secure system secures the data more than any other method that is trustable. The adversarial attack detection system was developed in [61] to detect the intrusion present in the system. The normal data were extracted with the assistance of a support vector machine. The IDS extracted the adversarial attack with the help of the trained models. The method does not work well in high-noise situations.

In the prior research, there were numerous demerits like high time consumption, expected accuracy not obtained, the detection rate decreasing owing to the rise of malevolent nodes, and high calculation time. The false rate of prediction also improved. So, these above disadvantages are overcome in this research through the BHO-based DCNN to detect the intrusion in WSN through optimum mobile-agent.

5.3 Mobile Agent-based intrusion detection using Border-hunting Optimization with Deep Convolution Neural Network

The major intention of the research is to design mobile agent (MA)-based intrusion detection with a Deep Convolution Neural Network (DCNN) for detecting interruption that occurs in the network. A wireless network comprises several sensor nodes to utilize the information from several resources. Generally, the sensor nodes are considered to be moving objects that are present in a random way, and the information available in the sensor node is passed to the cluster head, which is located in certain regions that gathers the information from sensor nodes and fed forward to the MA to aggregate the information. Then, the collected data is fed to the sink node by MA to reduce the propagation delay and energy consumption.

Subsequently, the cluster head is chosen depending on the multi-objective function that comprises energy, delay, distance, traffic, and throughput with the optimized cluster head selection approach through the BHO. Then, the transmitted data is aggregated and employed for classification to classify as normal or malicious. This optimization is developed based on the standard features acquired from the Border collie optimization (BCO) [56] and grey wolf optimization (GWO) [62] to select the cluster head and detect the intrusion.

The BHO is regarded as a hybridized optimization of lupus and collie, designed arithmetically with the characteristics of lupus-like social hierarchy, environment, and group hunting, but it slows in the prey's attacking phase and drops in local optima, so the behaviors of the collie are incorporated arithmetically with lupus-like assembly, pestering, and observing. The velocity is also raised during the pestering phase, and the mathematical combination of these two optimizations has a high convergence rate and efficiency in achieving the most ideal solution.

Finally, the proposed BHO-based DCNN will declare the node as normal or malicious. If it is a malicious node, then further communication with the node will be blocked, the implementation of the research will be done in MATLAB, and the efficiency of the developed system will be calculated through comparison with prior techniques. The performance metrics compared for the comparison will be energy consumption, delay, traffic overhead, and throughput. The graphical illustration of the research is revealed in Figure 5.1.

5.3.1 WSN-System Model

WSN includes a number of sensor nodes with a particular BS that is referred to as N_B, in which the wireless relations are termed as the straight statement amongst sensor nodes at a certain range. Every sensor node is discrete equally, with a particular size of S_d and T_d, through its overwhelming greatest range. Each sensor node comprises a particular ID as well as it is gathered as clusters. Thus, the sink node is positioned in place of $(0.5S_d, 0.5T_d)$ that is set close to the optimal location to acquire every symbol of data from nodes that are interconnected to the present system. The sensor node position is calculated through its direct values like S_i and T_i. The information is moved to BS from every sensor node with the Selection of CH, where S_c is referred to as the number of sensor nodes initiated at CH,H_M. , which is present in a cluster and denoted as H_M. When the clusters are created in the sensor system, the information sent from every node is directly forwarded to the equivalent H_M, which collects the information. Then, this collected information is subjected to the sink through the mobile agent (MA). Thus, the

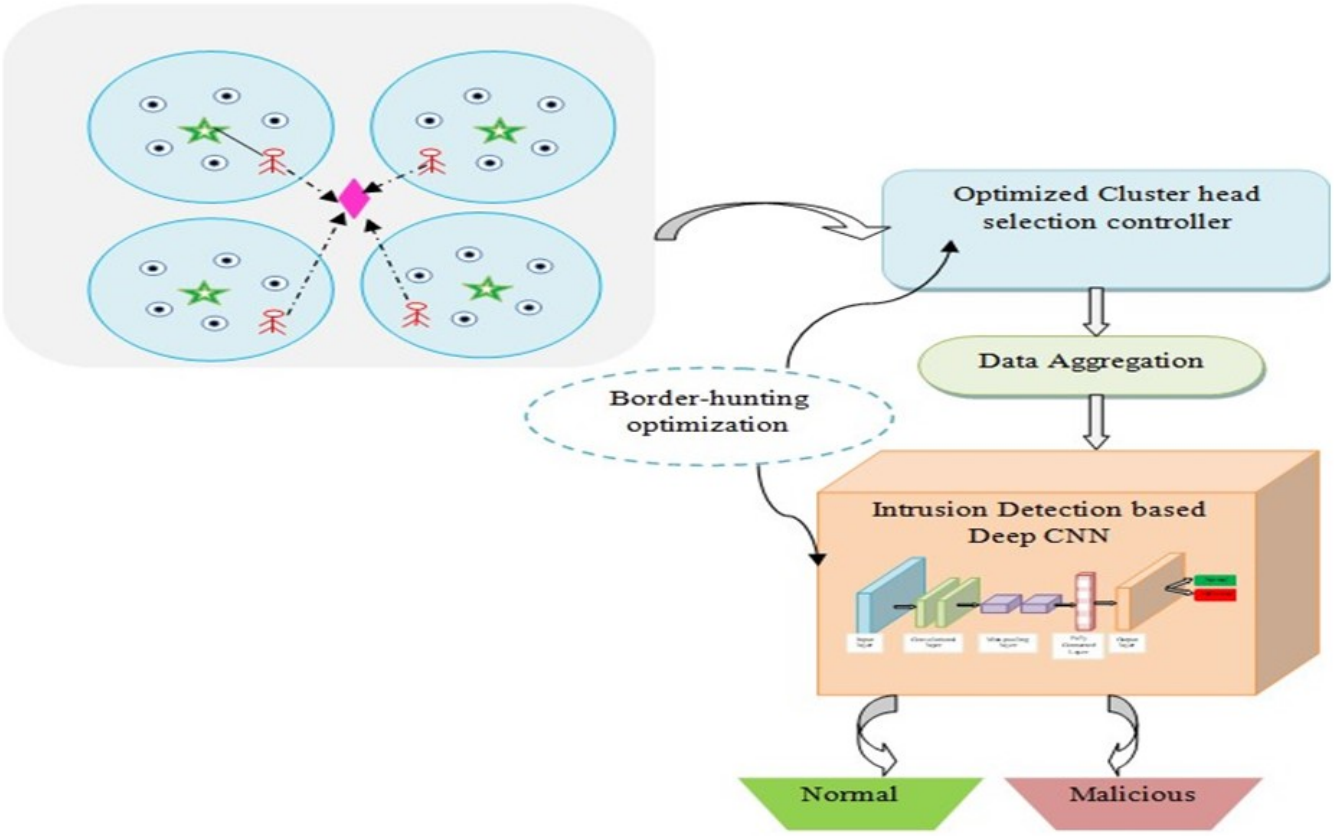


Fig 5.1: Graphical Depiction of the Developed Model

simulated system of WSN is depicted in Figure 5.2.

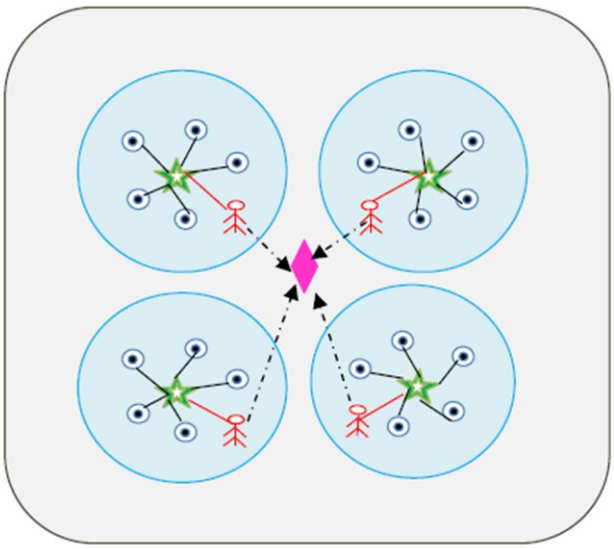


Fig 5.2: Architecture of WSN

1. **Energy model**

Initially, every sensor node acquires the first energy as E_0, referred to as a non-chargeable. Every information packet travels to CH from the normal node in random space and the scheme of multipath fading, which is based on distance from sources to destination. Every scheme of information transfer utilizes the TDMA protocol. The transmitter, which is available in the node, consists of both radio electronics as well as a power amplifier, in which the receiver contains the radio electronics to achieve the method of dissipating energy. Furthermore, this dissipating energy of every piece of information is in the dimension of I_d, which contains two several approaches for dissipation. When the normal node transmits data bytes I_d, then it is represented as

$$E_d\left(P_d^k\right) = \begin{cases} E_{ee} * I_d + E_{rs} * I_d * \left\|P_d^k - P_b'\right\|; if \left\|P_d^k - P_b'\right\| \geq g_0 \\ E_{ee} * I_d + E_{tu} * I_d * \left\|P_d^k - P_b'\right\|; if \left\|P_d^k - P_b'\right\| < g_0 \end{cases} \tag{1}$$

$$\epsilon_{g_0} = \sqrt{\frac{E_{tu}}{E_{rs}}} \tag{2}$$

where E_{ee} denotes the electronic energy, which is expressed as

$$E_{ee} = E_t + E_a, \tag{3}$$

where E_t and E_a denotes the energy of the transmitter as well as aggregated information. The parameter of energy such as E_{rs} denotes the power amplifier in the transmitter, whereas $\left\|P_d^k - P_b'\right\|$ signifies the distance of both the normal nodes as well as CH. When the CH node receives I_d then, the energy dissipation is assigned as

$$E_d\left(P_A^k\right) = E_{ee} * I_d \tag{4}$$

The value of every energy node is efficient once sending or getting data bytes I_d, which is expressed as:

$$E_{l+1}\left(P_d^k\right) - E_l\left(P_d^k\right) - E_d\left(P_d^k\right),$$
$$E_{l+1}\left(P_A^k\right) - E_l\left(P_A^k\right) - E_d\left(P_A^k\right). \quad (5)$$

When the energy node is below zero, the information process is repeated until every node reaches the dead node. When the energy node is below zero, the information process is repeated until every node reaches the dead node.

2. **Data Aggregation**

   The data from numerous sources are gathered together in the summarized form where the sensor node, which is a randomly moving object, invokes the information, and then the information is collected from the sensor node to CH. The gathered information is forwarded to the MA for aggregation using the aggregation function.

5.4 Border-Hunting Optimization for Cluster Head Selection in WSN

The node is selected randomly from the system, and Cluster Head (CH) is selected for transferring the information to the required node through the BHO, in which the characteristics of both the BCO [56] and GWO [62] are hybridized for choosing the suitable CH. The qualities of optimizations regarding hunting are hybridized in BHO, which enhances the optimal convergence of integrated hunt optimization. BHO is termed a method of swarm intelligence (SI), which takes over numerous qualities.

The developed optimization comprises the parameters of minimal adjustment as well as utilizes the parameters of minimal optimization. BHO ensures efficient stability by improving the convergence rate to find the best solution. Normally, the attacking phase of Lupus is not effective in attacking prey, so the features of assembly, pestering, and observing from the Collie are merged with the Lupus to improve the hunting behaviors for selecting the global best solution.

Usually, the lupus is present under the family related to Candidate, which are referred to as the Apex predators that are present at the peak of the food chain. The first group is termed as a group, which is responsible for making the decisions regarding hunting, wake-up time, sleeping location, etc. The leaders in this group are male as well as female. The second level in the food chain hierarchy is the group, which is referred to as the subordinate group that assists the initial group during decision-making. The final low-

level group in the food chain is considered a group that plays the role of Bovidae, which always possesses the ability to submit to every dominant wolf. The major phases included in this GWO are as communal hierarchy, surrounding, and group hunting.

- **Communal hierarchy** In the communal hierarchy, the fittest solution of the group is considered while designing, and the second as well as third resolutions are referred to as a group correspondingly. The residual candidate solutions are referred to as groups. Here, the hunting procedure is guided through the groups of, whereas group follows the first three groups.

- **Surrounding** During prey hunting, lupus surrounds the prey and models the surrounding behavior.

$$\vec{E} = \left| \vec{G} \cdot \vec{V}_p \left(t_p \right) - \vec{V} \left(t_p \right) \right| \tag{6}$$

$$\vec{V} \left(t_p + 1 \right) = \vec{V}_p \left(t_p \right) - \vec{B} \cdot \vec{E} \tag{7}$$

where t_p denotes the present iteration, $\vec{B}$ and $\vec{C}$ signifies the coefficient vectors, and the position vector of lupus is denoted as $\vec{V}_p$. The vector $\vec{B}$ and $\vec{C}$ is evaluated as

$$\vec{B} = 2\vec{b} \cdot \vec{r}_1 - \vec{b} \tag{8}$$

$$\vec{G} = 2\vec{r}_2 \tag{9}$$

where $\vec{b}$ element is decreased linearly over the iterations as well as the random numbers such as r_1 and r_2.

- **Group hunting** Lupus can detect the location of prey and enclose them, which is directed through the K group. Also, the group of both H and P participate in hunting irregularly. The first three groups, such as K, H, and P, are to suggest the hunting characteristics of lupus, as well as possess better information concerning the prey situation. The below equations are developed to update the point to the position of the best agents.

$$\vec{E}_{\mathrm{K}} = \left| \vec{G}_1 \cdot \vec{V}_{\mathrm{K}} - \vec{V} \right|, \vec{E}_{\mathrm{H}} = \left| \vec{G}_2 \cdot \vec{V}_{\mathrm{H}} - \vec{V} \right|, \vec{E}_{\mathrm{P}} = \left| \vec{G}_3 \cdot \vec{V}_{\mathrm{P}} - \vec{V} \right| \tag{10}$$

- **Prey Attacking**

In this phase, the characteristics of Collie, such as assembly, pestering, and observ-

ing, are merged with lupus to enhance the hunting features for choosing the global optimal solution.

– **Assembly** Lupus controls the sheep (prey) under the family of Bovidae from the front as well as sides, which leads to the collection and straightens them towards the ranch, which is called assembly. The Bovidae move towards the direction of lead lupus, and the Bovidae are selected based on their fitness values.

$$W_g = (F_f - F_b) - \left[\left(\frac{(F_L + F_R)}{2}\right) - F_s\right], \tag{11}$$

where F_f and F_b represent the best fitness of the individual and fitness value of the Bovidae. If W_g is positive, then it denotes the Bovidae is closer to lupus; thus, the Bovidae is gathered and elected based on their fitness values as,

$$\vartheta_b(\tau + 1) = \sqrt{\vartheta_f(\tau + 1)^2 + 2 \times Ac_f(\tau) \times P_b(\tau)} \tag{12}$$

where the velocity of Bovidae is represented as ϑ_b, which is openly prejudiced through the velocity of lupus at time $(\tau+1)$, as well as the acceleration of lupus at time (τ), and $P_b(\tau)$ is the current position of the Bovidae to be collected.

– **Pestering** In this stage of pestering, lupus controls the Bovidae by bending down their heads, placing their rear leg high, and setting their tails down. The Bovidae, which are present near both left and right lupus, need to be bothered from the sides to maintain one path. The updated velocity of the pestered Bovidae is,

$$v_R = \sqrt{\left(\vartheta_R(\tau + 1)\tan(\theta_1)\right)^2 + 2 \times Ac_R(\tau) \times P_R(\tau)}, \tag{13}$$

$$v_L = \sqrt{\left(\vartheta_L(\tau + 1)\tan(\theta_2)\right)^2 + 2 \times Ac_L(\tau) \times P_L(\tau)}, \tag{14}$$

$$\vartheta_{pb}(\tau + 1) = \frac{v_L + v_R}{2}, \tag{15}$$

where ϑ_{pb} is denoted as the pestered Bovidae, which is based on the velocities of both the left as well as the right lupus. The values of both the tangent θ_1 and θ_2 are selected randomly.

– **Observing** Lupus imitates the fatality selection characteristics of lupus; these

cunning lupus stare at the Bovidae with their eye when Bovidae move off track. Lupus with low fitness is assigned to move behind the Bovidae and observe it, which is expressed in the below expression,

$$\vartheta_b(\tau+1) = \sqrt{\vartheta_L(\tau+1)^2 - 2 \times Ac_L(\tau) \times P_L(\tau)} \tag{16}$$

$$\vartheta_b(\tau+1) = \sqrt{\vartheta_R(\tau+1)^2 - 2 \times Ac_R(\tau) \times P_b(\tau)} \tag{17}$$

where the $\vartheta_L(\tau+1)$ and $Ac_L(\tau)$ signifies the velocity as well as the acceleration of left lupus, when it holds the worst value of fitness, whereas $\vartheta_R(\tau+1)$ and $Ac_R(\tau)$ represents the velocity as well as the acceleration of right lupus when it possesses the least value of fitness. P_b reveals the current location of the Bovidae to be collected.

where the $\vartheta_L(\tau+1)$ and $Ac_L(\tau)$ signifies the velocity as well as the acceleration of left lupus, when it holds the worst value of fitness, whereas $\vartheta_R(\tau+1)$ and $Ac_R(\tau)$ represents the velocity as well as the acceleration of right lupus when it possesses the least value of fitness. P_b reveals the current location of the Bovidae to be collected.

- **Termination**

Finally, the position of the lupus is updated as

$$P_f(\tau+1) = \vartheta_f(\tau+1) \times T_f(\tau+1) + \frac{1}{2} Ac_f(\tau+1) \times T_f(\tau+1)^2 \tag{18}$$

The above equation updates the position of lead lupus, whereas the location of both the left and right lupus is updated by the below equations.

$$P_L(\tau+1) = \vartheta_L(\tau+1) \times T_L(\tau+1) + \frac{1}{2} Ac_L(\tau+1) \times T_L(\tau+1)^2 \tag{19}$$

$$P_R(\tau+1) = \vartheta_R(\tau+1) \times T_R(\tau+1) + \frac{1}{2} Ac_R(\tau+1) \times T_R(\tau+1)^2 \tag{20}$$

The location of the Bovidae is efficient with the below equations when it is fit to the groups of assembly as well as the pestering.

$$P_{ab}(\tau+1) = \vartheta_{ab}(\tau+1) \times T_{ab}(\tau+1) + \frac{1}{2} Ac_{ab}(\tau+1) \times T_{ab}(\tau+1)^2 \quad (21)$$

$$P_{pb}(\tau+1) = \vartheta_{pb}(\tau+1) \times T_{pb}(\tau+1) - \frac{1}{2} Ac_{pb}(\tau+1) \times T_{pb}(\tau+1)^2 \quad (22)$$

The expression for the observed Bovidae is expressed as,

$$P_{ob}(\tau+1) = \vartheta_{ob}(\tau+1) \times T_{ob}(\tau+1) - \frac{1}{2} Ac_{ob}(\tau+1) \times T_{ob}(\tau+1)^2 \quad (23)$$

The pseudo code of BHO-DCNN is represented in **Algorithm 1.**

Algorithm 1. BHO-DCNN

SI. No Pseudo code of BHO-DCNN

1. Input: Initial population
2. Output: Optimal solution
3. Initialization
4. (i) Communal hierarchy:
5. (ii) Surrounding: $\vec{E} = \left| \vec{G}.\vec{V}_p\left(t_p\right) - \vec{V}\left(t_p\right) \right|$
6. (iii) Group hunting: $\vec{E}_{\mathrm{K}} = \left| \vec{G}_1 \cdot \vec{V}_{\mathrm{K}} - \vec{V} \right|, \vec{E}_H = \left| \vec{G}_2 \cdot \vec{V}_H - \vec{V} \right|, \vec{E}_{\mathrm{P}} = \left| \vec{G}_3 \cdot \vec{V}_{\mathrm{P}} - \vec{V} \right|$
7. (iv) Prey attacking:
8. Assembly: $W_g = (F_f - F_b) - \left[\left(\frac{(F_L + F_R)}{2} \right) - F_s \right]$
9.

Pestering: $v_R = \sqrt{\left(\vartheta_R(\tau+1)\tan\left(\theta_1\right)\right)^2 + 2 \times Ac_R(\tau) \times P_R(\tau)}$

$v_L = \sqrt{\left(\vartheta_L(\tau+1)\tan\left(\theta_2\right)\right)^2 + 2 \times Ac_L(\tau) \times P_L(\tau)}$

$\vartheta_{pb}(\tau+1) = \frac{v_L + v_R}{2}$

Observing: $\vartheta_b(\tau+1) = \sqrt{\vartheta_L(\tau+1)^2 - 2 \times Ac_L(\tau) \times P_L(\tau)}$

$\vartheta_b(\tau+1) = \sqrt{\vartheta_R(\tau+1)^2 - 2 \times Ac_R(\tau) \times P_b(\tau)}$

5.5 Mobile Agent-based Intrusion Detection with Deep CNN

19X1 Layer array with layers:

1	Input image	$6 \times 1 \times 1$ images with the normalization of "zero center"
2	Convolution	41×1 convolutions with stride [11]
3	ReLU	ReLU
4	Max pooling	1×1 max pooling with stride [5 5]
5	Convolution	81×1 convolutions with stride [11]
6	ReLU	ReLU
7	Max pooling	1×1 max pooling with stride [5 5]
8	Convolution	161×1 convolutions with stride [1 1]
9	ReLU	ReLU
10	Max pooling	1×1 max pooling with stride [5 5]
11	Convolution	321×1 convolutions with stride [1 1]
12	ReLU	ReLU
13	Max pooling	1×1 max pooling with stride [5 5]
14	Convolution	641×1 convolutions with stride [1 1]
15	ReLU	ReLU
16	Max pooling	1×1 max pooling with stride [5 5]
17	Fully connected	2FCL
18	Softmax	Softmax
19	Classification output	Cross entropy

In general, the function of DL with a huge quantity of information is exclusive when functions with multifaceted information, and it involves a wide calculation overhead to achieve mathematical computations. Thus, the above challenges are conquered through DCNN, and the structure of the BHO-based DCNN form is established by integrating the intended optimization with the deep CNN model established here.

In general, CNNs are qualified to set up parameters that reduce errors. The structural design of DCNN for intrusion detection is exemplified in Figure 5.3, which comprises layers such as convolution, max-pooling, as well as fully connected (FCL). The Max pooling layers include the kernel size of 1×1, which is available among the convolution and ReLU layers. These feature maps are connected to a tiny part of the input, which is termed the receptive field, and a new feature map is formed during the filter descending by the evaluating dot product. Every component distributes equivalent weights to reduce

the error function.

1. **Convolutional Layer** A convolution layer multiplies a group of weights through input from NN. Here, the convolution layer includes the set of learnable kernels, which forces to get rid of local texture from input, as well as each kernel is employed to estimate the feature map with sizes of Windows 4, 8, 16, 32, and 64, as well as a kernel size of 1×1. During the multiplication process, the input is iterated over several times by a kernel.

2. **Max Pooling Layer**

   The node size is decreased gradually through the pooling layers, which preserve only the most crucial details. An idealized depiction of the convolved features is provided, and thus, the max pooling layer assists in minimizing its spatial size as well as over-fitting, which employs the splitting technique based on samples. It resembles the previous convolution layer by considering the maximum input region such that the kernel overlaps rather than the dot product of the input and kernel.

3. **ReLU Layer**

   The model gains non-linearity through activation functions, which enables it to learn intricate function mappings among the input and response variables. Although there are several alternative activation functions, RelU is the most popular one for many different types of neural networks because of its linear behavior, which makes it simpler to train and frequently results in higher performance.

4. **Fully Connected Layer**

   In FCL, each node in the subsequent layer is linked with each node in the input layer. One or more completely linked layers are utilized at the CNN. An FCL is added to learn non-linear combinations with the high-level characteristics produced by the convolutional layers.

5. **Softmax Layer**

   Using a softmax layer, the input is given a softmax function. The neural network layer is utilized to execute Softmax, and thus, the nodes in the Softmax layer are equal to that of the output layer.

6. **Output Layer** In the DCNN, desirable predictions are made in the last layer, known as output. A NN possesses a single output layer, which forms the preferred consequences. Before determining the final output, a distinct collection of weights and biases is applied.

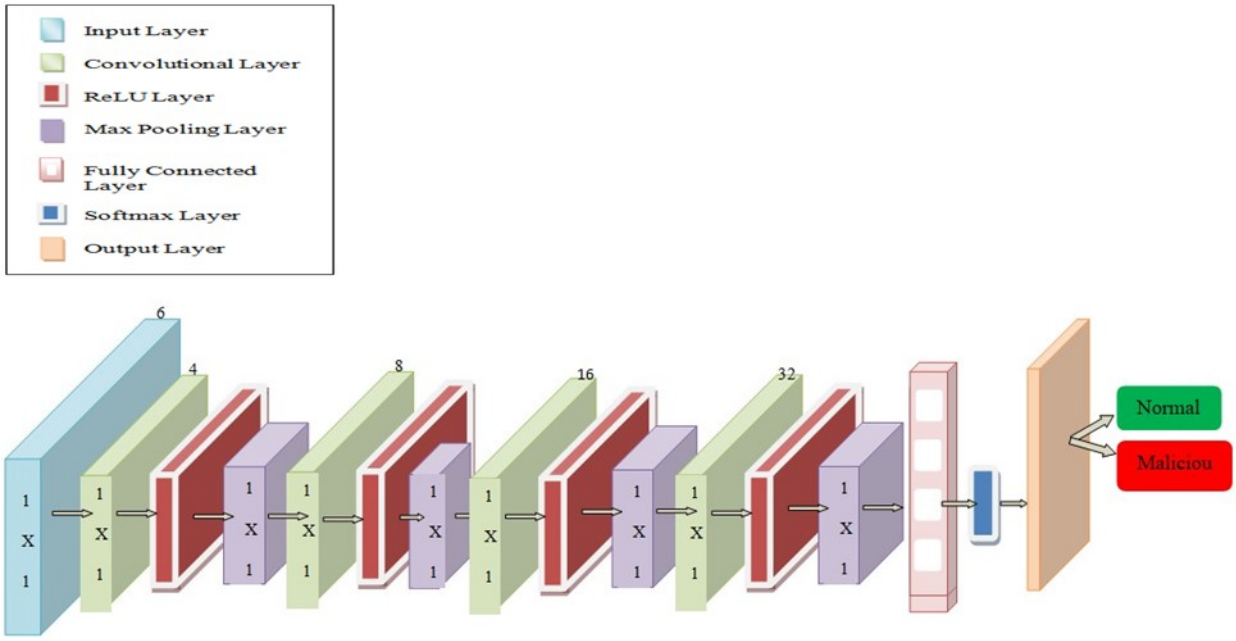


Fig 5.3: DCNN Architecture for Intrusion Detection

5.6 Results Analysis and Discussion

This segment clarifies the consequences and discusses BHO-based DCNN for intrusion detection in a precise manner. The effectiveness of the BHO optimization is validated by depending the performance metrics like alive nodes, delay, normalized energy, as well as throughput,. In addition, a relative analysis is attained with standard techniques.

5.6.1 Experimental Setup

The MATLAB tool is utilized for the execution of intrusion detection in WSN with BHO-based deep CNN in Windows 10 OS through 8GB RAM.

5.6.2 IDS 2018 Intrusion CSV's

The input is taken from IDS 2018 Intrusion CSV's dataset, which is a standard and widely used dataset that has a high adaptation capacity that helps efficiently in examining the models. The dataset comes with the labeled data that specifies whether the network is normal or not, which comes with the preprocessed features.

5.6.3 Simulation Results

The simulation of the network 0,500,1000 and 1500 is specified in detail through the BHO algorithm developed. An intrusion in the network is detected precisely through the developed BHO. Primarily, the nodes are dispersed randomly. The energy is lost between the nodes during detection, but the alive nodes are active while comparing with the prior approaches by expanding throughput. The attained consequences from BHO for both 50 as well as 100 nodes at several Rounds (R) of 0, 500, 1000, and 1500 are shown in Figures 5.4 and 5.5

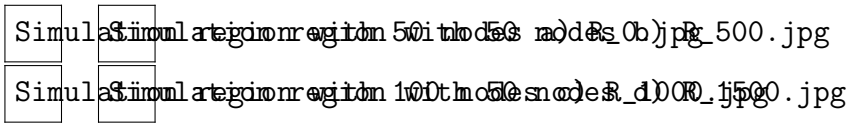


Fig 5.4: Simulation region with 50 nodes with Rounds 0, 500, 1000 and 1500

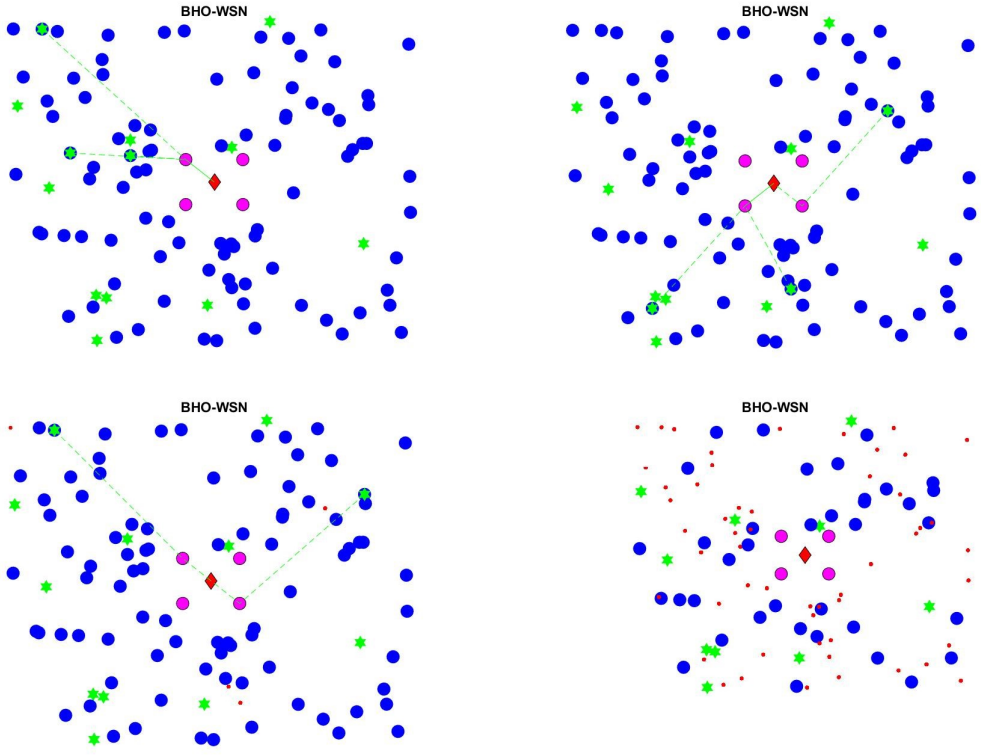


Fig 5.5: Simulation region with 100 nodes with Rounds 0, 500, 1000 and 1500

5.6.4 Performance Analysis

The outcomes of BHO-DCNN in terms of using their performance matrices like alive nodes, delay, throughput, and normalized energy are elaborated here with the dataset of IDS 2018 Intrusion CSVs for several rounds.

5.6.4.1 Performance Metrics

The metrics employed in the comparative techniques through the developed method are depicted below.

1. **Alive Nodes** The number of active nodes at the final phase of the sending procedure creates the alive node, and the optimal approach attains the maximal number of alive nodes.

2. **Delay** Delay is referred to as the average time assigned among the transferred and attained information that is employed to compute the average delay during the data transmission in the system.

3. **Normalized energy** The quantity of energy entering each node is measured as an essential thing since it depicts the network's life span as well as the formulation of energy.

4. **Throughput (T)** It calculates the number of data received at each second at the precise intention, which is referred to as the summation of every rate of information, which is offered for every secondary user in the system that measures the delivery rate of successful data over the entire network that is expressed as 5.1

$$T = \frac{Sizeofdata}{Timeoftransmission} \tag{5.1}$$

1. **Setup fo Node 50**
   The effectiveness of the BHO-DCNN approach with a number of alive nodes, delay, normalized energy, as well as throughput is depicted in the Figure 5.6 . Before reaching round 350, the number of nodes is 50, in addition the nodes diminishes slowly before reaching round 1500. When the size of the population is 25, 50, 75, and 100, then the number of attained alive nodes are 30, 40, 41 and 43, which is illustrated in the Figure 5.6 to represent the superior effectiveness. The delay of the BHO-DCNN algorithm is examined for the numerous population sizes such as 25, 50, 75, and 100, then the delay at the 1500th round is 3.26 ms, 2.87 ms, 2.34ms, and 1.68 ms, in which the preliminary population size of 25 is high when compared with the 100 size of the population that is shown in Figure 5.6. The normalized

energy of BHO-DCNN is evaluated for several population sizes, shown in the figure, in which it attains 0.05 J at the population size 25 and 0.16 J at 100 population size. The throughput of the BHO-DCNN algorithm is considered for the different sizes of population such as 25, 50, 75, and 100, here throughput rate reduces from 1 before obtaining round 300, and obtains 0.98 % at round 500 for population 25, and acquires 0.96 % for population 100 at round 800 that is represented in Figure 5.6.

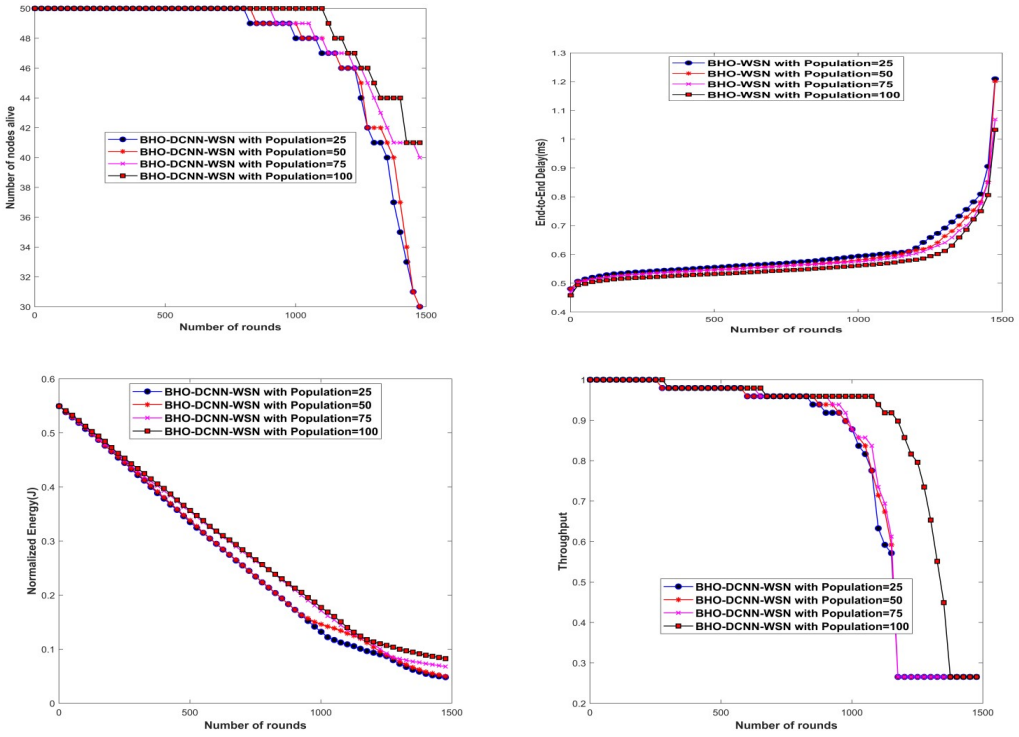


Fig 5.6: Analysis for 50 Nodes: a) Number of Alive Nodes b) Delay c) Normalized Energy d) Throughput.

2. **Setup for Node 100**

The effectiveness of the BHO-DCNN technique at node 100 is represented in the Figure 5.7. When the alive node is 100, then the total node decreases slowly before reaching around 1500. When the population size is 25, 50, 75, and 100, then the attained alive nodes are 54, 56, 60, and 82, which is illustrated in figure to show the superior effectiveness. The delay of BHO-DCNN is examined for numerous sizes of populations such as 25, 50, 75, and 100; then it attains the value of 4.04 ms, 2.60 ms, 2.32 ms, and 2.05 ms at the 1500th round that is shown in the Figure 5.7. The normalized energy of BHO-DCNN is examined and thus, the acquired values at the population sizes 25 and 100 is 0.04 J and 0.15 J that is depicted in Figure 5.7. Finally, the throughput of BHO-DCNN is examined at several sizes of the

populations, such as 25, 50, 75, and 100, in which its rate diminishes slowly from 1 before obtaining the 125th round. Thus, the acquired throughput for populations 25 and 100 at round 500 and 1100 is 0.98 %, and 0.92 % that is represented in Figure 5.7.

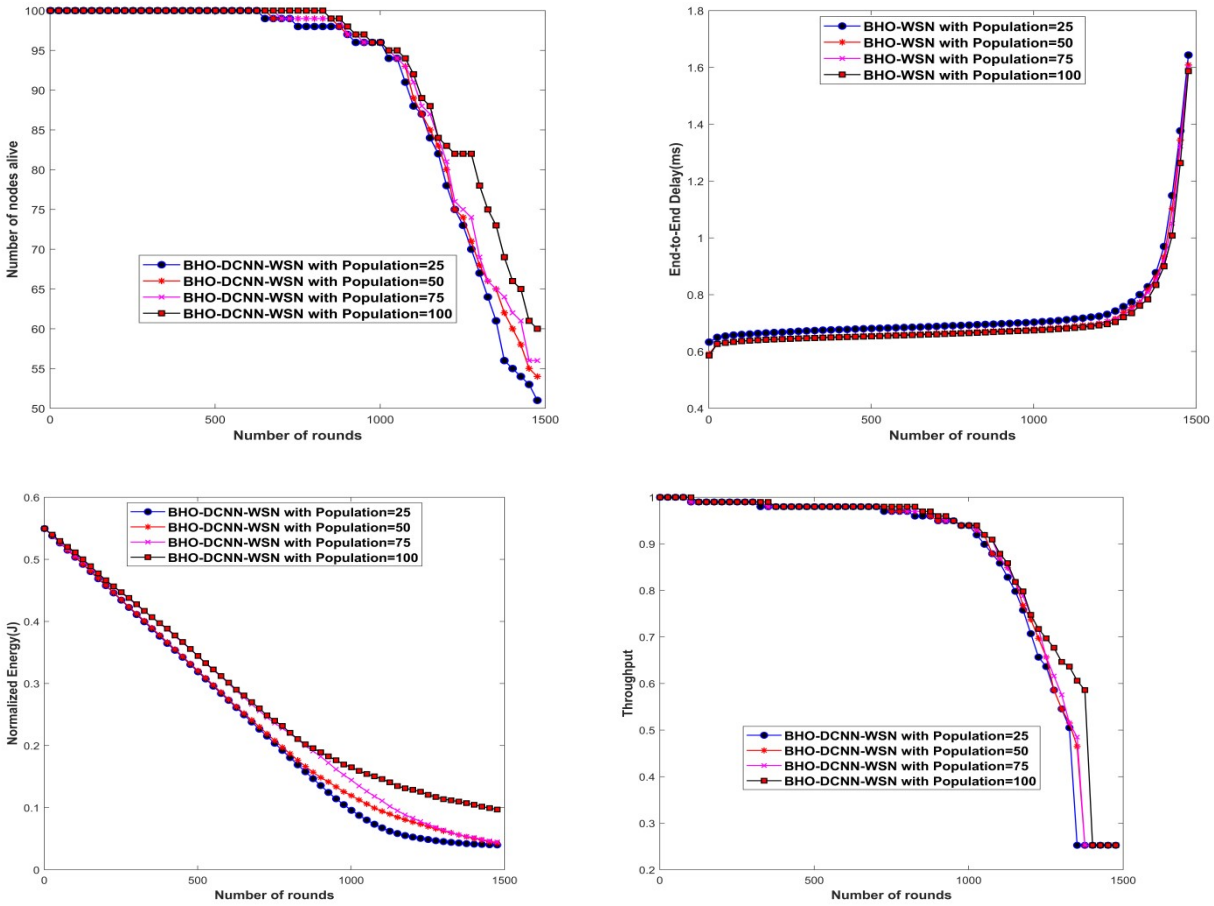


Fig 5.7: Analysis for 100 Nodes: a) Number of Alive Nodes b) Delay c) Normalized Energy d) Throughput.

3. **Setup for Node 200**

Figure 5.8 depicts the analysis of the BHO-DCNN method with 200 nodes. The alive nodes accessible for Md1, Md2, Md3, and Md4 at around 1500 are 73, 83, 84, and 84, respectively. The BHO-DCNN holds 137 alive nodes at the final round of 1500, which represents the efficiency of the developed technique that is depicted in Figure 5.8. The entire energy of the network at round 1500 for Md1, Md2, Md3, and Md4 are 0.036 J, 0.040 J, 0.046 J, and 0.047 J, whereas the available energy

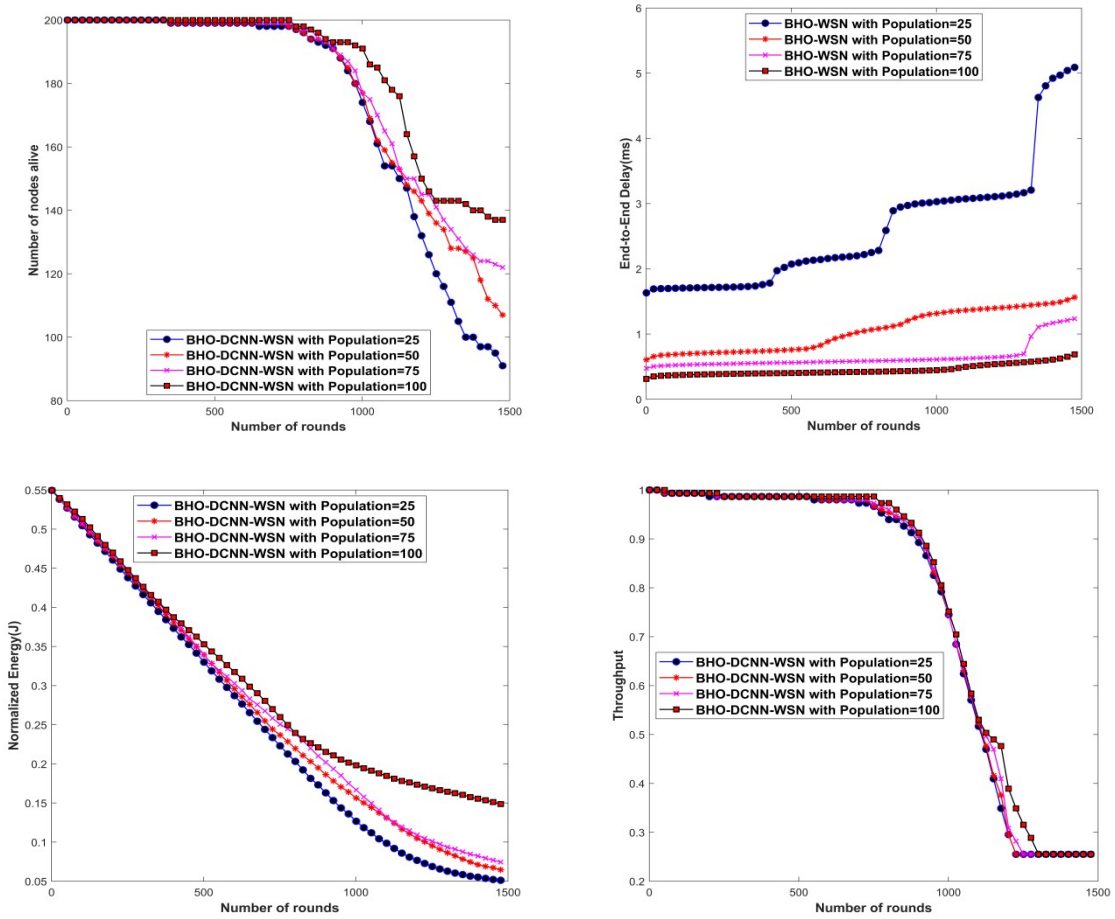


Fig 5.8: Analysis for 200 Nodes: a) Number of Alive Nodes b) Delay c) Normalized Energy d) Throughput.

of the developed BHO-DCNN is 0.149 J, which is better than the existing methods that is represented in Figure 5.8. The delay at the node of 1500 for the developed BHO-DCNN is reduced as 0.687049 ms, while comparing with the other existing approaches of Md1, Md2, Md3, and Md4 is 19.74291 ms, 12.51347 ms, 6.957651 ms, and 6.617461 ms, which is shown in Figure 5.8. Throughput for approaches like Md1, Md2, Md3, and Md4 at round 1500 are 0.16 bps, 0.21 bps, 0.26 bps, and 0.26 bps is shown in Figure 5.8. Therefore, the developed BHO-DCIN optimization technique exhibits better outcomes.

5.6.5 Analysis using Traditional Models with 200 Nodes

To prove the efficacy of the developed system, the analysis is done by replacing the deep CNN by AlexNet and GoogleNet with the usage of 200 nodes. The performance metrics considered for analysis is Alive nodes, Normalized energy, delay and throughput.

Analysis of AlexNet

Figure 5.9 depicts the analysis of developed approaches with 200 nodes and is tabulated in Table 5.1, and the analysis of the approaches is depicted in Figures a), b), c) and d). The alive nodes for Md1, Md2, Md3, and Md4 at the 1500th round are 63,73, 74, and 74, respectively. On the other hand, the AlexNet holds 127 alive nodes at the final round of 1500, which represents the effectiveness of the developed technique that is depicted in Figure 5.9. At round 1500, the network energy for Md1, Md2, Md3, and Md4 are 0.02 J, 0.02 J, 0.03 J, and 0.03 J, and for the AlexNet, the available energy is 0.13 J, which is improved than the existing methods that are represented in Figure 5.9. The delay at the node of 1500 for the AlexNet is reduced as 1.019 ms, while comparing with the other existing approaches of Md1, Md2, Md3, and Md4 is 2.44 ms, 13 ms, 8.2 ms, and 7.3 ms, which is shown in Figure 5.9. At round 1500, the performance of throughput for approaches like Md1, Md2, Md3, and Md4with 200 nodes are 0.125 bps, 0.185 bps, 0.185 bps, and 0.205 bps, respectively, while the AlexNet attained the throughput of 0.23 bps is represented in Figure 5.9. Therefore, the AlexNet displays a better consequence while comparing with preceding approaches.

Table 5.1: Comparative Analysis using AlexNet for Nodes = 200

Analysis using AlexNet for Nodes = 200				
Techniques	**Alive Nodes**	**Normalized Energy**	**Delay**	**Throughput**
M_d1	63	0.019	2.441	0.125
M_d2	73	0.023	12.99	0.185
M_d3	74	0.029	8.19	0.185
M_d4	74	0.029	7.32	0.205
AlexNet	127	0.130	1.02	0.239

Table 5.2: Comparative analysis using GoogleNet for Nodes = 200

Analysis using GoogleNet for Nodes = 200				
Methods	**Alive Nodes**	**Normalized Energy**	**Delay**	**Throughput**
M_d1	68	2.44	19.75	0.133
M_d2	78	12.9	12.51	0.193
M_d3	79	8.1	6.958	0.193
M_d4	79	7.2	6.617	0.213
GoogleNet	132	0.9	0.687	0.247

Analysis of GoogleNet

Figure 5.10 depicts the analysis of developed method and is tabulated in Table 5.2. The alive nodes for Md1, Md2, Md3, and Md4 at round 1500 are 68, 78, 79, and 79. On

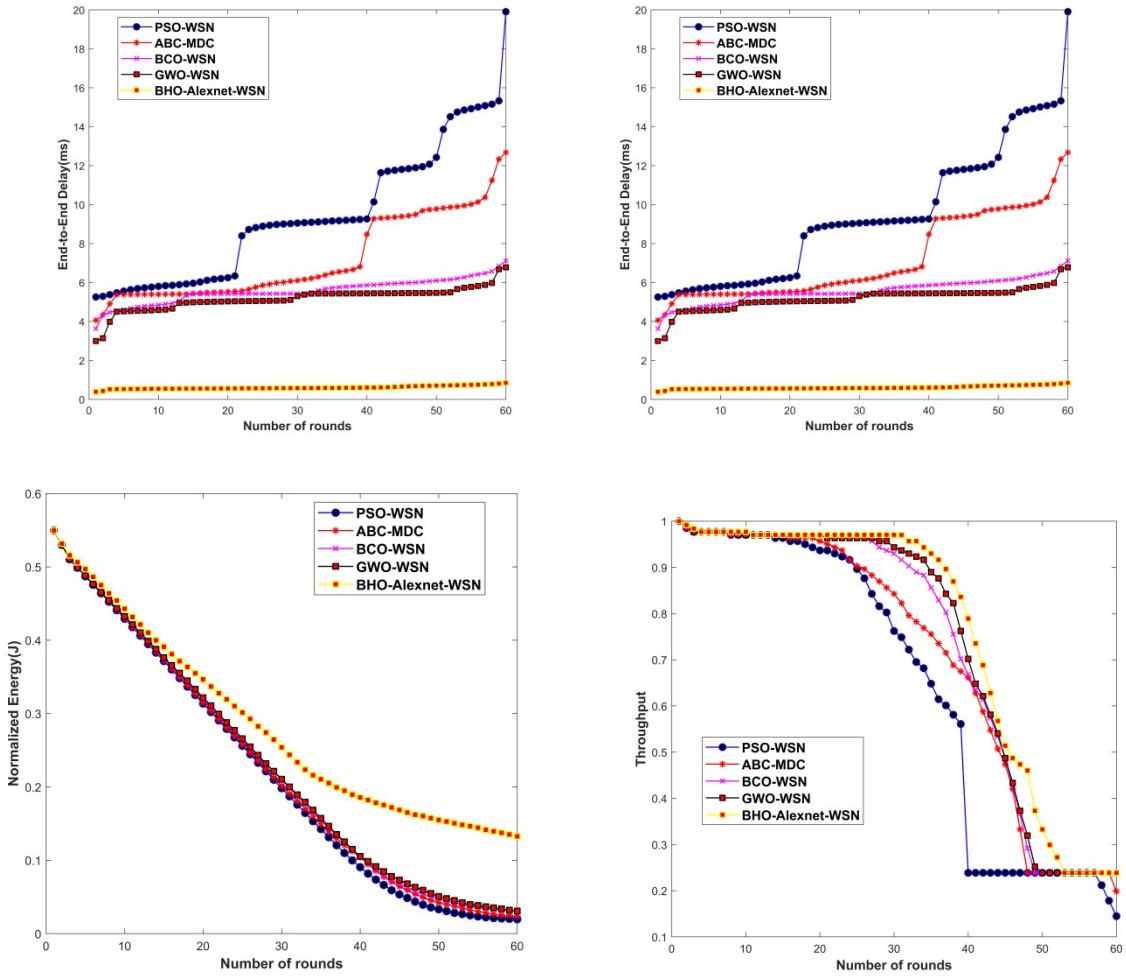


Fig 5.9: Analysis with 200 Nodes using AlexNet: a) Alive Nodes b) Normalized Energy c) Delay d) Throughput

Table 5.3: Comparative Analysis of the BHO-DCIN for Nodes = 50, 100 and 200.

Method	Alive Nodes			Normalized Energy			Delay			Throughput		
	50	100	200	50	100	200	50	100	200	50	100	200
M_d1	24	44	73	0.036	0.036	0.036	250.5	399.34	19.75	0.26	0.25	0.16
M_d2	28	46	83	0.039	0.037	0.040	6.293	6.219	12.51	0.26	0.25	0.21
M_d3	29	48	84	0.039	0.038	0.046	4.923	5.674	6.958	0.26	0.25	0.26
M_d4	29	50	84	0.041	0.038	0.047	4.717	5.006	6.617	0.26	0.25	0.26
M_d5	43	82	137	0.162	0.148	0.149	1.675	2.055	0.687	0.26	0.25	0.26

the other hand, the GoogleNet holds 132 at the final round of 1500, which is depicted in Figure 5.10. The total network energy at round 1500 for Md1, Md2, Md3, and Md4 are 0.27 J, 0.31 J, 0.37 J, and 0.37 J, whereas the GoogleNet attained the value of 0.138 J, which is represented in Figure 5.10. The delay at the node of 1500 for the GoogleNet is reduced as 0.9 ms, while comparing with the other existing approaches of Md1, Md2,

Md3, and Md4 is 2.44 ms, 12.9 ms, 8.1 ms, and 7.2 ms, which is shown in Figure 5.10. Throughput for the approaches, such as the Md1, Md2, Md3, and Md4 at 1500th round are 0.133 bps, 0.193 bps, 0.193 bps, and 0.213 bps, while the GoogleNet acquired the throughput of 0.247 bps that is represented in Figure 5.10. Therefore, the GoogleNet model exhibits better outcomes when compared with previous methods.

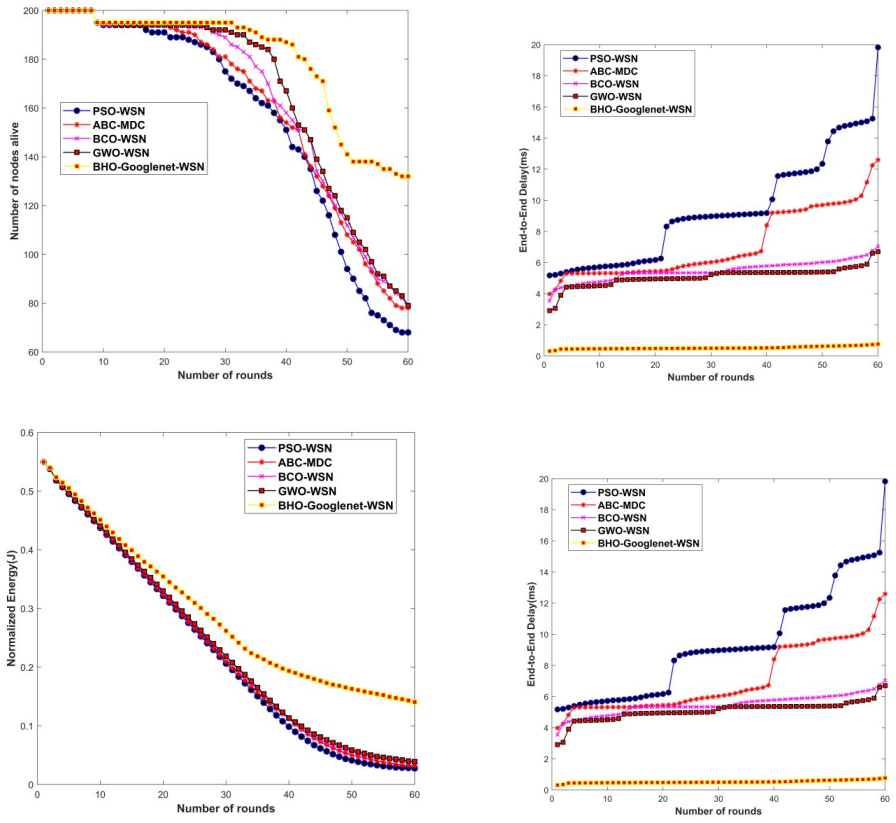


Fig 5.10: Analysis with 200 nodes using GoogleNet: a) Alive Nodes b) Delay c) Normalized Energy d) Throughput

5.7 Summary

In this research, the Mobile Agent is focused as well as data aggregation is intended accordingly. The optimal as well as efficient intrusion detection is achieved through the Border-Hunting Optimization, which supports the selection of a mobile agent as well as the Cluster head. When the Luster head selection is done depending on the developed optimization of Border-Hunting, the deep CNN is utilized to detect the intrusion that arose in the network while transferring the data from sender to receiver. The attained number of alive nodes with the developed BHO-DCNN algorithm for nodes 50, 100, and

200 at a population rate of 100 is 43, 82, and 137. The delay of the proposed BHO-DCNN algorithm at nodes 50, 100, and 200 at a population rate of 100 is 1.67549 ms, 2.055 ms, and 0.688 ms. The normalized energy at nodes 50, 100, and 200 for the BHO-DCNN at a population rate of 100 is 0.1622 J, 0.1490 J, and 0.1486 J. Finally, the throughput of the developed BHO-DCNN at nodes 50, 100, and 200 at a population rate of 100 is 0.2653 bps, 0.2525 bps, and 0.2550 bps. In the future, this research is to be extended with the other classifier to get better performance of the entire system.

Chapter 6

Self-configuring mobile agent-based intrusion detection using hybrid optimized with deep LSTM

This chapter introduces a self-configuring mobile agent-based intrusion detection system (MA-IDS) enhanced with Deep Long Short-Term Memory (Deep LSTM) networks and hybrid optimization. Designed for dynamic networks like Internet of Things (IoT) and Wireless Sensor Networks (WSNs), this system autonomously adapts to changing conditions, improving real-time intrusion detection accuracy. By leveraging mobile agents and advanced machine learning, the approach offers a robust, scalable solution to modern network security challenges.

Sensor nodes can be deployed in harsh or hostile environments in many applications, making these nodes more prone to failure. The illegal movement monitoring within the sensor networks is a most challenging problem. The attacker prefers mobile malicious nodes to maximize his impact. For a dynamic environment, a promising technology of sensor networks is expected for Intrusion detection. After verification, multi-mobile agents utilize many approaches that collect data from sensor nodes. However, these approaches are inefficient in verifying all the network sensor nodes (SNs) due to their high delay, energy consumption, and mobility.

The proposed Dunnock Ibis optimization Long Short Term Memory (LSTM) model (DIO opt LSTM) solves this problem. Here, the sensor nodes are grouped into clusters; hence, the mobile agent performs verification only of the cluster heads instead of verifying all the SNs. The proposed DIO optimization combines the unique behavior of Egret Swam and Ibis optimization algorithm, which efficiently tunes the LSTM classifier, resulting in the model providing better convergence. The simulation results show the proposed system shows a better result than the existing system by utilizing the database IDS 2018 Intrusion CSVs; the analysis is done based on performance metrics such as End-end-delay (ED), normalized energy (NE), and throughput. At 200 nodes and 1500 rounds, the DIO opt LSTM method has efficiently performed 146 numbers of alive nodes, 0.46ms of delay, 0.15J of normalized energy, and 0.89 bps of throughput.

6.1 Introduction

Due to enabling attractive technologies, pervasive computing scenarios, and supporting next-generation ubiquitous, Wireless Sensor Networks (WSNs) have witnessed explosive growth in research. To organize in a particular region of interest, WSN contains many autonomous sensor nodes performed by a task-specific executable code. These deployed sensor nodes do not necessitate centralized control or common infrastructure and communicate using wireless channels. Many intermediate relay nodes possibly by involving to may cooperate with the nodes by relaying or forwarding each other packets; in the literature, several approaches have been proposed to model the reliability of WSNs; these approaches include probabilistic models, statistical models, and simulation-based models.

Various factors that affect the reliability of WSNs are captured by these models, which include the sensor node characteristics, communication protocols, and network topology[63]. By moving among the sensor nodes, the mobile agent collects the data from the sensor network[49]. While sensing the environment, the sensor nodes generate highly redundant data, and these nodes are located nearer to each other while sensing the environment. Due to this, a high volume of data is transmitted to the sinks.

The malicious node (MA) visits the sensor nodes individually, collects the sensed data, and performs a data aggregation function at each node. It will be eliminated automatically if redundancy exists in the data, and hence, using the mobile agent reduces the volume of data transmitted to the sink. The link bandwidth is more limited in the case of data traffic of the sensor network than the network capacity. Attacking the data by the attacker may be easy, but attacking the computation machine is difficult.

Instead of moving data to the computation machine, mobile agents move the computation machine to the data. Mobile agents offer sensor network security solutions in this manner. The mobile agent is capable of perceiving its surroundings, and it can dynamically adjust to changes in it[54]. Unknowingly, if an internal or external malicious node compromises a sensor node, the malicious adversary node can seriously disrupt an agent's execution by gaining access to the security keys. The information can be impacted by altering or corrupting the code or state information of the mobile agent, preventing the agent from executing its code or rejecting agent service requests[44].

Any layer of the network's protocol stack is vulnerable to attack. Attacks from Sinkhole and Sybil are the most destructive. A sinkhole attack is a prevalent and severe form of network layer attack. The major goal of the sinkhole assault is to lead all nodes in a

specific area astray such that the routes form a symbolic attack with the challenger at its center[64]. After obtaining traffic, the node will either drop or modify it illegitimately. For instance, an attacker in RPL can advertise themselves as a selected routing node by employing deceptive advertisements to take advantage of the ranking algorithm. The adversary tricks other nodes into transmitting data by pretending to be a member of the optimal route. By doing this, the adversary can undermine the lifetime and performance of the network and launch additional network attacks. Comparably, by generating several identities and flooding the network with undesired traffic, a Sybil attack might shorten the lifespan of a network[65].

6.1.1 Limitations of Existing Methods

According to the findings for test data that is typically labeled, the conventional approach of assessing detection methods and procedures offers good efficiency and accuracy. The fundamental premise is that a specially designed DDoS detector would consistently execute generalization and accurately anticipate the upcoming attack in the actual world. Since DDoS attack detection is typically considered a binary or multiclass classification problem, this approach makes sense when applying machine learning approaches to pattern recognition difficulties. However, this assumption is difficult to meet because DDoS attacks are evolving daily, and network traffic has become more sophisticated and subject to change anytime.

As a result, the DDoS assault detection system may make numerous mistakes during the real detection process, such as true negatives and false positives, and certain new standard patterns or unknown attacks may differ from the patterns discovered from the initial training data[54]. Integrity protection and symmetric key authentication alone are insufficient to stop the incursion of such malicious nodes; instead, nodes' dependability must be assessed, and compromised nodes should be avoided when creating an MA route or having an agent visit source nodes[44].

Due to these models' subpar performance on large datasets, research has switched to employing sophisticated deep learning (DL) in constructing intrusion detection systems. Nonetheless, the advent of several innovative and advanced assaults presented two additional difficulties: flexibility and scalability. However, most DL-driven solutions are challenging to scale up and necessitate regular model retraining[66].

6.1.2 Contribution

The primary goal of the research is to provide more accurate and alternative solutions for IDS using the proposed DIO opt LSTM model. Dunnock optimization and Ibis optimization features of searching, hunting, and observing are combined to create the DIO. The primary contribution of the current research consists of followings:

1. **Dunnock Ibis optimization (DIO)** The typical hybridization of Dunnock and Ibis results in DIO. The Dunnock faces several difficulties due to predators and other animals; to improve the Dunnock searching strategies, the Egret Swam optimization and Ibis optimization's fitness are combined and named Dunnock Ibis optimization. Thus, Dunnock Ibis have the ability to disguise themselves as enemies based on their fitness. This boosts the searching speed and attacking capacity by reducing time complexity, resulting in faster convergence and reaching the global destination.

2. **Dunnock Ibis opt LSTM model(DIO opt LSTM)** The hyper layers of the LSTM are successfully tuned through Dunnock Ibis optimization, which effectively detects the intrusion and reduces the problem of class imbalance. By leveraging Dunnock Ibis Optimization to fine-tune the hyper parameters of an LSTM model, we significantly enhance the model's performance in detecting intrusions and addressing the class imbalance problem. This results in a more accurate, reliable, and robust intrusion detection system capable of effectively distinguishing between normal and malicious activities in a network environment.

6.2 Motivation

There are numerous innovative ways to network ID technologies. However, each has its own suitable environment. Improving the detection rate of network ID with a single class characteristic or detection model is extremely tough. This section describes the tactics taken by researchers to increase the effectiveness of the ID model.

6.2.1 Literature review

An advanced mobile agent-based ID system was developed by [53] to protect the network of connected medical equipment. Although cryptography-based solutions are often costly in terms of computation and challenging to implement on medical devices with limited resources, this system achieved high detection accuracy while consuming no resources. [54] developed a mobile agent for cluster-based WSNs with three-step negotiation to identify trusted CH and detect mobile hazardous nodes in order to obtain data exclusively

from trusted nodes. Despite the fact that the mobile agent just aggregates data with the CH, it prevents the majority of attacks from occurring here. It improves detection accuracy because it is difficult to relocate mobile agents among all of the SN.

A new ID scheme for IOT was created[67]. It was a lightweight ID method combined with two algorithms that could improve detection effectiveness and reduce the false positive rate. Still, it presented a new challenge due to its limited data processing capacity for large, complex decision-making. A strategy for ID and prevention system (SA-IDPS) was developed by [68] to mitigate attacks in MANET by ML approaches, which enhanced the attack detection rate and reduced the false alarm rate. but had a high computational complexity in this model. [69] designed an intrusion prevention system to achieve a network-based intrusion prevention system to protect the top-to-bottom engineering that succeeded in using high-performance computing to scale its operation, but it has certain over-fitting problems. [70] created the ad-hoc mobile-based edge cloud model, which integrated a multi-objective algorithm and genetic solution to provide intelligent decisions from the ad-hoc mobile edge cloud devices.

However, selecting the best distribution of chunks on the different nodes is a challenging problem. Smish Logish-Graph Attention Network (SL-GAT) is an MA-based intrusion detection framework which is inaugurated by [71]. The SL-GAT performs better than the prevailing classifiers and provides better security to the mobile agents. However, the system utilizes more resources, which causes costs.[72] demonstrated a border-hunting optimized deep CNN (BHO-DCNN) model that detects mobile agents-based intrusion in WSNs and provides an enhanced rate of convergence with the help of optimization in case of detection. However, more integrated optimization is needed to improve the performance of DCNN.[66] introduced an Attention-based Deep-Q-Network (A-DQN) for detecting intrusion in multiple agent systems. Moreover, each agent is separately learned, and an agent-specific attention component is presented at central IDS, which improves the scalability of the distributed network model. The key limitation of the model is the compromise of routers because the routers have potentially exploitable services with a lower number than the hosts.

Intrusion detection from a mobile network teaching system based on an intelligent algorithm, developed by [73]. This is used to effectively detect the intrusion behavior of the system with a low false alarm rate and high detection accuracy, which assures the system's stability and security.[74] created a Framework for DL-based IDSes using CIC-IDS2017 and CSE-CIC-IDS2018 datasets and achieved an average detection rate of above 95 % .

The DL-based IDSes reduce the computational cost and time of the network, but these

frameworks are more suitable for labeled datasets.[75] introduced an ensemble model, which is a combination of ML and sequential feature selection techniques. This model reduces the computation and transmission costs and false alarm rates.

In such cases, this method may not be completely neutralizing. Venkates et al. delivered a wireless intrusion detection system named Neuro Deep Learning (NDL) that distinguishes the attacks in mobile ad-hoc networks [76]. The NDL model improves the identification and diminishes the rate of false failure and caution, but the DL techniques need a large amount of data for better training. Then, the literature review is analyzed in Table 6.1.

6.3 Challenges in Dynamic Environment

Spread spectrum-based systems, however, face problems with code management and dissemination. A further issue for low-cost and resource-constrained IoT devices is that most of these techniques necessitate placing several antennas at the transmitter.

1. It is difficult to identify intrusions with a high detection rate and few false positives simultaneously.
2. Finding the optimal distribution of chunks across the various nodes that produce the greatest results is a complex task. also, detecting unauthorized movement within sensor networks is equally challenging.
3. It is difficult to discern between regular and attacker nodes using packet features. To lessen the effects of any harmful activities, it's critical to quickly identify attackers in a network.
4. In a highly dynamic environment, selecting an algorithm for the successful ID of any specific intrusion can be challenging.

6.4 Mobile Agent-based ID using DIO opt LSTM model

The research aims to create a self-configuring mobile agent-based ID system using hybrid optimized deep LSTM. Many different SNs are present in wireless networks and use data from numerous sources. Because the SN are moving objects, the cluster head (CH) receives the data that is stored inside each one. In the wireless network that is situated in a particular area, there are numerous CHs. Before sending the data to the base station

(BS), the CH aggregates data from the sensor nodes (SN). The optimized CH selection approach chooses the CH based on a multi-objective function that considers the trust factors and energy, distance, delay, energy, traffic, and TP.

The transmitted data is combined and supplied to the sink node through the mobile agent, where it is classified using the proposed Hybrid deep LSTM classifier. The standard features obtained from the Dunnock and Ibis optimizations are the basis for this optimization's development. In the end, the proposed optimization-based hybrid deep LSTM classifies the node as benign or malicious, and if the node is benign, further communication with it will be prevented. To increase the identification of new incursions, we train a model that is based on the self-configuring concept.

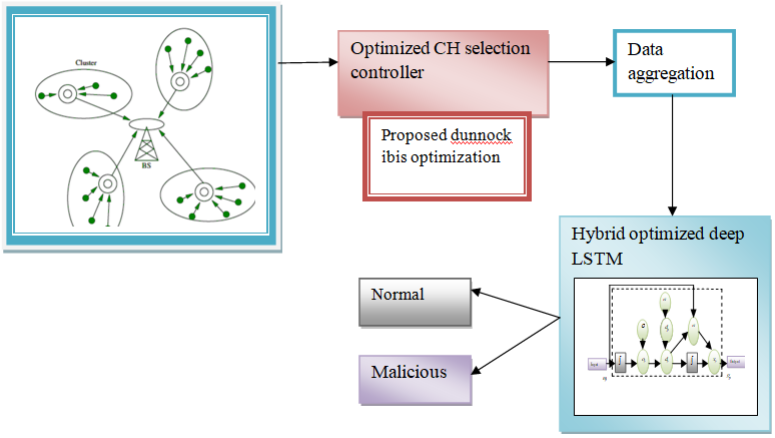


Fig 6.1: Architecture of the proposed Mobile Agent-based ID

The WSN comprises SN, CH, sink, and administration nodes. The WSN is clustered to offer easy administration processing, ensuring the network's dependable performance. In the monitoring area, a large number of sensing nodes have been deployed, and they have formed a network. Through a multi-hop relay that links to the BS via satellite or the Internet, the CH nodes send the data they have gathered to the sink node. The user can remotely configure or administer the network and send tasks for monitoring through the BS. This model consists of a BS, a small number of dispersed sink nodes, and a large number of SN disseminated across the observation area.

Each component serves the following purposes: In a WSN, a sensor node's main job is to gather data from each range, process it, and then transmit it to an upper node. The BS receives the data information that the sink node has combined from the CH sensor. Common SN and CH nodes make up this part. The network's operational state is monitored by the user-facing BS, which also performs ID and analysis on the data sent by the sink nodes.

The four underlying presumptions of the WSN environment are as follows:

1. A clustered network is the WSN. Direct communication between the CH nodes and common nodes is possible.
2. Every node in the cluster is fixed, has a unique identifying ID and only belongs to one cluster.
3. The sink and CH nodes have direct and indirect communication channels with the BS and other CHs, respectively.
4. The CH node must be identical to the common node, and the sink node must be more resourceful and energetic.

6.4.1 System Model

The WSN consists of m number of nodes. These SNs successfully send the sensed data to the sink node T_f. The uniform distribution, in which the data is disseminated at the highest radio level with the dimensions of J_k and J_s in meters, determines the transmission pattern for all SNs. The SN forms a cluster that is represented by O_k. The term P_o^k designates the identical CH in each cluster. CH-oriented routing is a brand-new method for sending data from all nodes to the BS. Here, P_{RT} denotes the distance between the BS and CH, while E_{DM} represents the distance between the D^{th} normal node and the N^{th} CH.

The key benefit of this model is improvement in life span and structural data flow.

6.4.1.1 Energy Model

Transmitter and the receiver determine which of the two channels in the radio model-free space (m^2 path loss) and multi-path fading (m^4 path loss) is used. When the distance is below a threshold value (m_0), the free space channel is used; otherwise, the multi-path fading channel is used. As demonstrated in equation6.2, the amount of energy needed to convey a k-bit packet over a distance n is as follows

$$G_{FC}(L,m) = \begin{cases} L \times G_f + L\times \in xy \times m^2, & m < n_o \\ L \times G_f + L\times \in_{hq} \times m^4, & m \geq n_o \end{cases} \tag{6.1}$$

The transmission energy G_{FC} and distance m are both present in the aforementioned equation.G_g is a circuit's transmitter or receiver's energy consumption. In the free space and multi-path models, the transmitter amplifier's energy requirements are ${}_xy$ and ${}_hq$,

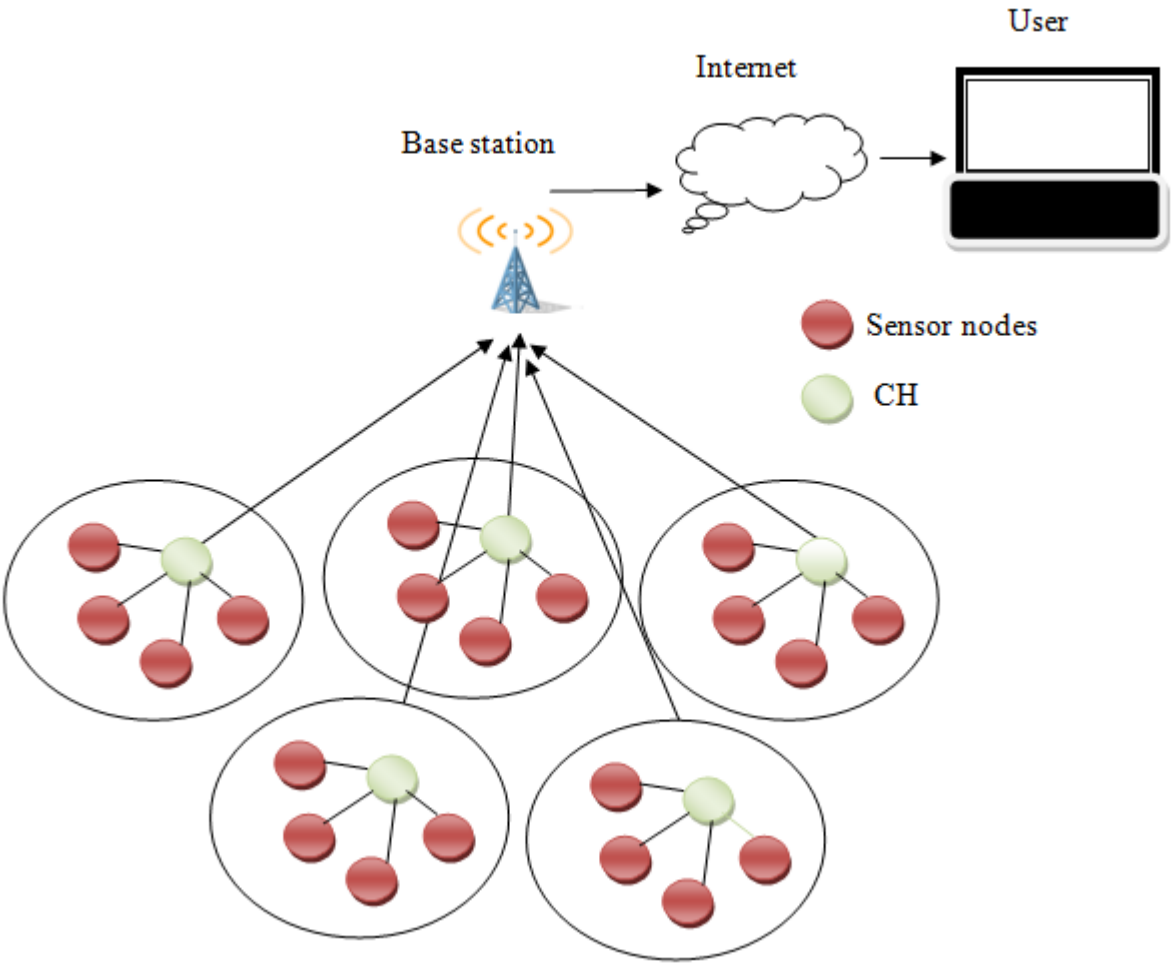


Fig 6.2: Architecture of the WSN System Model

respectively.m_o is a threshold separation determined as follows: The energy needed to receive a k-bit packet in a radio model is shown in the equation

The transmission energy G_{FC} and distance, m, are both present in the aforementioned equation. G_g is a circuit's transmitter or receiver's energy consumption. The transmitter amplifier's energy requirements in the free space and multi-path models are ϵ_{xy} and ϵ_{hq}, respectively. m_o is a threshold separation determined as follows:

$$m_o = \sqrt{\frac{\epsilon_{xy}}{\epsilon_{hq}}} \tag{6.2}$$

The energy needed to receive a k-bit packet in a radio model is shown in equation 6.5.

$$G_{TC}(L) = L \times Gg \tag{6.3}$$

G_g is influenced by signal processing, modulation, digital coding, and filtering factors.

In common transmission, all nodes employ the range m_r, and $Delta$ denotes the impact of interference. When node x_p certain sub channel over the transmits node x_q, then the transmission is effectively done by following two conditions:

$$|x_p - x_q| \leq m_r \tag{6.4}$$

$$|x_p - x_q| \geq (1 + \Delta)m_r \tag{6.5}$$

6.4.2 Problem Formulation

Under delay constraints minimizes of tour path for WSN, round means the mobile sink travels from the start point to CH and back to the start point. The total energy consumption of SN is,

$$EC_{SNp} = \sum_{e \in CH} \left(L + G_f \times m^2\right) \times x_{qe} + \sum_{q=SN} L \times x_{qp} \tag{6.6}$$

Moreover, each SN is assigned to only one CH and measures the binary component Φ_{pq} which is expressed by,

$$\Phi_{pq} = \begin{cases} 1, & \text{if node as signed to CHq} \\ 0, & \text{otherwise.} \end{cases} \tag{6.7}$$

6.5 Proposed Dunnock Ibis Optimization

The performance parameter computed for this deployment is analyzed by attack type on a node based on consumed energy criteria, an essential metric for dependability evaluation. The coordinates of attacks are normalized using the feature extraction technique to get the trained data. A subset of the WSN dataset is taken to extract the feature from all the attack cases that are classified correctly. The performance metric of features on

the machine learning model is used and compared Dunnock Ibis optimization (DIO), a typical combination of Egret Swarm Optimization algorithm (ESO) [78] Ibis [79] is used to transport the information to the needed node by selecting the CH at random from the network and fine-tuning the hyper layers of the LSTM model. The Dunnock behavior involves reaching the global optimum for food, and Dunnock encounters several challenges from predators and other animals. To improve the Dunnock searching strategies, Egret swarm optimization and Ibis optimization are combined to improve fitness, and as a result, Dunnock Ibises have the ability to hide from obstacles based on their fitness. Because of the reduced time complexity and faster convergence, this arrives at the global destination earlier, which boosts searching speed and attacking capacity.

Motivation

The four bird collective egret species are the great Egret, the Middle Egret, the Little Egret, and the Yellow-billed Egret. The prey-catching and adapting capacity of the Great Egret are 50 % more efficient than other egrets. Great Egret has a higher exertion strategy to pursue prey aggressively and capable of capturing larger prey, nevertheless difficult to travel through an identical place multiple times for large prey. To overcome this problem, the searching and hunting behavior of the Egret swarm is integrated with the Bald Ibis. Northern Bald Ibises are migration birds with better searching abilities, and they travel long distances with the help of wing creatures without stopping in any middle. Hence, the proposed DOI optimization provides faster convergence and reduces the model's time complexity.

1. **Initialization**

   An arbitrary population of sparrows serves as the basis for the sparrow search algorithm. The following matrix is used to represent the population in this algorithm, which displays the position of the sparrows:

$$D = \begin{bmatrix} D_{1,1} \ldots\ldots\ldots D_{1,m} \\ \vdots\vdots\vdots\vdots\vdots\vdots\vdots\vdots\vdots \\ D_{g,1}, \ldots\ldots\ldots D_{g,m} \end{bmatrix}$$

Fig 6.3: Positions of Sparrows

   Where g denotes the total number of sparrows and the dimension of the decision factors. Therefore, the producer sparrows supply foraging areas for the scavengers with a significant amount of energy stored. They need to locate the regions with abundant food resources. The amount of available energy reserves is determined by

$$T_D = \begin{bmatrix} t([D_{1,1} \ D_{1,2} \ D_{1,m}]) \\ t([D_{2,1} \ D_{2,2} \ D_{2,m}]) \\ t([D_{g,1} \ D_{g,2} \ D_{g,m}]) \end{bmatrix}$$

Fig 6.4: Cost of Sparrows

evaluating the cost values of individuals. The cost value of the sparrows is described by the vector below:

2. **Position update of producer**

   The SSA, a competitive strategy, allows individuals (producers) with lower cost value a better chance of getting food in the solution space. Instead of scavengers, producers look for food in various search locations. The following mathematical model represents the producer's position:

$$D_{u\nu}^{x+1} = \begin{cases} D_{u\nu}^{x} \times \exp\left(-\frac{u}{\delta \times iter_{\max}}\right) & \text{if } N < HI \\ D_{u\nu}^{x} + J \times K & \text{if } N \geq HI \end{cases} \tag{6.10}$$

   Where the current iteration is denoted as x, D_{uv}^{x+1} is the value for the v^{th} dimension of the u^{th} persons at x th iteration, iter max is a constant, $v = 1, 2, \ldots .m.$ is a random number, J is a properly distributed random number, and K denotes an m-dimensional vector whose elements are all one. In this instance, the safety threshold, HI, is a value between $[0.5, 1]$, while the alert value, N2, is a number between $[0, 1]$. if $(N_2 < HI)$, there are no nearby predators, and the producer has moved to wide search mode. In contrast, if ($N_2 \geq HI$), some individuals have already detected the predator, necessitating a quick flight to another safe area for everyone. According to previous explanations, some scroungers continuously follow the producers so that when they find the producer with nice food, they immediately steal it.

3. **Position update of scrounger** According to previous explanations, some of the scroungers continuously watch the producers so that when they spot a producer with good food, they immediately move toward that location to compete for food. In this case, scroungers had trouble finding food because of their poor global search ability. The Ibis population's fitness is integrated to strengthen scroungers, causing the Dunnock Ibis to compete for food with the strong fitness that provides fast

convergence and good global search ability. If they won the competition, the producer might provide them with food; if not, they would still have to abide by the regulations. The scrounger's location has been revised as follows:

$$D_{uv}^{x+1} = 0.5 \left\{ \; J \times \exp\left(\frac{D_{\text{worst}}^{x} - D_{w}^{x+1}}{u^2}\right) D_{w}^{x+1} + |D_{tv}^{x} - D_{w}^{x+1}| \times F^1 \times K \right. \quad (6.11)$$

Where D_w denotes the producer's ideal founded position andD_worst denotes the current globally worst position. F depicts a vector in which the elements are randomly given the values 1 or 1. The following locations are attained by the sparrows when they first become aware of the threat: If a team member or a population member discovers the best solution, the real gradient is assigned to them. Q_Gbest denotes the flight direction of the best Ibis individual. The best fitness level for the egret team and the egret population is C_Gbest. Pseudo code for Dunnock Ibis Optimization is shown in Algorithm 6.5:

6.6 Cluster Head selection using the Dunnock Ibis Optimization algorithm

Every node in a group can be the CH. All nodes in the group must compare their remaining energy, and the node with the greatest remaining energy will be the CH for the round. The B_max indicates each node's maximum transmission range. All SNs that fall within theB_max range are included in the initial cluster. The highest energy node in each cluster determines the CH. Until there is no longer a node in the network, the same procedure is carried out on the remaining SNs.

The entire process for cluster construction and CH selection is shown in Figure 6.6. The CH is typically chosen depending on energy, distance, delay, energy, traffic, and TP. The developed DIO algorithm, on the other hand, was designed expressly for these networks. In place of delay, one new component is included as trust is considered to boost performance. Direct, indirect, existing, and historical trusts are all characteristics that depend on trust.

The objective functions for the CH selection divide the distance into the following two categories: inter-cluster distance and intra-cluster distance. Each clustered SN group, divided into smaller groups, is given a cluster leader. The cluster's data from each node is collected and provided as clustered data to the BS. The wireless transmission network

S.NO	Pseudo code for DIO opt LSTM
1	Initialization: $D = \begin{bmatrix} D_{1,1} \ldots\ldots D_{1,m} \\ \vdots \quad \vdots \\ D_{g,1} \ldots\ldots D_{g,m} \end{bmatrix}$
2	Position update of producer: $D_{iw}^{x+1} = \begin{cases} D_{iw}^{x} \times \exp\left(-\frac{u}{\delta \times iter_{\max}}\right) & if\ N < HI \\ D_{iw}^{x} + J \times K & if\ N \geq HI \end{cases}$
3	Individual fitness value: $T_D = \begin{bmatrix} t([D_{1,1}\ D_{1,2}\ D_{1,m}]) \\ t([D_{2,1}\ D_{2,2}\ D_{2,m}]) \\ t([D_{g,1}\ D_{g,2}\ D_{g,m}]) \end{bmatrix}$
4	$if(N_2 < HI)$
5	{
6	There is no predator nearby
7	}
8	Else
9	{
10	Some individuals have already discovered the predator
11	}
12	Position update of scrounger: $D_{iw}^{x+1} = 0.5 \begin{cases} J \times \exp\left(\frac{D_{worst}^{x} - D_{w}^{x+1}}{u^2}\right) \\ D_{w}^{x+1} + \left\lvert D_{iv}^{x} - D_{w}^{x+1} \right\rvert \times F^1 \times K \end{cases} + \frac{T_{Gbest} - T_M}{\lvert T_{Gbest} - T_M \rvert} * \frac{C_{Gbest} - T_M}{\lvert T_{Gbest} - T_M \rvert} + Q_{Gbest}, \quad \begin{matrix} if(u > g/2) \\ otherwise \end{matrix}$
13	$if(c > y/2)$
14	{
15	Lowest fitness value
16	}
17	Else
18	{
19	Highest fitness value
20	}
21	Termination

Fig 6.5: Algorithm for DIO opt LSTM

offers CH selection to deliver the most efficient and secure data communication network. Additionally, data travel between CHs reduces the energy used by the network. The DIO method is used in this study to calculate the CHs for each group of nodes. An Equation is necessary for the proposed DIO technique, which illustrates the resource process for CH selection.

$$VIXKS_{\text{QquFym}}^{\text{new}} = VIXKS_{\text{low}} + J_{o,p}^{F+1} \times (VIXKS_{\text{upper}} - VIXKS_{\text{low}} | \tag{6.12}$$

$$J_{o,p}^{F+1} = J_{best}^{F} + \gamma \left| J_{o,p}^{F} - J_{best}^{F} \right| \tag{6.13}$$

J_{best} represents the probable current optimal position, and the step size is determined by the parameter γ.

The clustering establishment needs to consider these objective functions: TP, traffic, energy, and distance. Following is a list of the expected clustering goals and a mathematical representation of the multi-objective function:

$$BZ = \gamma \times F_Q^u + \kappa \times E_Q^u + \chi \times EM_Q^u + \alpha Eb_Q^u + \beta r_Q^u \tag{6.14}$$

The developed DIO algorithm, in this instance, calculates the fusion parameters, denoted as $\gamma, \kappa, \chi, \alpha$, and β.

The CH selection thus establishes a distance between the BS and the sensors and distributes the energy use of the SN equally.

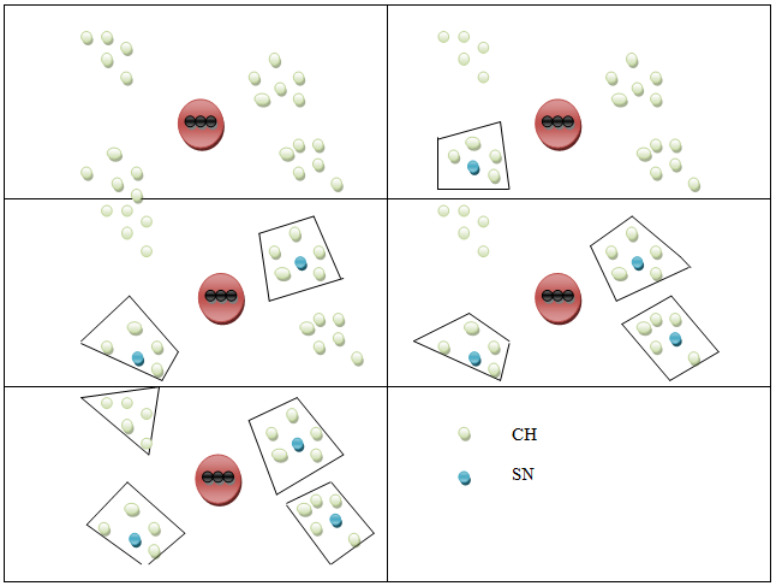


Fig 6.6: Architecture of step-by-step cluster creation and CH selection approach

6.7 Self-Configuration

The capacity of SNs in WSNs to autonomously and dynamically configure themselves for network functioning without requiring manual intervention or external configuration is called self-configuration. The shifting conditions and demands of the network entail the automatic construction and adaption of a number of network parameters, including node identities, routing paths, transmission power, and network structure. Due to the large-scale and dynamic nature of WSNs, where nodes may be placed in remote or inaccessible locations, self-configuration procedures are important. A model based on

the self-configuring idea is trained to improve the system's capacity to detect new incursions. The system constantly updates and adapts detection techniques based on new data and changing network conditions. The self-configuration procedure enables the ID system to learn and develop over time, assuring its effectiveness against new threats. Overall, the self-configuring mobile agent-based ID system uses a combination of data collection, aggregation, optimized selection, mobile agent-based data transmission, deep LSTM classification, and ongoing self-configuration to achieve its primary goal of detecting and preventing intrusions in wireless networks.

6.7.1 ID using Dunnock Ibis opt LSTM model

The most well-known model for sequence data training is the RNN. When trained with a large step size, the standard RNN struggles. The vanishing problem and the formalization of RNN are briefly covered in this section. We then describe long LSTM to address this problem. Unfortunately, a significant flaw brought on by a number of factors is the low sensitivity of the intrusion detection models currently in use. Compared to other optimization methods, the LSTM fitted with the Adam optimizer has a slower convergence rate. Additionally, the outcome of the model's accuracy is reduced because of the Adam optimizer's over fitting loss. To reduce the likelihood of entering local optima and modifying hyper parameters, the DIO optimizer is integrated into the LSTM, effectively detecting network intrusion. Moreover, DIO successfully tackles the high dimensional challenges, eventually resulting in a high convergence rate and increasing the LSTM model's accuracy.

1. **Recurrent neural network (RNN)** A traditional feed-forward neural network has evolved into the RNN. RNNs are better at simulating sequences since they have cyclic connections than feed-forward neural networks. Assume that the input, hidden, and output vector sequences are represented by M, G, and K, respectively. $M = (m_1, m_2, \ldots . m_F)$ describe the input order. The hidden vector sequence $(G = (g_1, g_2, \ldots g_F))$ and the output sequence vector $(G = (g_1, g_2, \ldots g_F))$ with $p = 1$ to P be calculated by a conventional RNN in the manner shown below.

$$g_p = \sigma \left(N_{mg} m_p + N_{gg} g_{p-1} + c_g \right) \tag{6.15}$$

$$k_p = N_{gh} g_p + c_k| \tag{6.16}$$

describe the input order. The hidden vector sequence and the output sequence vector are to be calculated by a conventional RNN, as shown in the equation.

Where$_c$ is a bias term, N is a weight matrix, and function A is a non linearity function. RNNs typically use Back Propagation Training Time (BPTT) to accommodate inputs with varied lengths. The model in BPTT is initially trained using the training set of data. For each step, the output error gradient is saved. Although the RNN is challenging to train, using the BPTT approach causes the gradient to explode or vanish.

2. **LSTM**

   Here, we explore the equations for computing the values of three gates and the cell state in the LSTM architecture.

$$o_p = \sigma\left(N_{mo}m_p + N_{go}g_{p-1} + N_{do}d_{p-1} + c_o\right) \tag{6.17}$$

$$_p = \sigma\left(N_{mh}m_p + N_{gh}g_{p-1} + N_{dh}d_{p-1} + c_h\right) \tag{6.18}$$

$$d_p = h_p d_{p-1} + \tanh\sigma\left(N_{md}m_p + N_{gd}g_{p-1} + c_d\right) \tag{6.19}$$

$$x_p = \sigma\left(N_{mx}m_p + N_{gx}g_{p-1} + N_{dx}d_p + c_x\right) \tag{6.20}$$

$$g_p = x_p \tanh\left(d_p\right) \tag{6.21}$$

The input gate, forget gate, output gate, and cell state are represented by the variables o, h, f, and d, respectively. A is the logistic sigmoid function. Weight matrices for peephole connections are denoted by N_{do}, N_{dh} and N_{dx}. Three gates govern the information flow in LSTM: (o, h, f). The input gate determines the input ratio. This ratio affects the solution of equation6.11, which determines the cell state. The previous memory g_{p-1} determines if the forget gate is successful. The input gate determines the input ratio. This ratio affects the solution of equation6.11, which determines the cell state. The previous memory g_{p-1} determines if the forget gate is successful. The prior memory ratio is determined in the equation6.10 and used in the equation 6.11. The output gate evaluates the output of a memory cell before being passed or not. This procedure is shown in equation6.13. Due to the three gates in LSTM, we can solve the vanishing and bursting gradient problems. In the LSTM-RNN architecture, the LSTM cell replaces the recurrent hidden layer.

6.8 Result and Discussion

In this section, the effects are precisely explained, along with an explanation of the DIO opt LSTM for ID. The effectiveness of the DIO optimization is evaluated based on

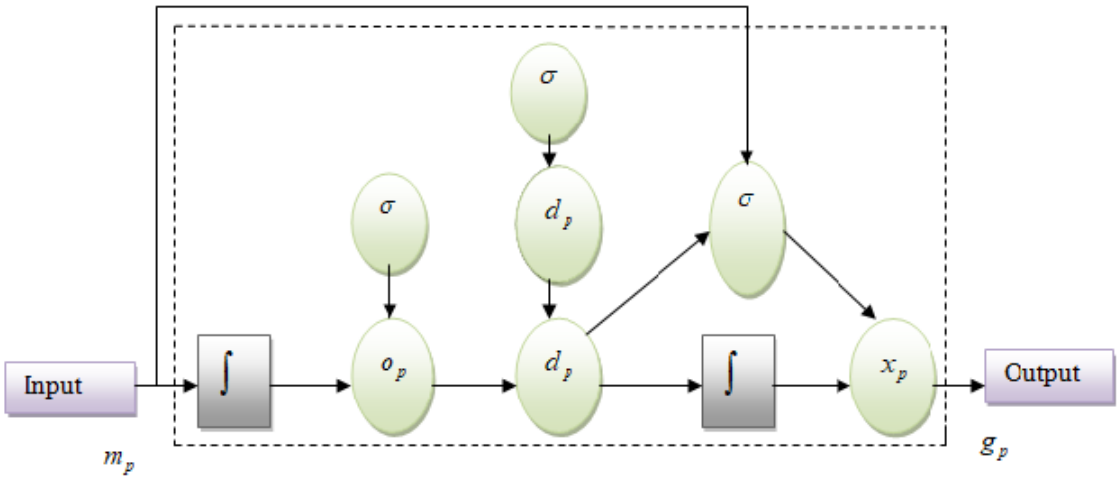


Fig 6.7: Long Short Term Memory Cell

performance measures.

6.8.1 Experimental Setup

Using 8GB of RAM and Windows 10 OS, the MATLAB tool executes ID in WSN with DIO opt LSTM. LSTM classifier utilizes the following key parameters to implement the model: Adam is the default optimizer, the loss is MSE and epochs of 100. Here, 100% of the dataset is used to train the model and the generated data conducts testing.

6.8.2 Dataset Description

The University of New Brunswick created the IDS 2018 Intrusion CSV file for DDoS data analysis. Based on the data, the dataset's data is separated into different files. This dataset consists of eighty columns, each representing a record in the IDS logging. The Protocol, Dest Port (Destination port), Flow Duration, Tot Fwd Pkts (Total forward packets), Tot Bwd Pkts (Total backward packets), and Label (Label) are the most significant columns in this dataset.

6.8.3 Performance Metrics

The outcomes of BHO-DCNN using their performance matrices like alive nodes, delay, throughput, and normalized energy are elaborated here with the dataset of IDS 2018 Intrusion CSVs for several rounds.

- **Delay** During transmission, the space between the termination and the source

increases, and the possibility of a loss of packets also increases. The delay of packets delivered to the termination is expressed by,

where T_Trans denotes the time at which data leaves from the transmitter $T_Reciever$, denotes the time at which the data reaches the termination, and A_nodes denotes the available nodes.

- **Normalized Energy** The proportion of total energy each WSN node uses to deliver every packet received to its intended location.
- **Throughput:** Throughput, a measure of a system's ability to handle and confirm transactions efficiently, is the number of transactions the system processes in a given amount of time. One important performance indicator is throughput, commonly measured in Transactions Per Second (TPS).

  $T = \frac{Sizeof Data}{Timeof Transmission}$

6.8.4 Performance Analysis

We extend the DIO opt LSTM results in this section by employing their performance metrics and the IDS 2018 Intrusion CSVs for many rounds of analysis.

6.8.4.1 Node 50 Analysis

Figure 6.8 displays the performance of the DIO opt LSTM approach in terms of the number of active nodes, delay, NE, and throughput. As seen in Figure 6.8, At round 1500, the active nodes accessible for the developed network are 42, 43, 44, and 45 for different epochs 40, 60, 80, and 100 respectively, which indicates the algorithm is more effective. When the EDs of the DIO opt LSTM method are studied for different epochs 40, 60, 80, and 100, the 1500th round is 1.65 ms, 1.16 ms, 0.83 ms, and 0.48 ms, respectively. The delay at epoch 40 has more value than at epoch 100, as shown in Figure 6.8, demonstrating the higher efficiency. After analyzing the NE of the DIO opt LSTM algorithm for various epochs, it is discovered that the NE for an epoch of 100 is higher than the epoch size of 40. The achieved NE for epochs 40, 60, 80, and 100 is 0.08J, 0.014J, 0.15J, and 0.16J, respectively, as shown in Figure 6.8. The throughput of the DIO opt LSTM algorithm is examined for epoch values of 40, 60, 80, and 100 and achieved 0.26bps, 0.26bps, 0.56bps, and 0.76bps respectively which is depicted in Figure 6.8. The throughput rate gradually declines from round one until the 300th one, where it reaches its maximum.

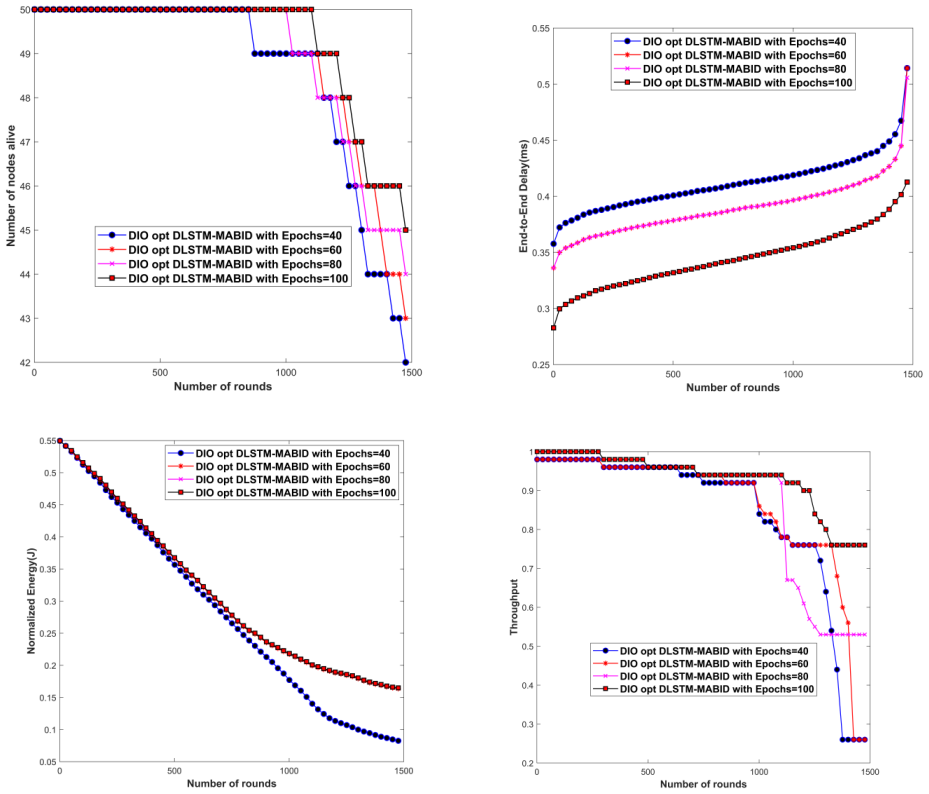


Fig 6.8: Analysis for 50 Nodes a) Number of Alive Node b) Delay c) Normalized Energy d) Throughput

6.8.4.2 Node 100 Analysis

Figure 6.9 displays the performance of the DIO opt LSTM approach in terms of the number of active nodes, delay, NE, and throughput. As seen in Figure 6.9, At around 1500, the active nodes accessible for the developed network are 58, 60, 82, and 87 for different epochs 40, 60, 80, and 100, respectively, which indicates the algorithm is more effective. When the EDs of the DIO opt LSTM method are studied for different epochs 40, 60, 80, and 100, the at the 1500^th round is 0.043ms, 0.095ms, 0.148ms, and 0.149ms, respectively. The value of delay at epoch 40 has more value than the delay at epoch 100, as shown in Figure 6.9 which demonstrates the higher efficiency. After analyzing the NE of the DIO opt LSTM algorithm for various epochs, it is discovered that the NE for an epoch of 100 is higher than the epoch size of 40. The achieved NE for epochs 40, 60, 80, and 100 is 0.25J, 0.25J, 0.25J, and 0.85J, respectively, at round 1500, as shown in Figure 6.9. The throughput of the DIO opt LSTM algorithm is examined for epoch values of 40, 60, 80, and 100 and achieved 0.25bps, 0.25bps, 0.25bps, and 0.85bps, respectively, which is depicted in Figure 6.9.

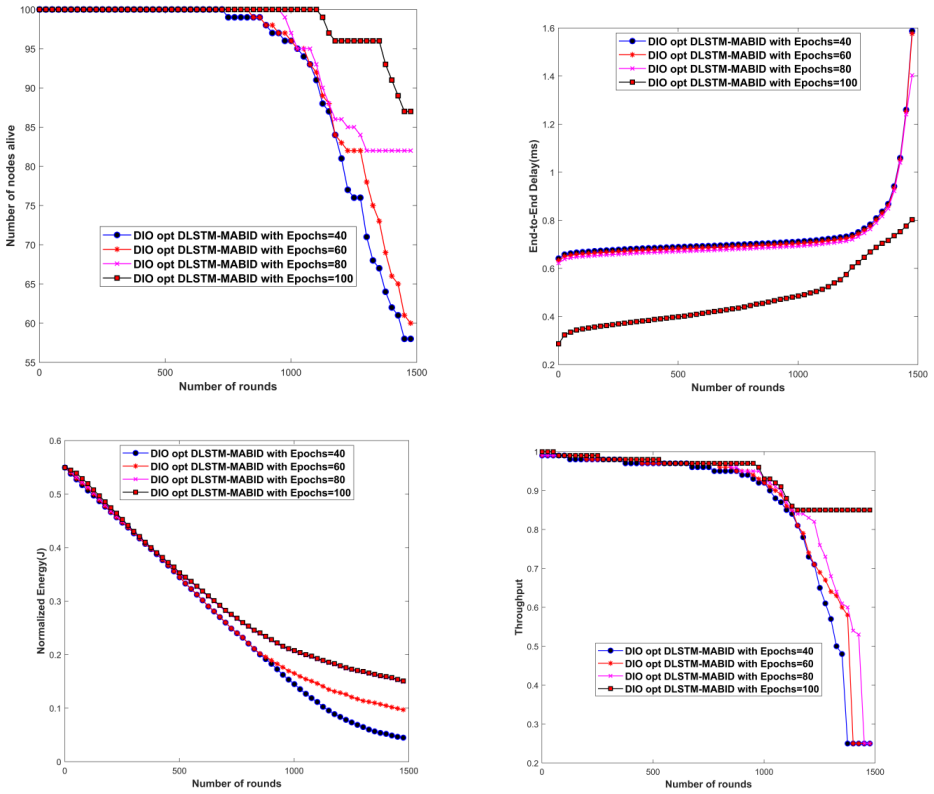


Fig 6.9: Analysis for 100 nodes a) Number of Alive Node b) Delay c) Normalized Energy d) Throughput

6.8.4.3 Node 200 Analysis

Figure 6.10 displays the performance of the DIO opt LSTM approach in terms of the number of active nodes, delay, NE, and throughput. As seen in Figure 6.10, At around 1500, the active nodes accessible for the developed network are 107, 122, 137, and 144 for different epochs 40, 60, 80, and 100, respectively, which indicates the algorithm is more effective. When the EDs of the DIO opt LSTM method are studied for different epochs 40, 60, 80, and 100, the at the 1500th round is 4.03ms, 1.34ms, 0.86ms, and 0.63ms, respectively. The delay at epoch 40 has more value than that at epoch 100, as shown in Figure 6.10, demonstrating higher efficiency. After analyzing the NE of the DIO opt LSTM algorithm for various epochs, it is discovered that the NE for an epoch of 100 is higher than the epoch size of 40. The achieved NE for epochs 40, 60, 80, and 100 is 0.063J, 0.073J, 0.146J, and 0.149J, respectively, at round 1500, as shown in Figure 6.10. The throughput of the DIO opt LSTM algorithm is examined for epoch values of 40, 60, 80, and 100 and achieved 0 0.25bps, 0.25bps, 0.25bps, and 0.89bps, respectively, which is depicted in Figure 6.10. The throughput rate gradually declines from round one until the 300th one, where it reaches its maximum.

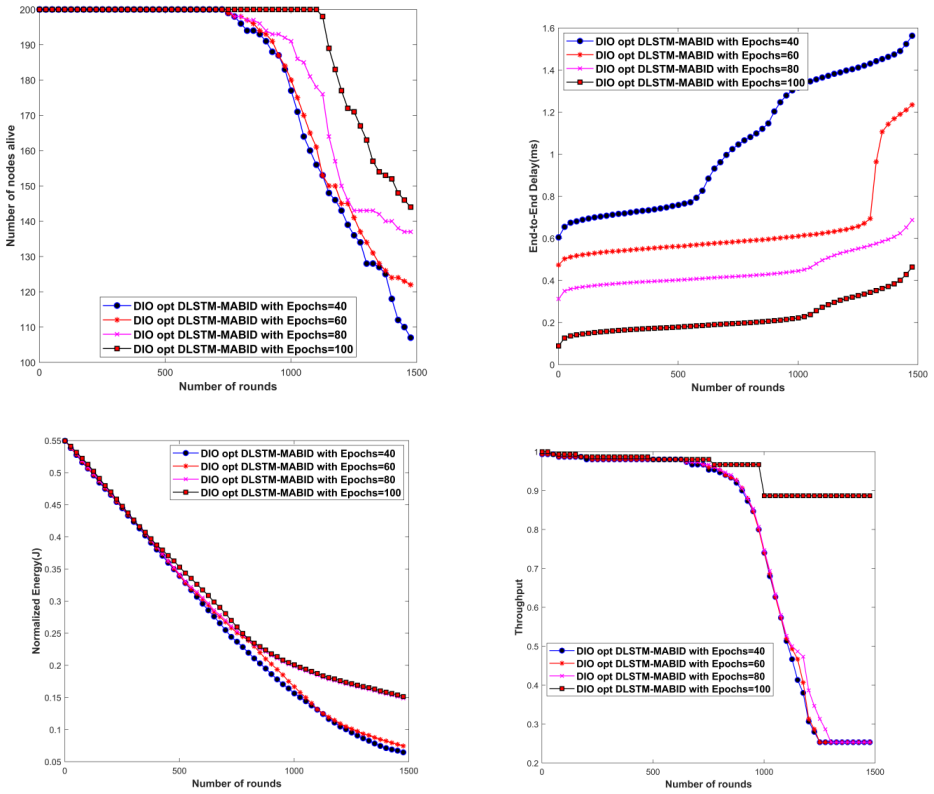


Fig 6.10: Analysis for 200 nodes a) Number of Alive Node b) Delay c) Normalized Energy d) Throughput

6.8.5 Comparative methods

PSO-WSN[80], ABC-MDC [81], BCO-WSN[82], GWO-WSN [83], GWO-CSA-DSAE [84], SL-GAT [71], A-DQN[66], ESO-MABID [85], SSA-MABID[86], and BHO-DCNN-WSN[72] are the methods used for comparison with the DIO opt LSTM model.

6.8.5.1 Analysis with 50 nodes

Figure 6.11 shows an analysis of the developed DIO opt LSTM methods with 50 nodes. The developed DIO opt LSTM contains 46 active nodes, demonstrating the success of the generated technique. The number of alive nodes in the existing methods such as PSO-WSN, ABC-MDC, BCO-WSN, GWO-WSN, GWO-CSA-DSAE, SL-GAT, A-DQN, ESO-MABID, SSA-MABID, and BHO-DCNN-WSN at 1500 rounds are 24, 28, 29, 29, 29, 30, 31, 31, 40, and 43 respectively. The proposed DIO opt LSTM obtains only 0.41 ms latency which is 2.05ms lower than PSO-WSN, 1.05ms lower than ABC-MDC, 0.93ms lower than BCO-WSN, 0.92ms lower than GWO-WSN, 0.92ms lower than GWO-CSA-

DSAE, 0.93ms lower than SL-GAT, 0.93ms lower than A-DQN, 0.93ms lower than ESO-MABID, 0.92ms lower than SSA-MABID, and 0.12ms lower than BHO-DCNN-WSN. The overall energy of the proposed DIO opt LSTM is 0.16 J, which is 0.128J, 0.126J, 0.125J, 0.123J, 0.123J, 0.120J, 0.117J, 0.117J, 0.117J, and 0.001J higher than the existing methods such as PSO-WSN, ABC-MDC, BCO-WSN, GWO-WSN, GWO-CSA-DSAE, SL-GAT, A-DQN, ESO-MABID, SSA-MABID, and BHO-DCNN-WSN. At 1500 rounds, 50 nodes accomplish the proposed DIO opt LSTM transfers data at the rate of 0.76bps, which is compared to the existing and the improvement of 0.49bps over PSO-WSN, 0.49bps over ABC-MDC, 0.49bps over BCO-WSN, 0.49bps over GWO-WSN, 0.49bps over GWO-CSA-DSAE, 0.50bps over ESO-MABID, 0.50bps over SL-GAT, 0.50bps over A-DQN, 0.50bps over SSA-MABID, and 0.49bps over BHO-DCNN-WSN. As a result, as compared to prior methods, the proposed DIO opt LSTM optimization procedure offers better results.

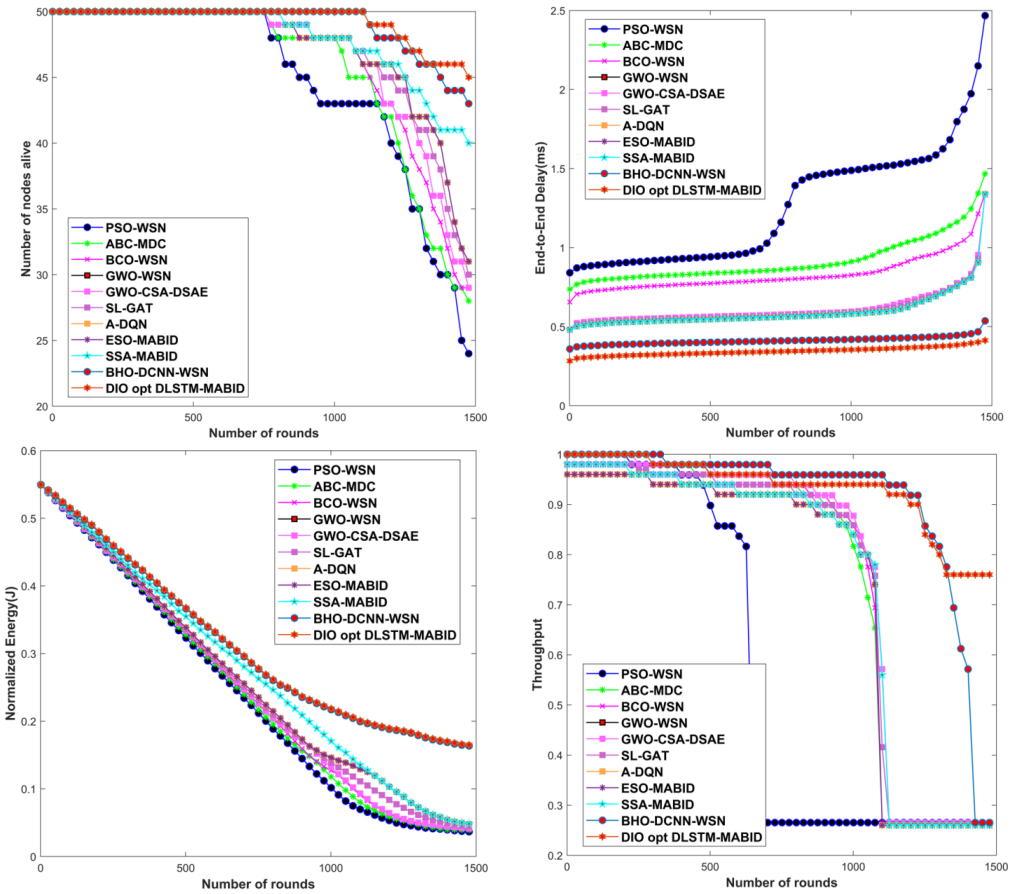


Fig 6.11: Comparative analysis for 50 nodes a) Number of Alive Node b) Delay c) Normalized Energy d) Throughput

6.8.5.2 Analysis with 100 nodes

Figure 6.12 shows an analysis of the developed DIO opt LSTM methods with 50 nodes. The developed DIO opt LSTM contains 87 active nodes, demonstrating the success of the generated technique. The number of alive nodes in the existing methods such as PSO-WSN, ABC-MDC, BCO-WSN, GWO-WSN, GWO-CSA-DSAE, SL-GAT, A-DQN, ESO-MABID, SSA-MABID, and BHO-DCNN-WSN at 1500 rounds are 44, 46, 48, 50, 50, 52, 54, 54, 55, and 82 respectively. The proposed DIO opt LSTM obtains only 0.8 ms latency which is 2.744ms lower than PSO-WSN, 1.026ms lower than ABC-MDC, 1.013ms lower than BCO-WSN, 0.999ms lower than GWO-WSN, 0.999ms lower than GWO-CSA-DSAE, 1.006ms lower than SL-GAT, 1.013ms lower than A-DQN, 1.013ms lower than ESO-MABID, 0.999ms lower than SSA-MABID, and 0.601ms lower than BHO-DCNN-WSN. The overall energy of the proposed DIO opt LSTM is 0.15J, which is 0.115J, 0.113J, 0.113J, 0.112J, 0.112J, 0.110J, 0.109J, 0.109J, 0.108J, and 0.002J higher than the existing methods such as PSO-WSN, ABC-MDC, BCO-WSN, GWO-WSN, GWO-CSA-DSAE, SL-GAT, A-DQN, ESO-MABID, SSA-MABID, and BHO-DCNN-WSN. At 1500 rounds, 50 nodes accomplish the proposed DIO opt LSTM transfers data at a rate of 0.85bps, which is compared to the existing and the improvement of 0.597bps over PSO-WSN, 0.597bps over ABC-MDC, 0.597bps over BCO-WSN, 0.597bps over GWO-WSN, 0.597bps over GWO-CSA-DSAE, 0.599bps over ESO-MABID, 0.6bps over SL-GAT, 0.6bps over A-DQN, 0.6bps over SSA-MABID, and 0.597bps over BHO-DCNN-WSN. As a result, as compared to prior methods, the proposed DIO opt LSTM optimization procedure offers better results.

6.8.5.3 Analysis with 200 nodes

Figure 6.13 shows an analysis of the developed DIO opt LSTM methods with 50 nodes. The developed DIO opt LSTM contains 146 active nodes, demonstrating the success of the generated technique. The number of alive nodes in the existing methods such as PSO-WSN, ABC-MDC, BCO-WSN, GWO-WSN, GWO-CSA-DSAE, SL-GAT, A-DQN, ESO-MABID, SSA-MABID, and BHO-DCNN-WSN at 1500 rounds are 73, 83, 84, 84, 84, 85, 85, 85, 85, and 137 respectively. The proposed DIO opt LSTM obtains only 0.46 ms latency which is 19.28ms lower than PSO-WSN, 12.05ms lower than ABC-MDC, 6.49ms lower than BCO-WSN, 6.15ms lower than GWO-WSN, 6.15ms lower than GWO-CSA-DSAE, 9.10ms lower than SL-GAT, 12.04ms lower than A-DQN, 12.04ms lower than ESO-MABID, 6.15ms lower than SSA-MABID, and 0.22ms lower than BHO-DCNN-WSN. The overall energy of the proposed DIO opt LSTM is 0.15J, which is 0.116J, 0.112J,

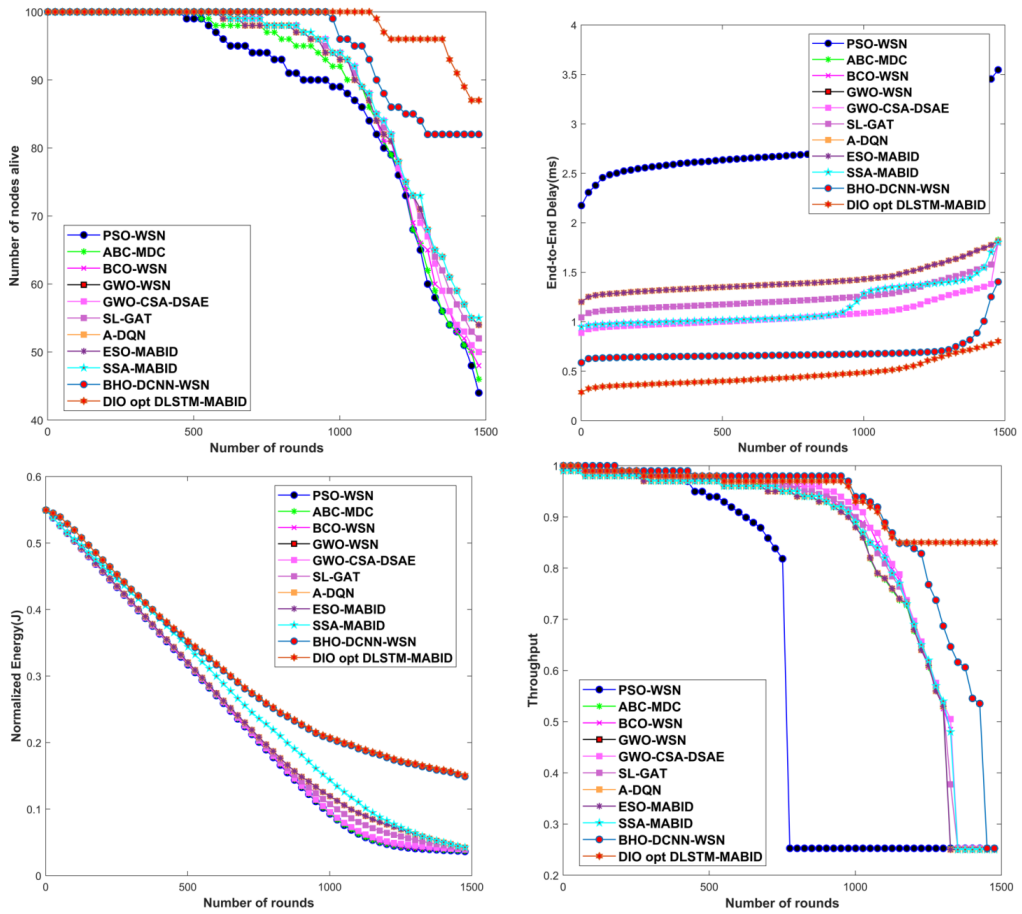


Fig 6.12: Comparative analysis for 100 nodes a) Number of Alive Node b) Delay c) Normalized Energy d) Throughput

0.105J, 0.104J, 0.104J, 0.108J, 0.111J, 0.111J, 0.105J, and 0.003J higher than the existing methods such as PSO-WSN, ABC-MDC, BCO-WSN, GWO-WSN, GWO-CSA-DSAE, SL-GAT, A-DQN, ESO-MABID, SSA-MABID, and BHO-DCNN-WSN. At 1500 rounds, 50 nodes accomplish the proposed DIO opt LSTM transfers data at a rate of 0.89bps, which is compared to the existing and the improvement of 0.726bps over PSO-WSN, 0.672bps over ABC-MDC, 0.632bps over BCO-WSN, GWO-WSN, GWO-CSA-DSAE, and ESO-MABID, 0.633bps over SL-GAT, A-DQN, and SSA-MABID, and 0.632bps over BHO-DCNN-WSN. As a result, as compared to prior methods, the proposed DIO opt LSTM optimization procedure offers better results.

6.8.5.4 Comparative discussion

The following describes each scheme utilized for protected ID in the network and demonstrates how the nodes' performance is determined across several rounds regarding the

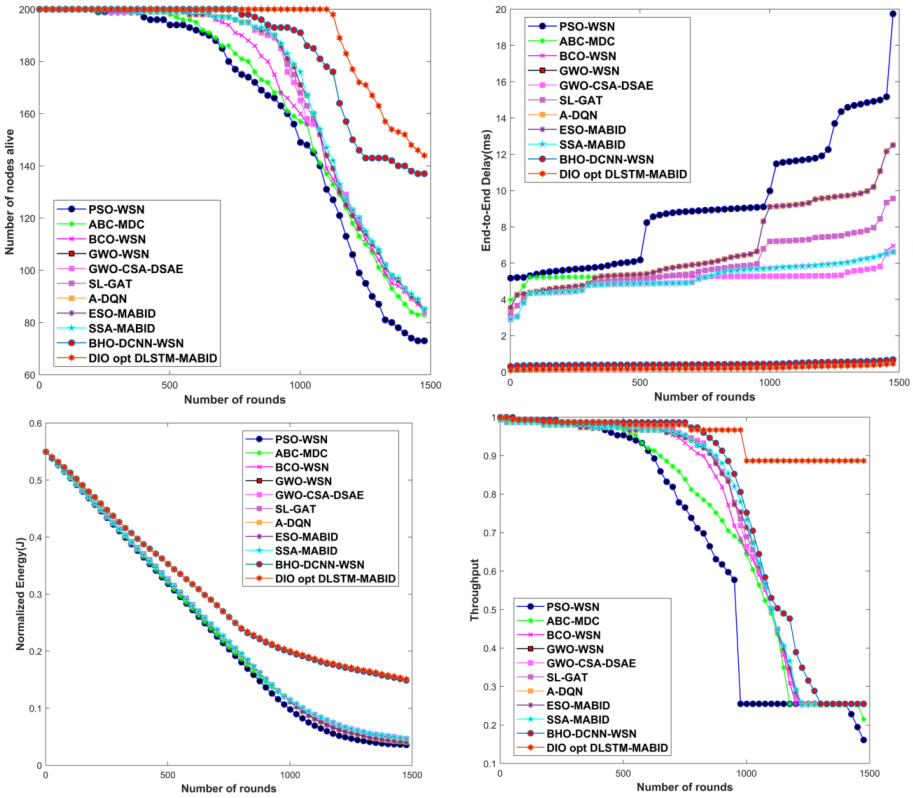


Fig 6.13: Comparative analysis for 200 nodes a) Number of Alive Node b) Delay c) Normalized Energy d) Throughput

metrics. Table 6.2 compares multiple methods with nodes 50, 100, and 200 at the round of 1500. The BHO-DCNN-WSN provides better intrusion detection and convergence, but the system's efficiency is low. The major limitation of the A-DQN is that the routers in the network have a limited number of potentially exploitable services compared to the hosts. SL-GAT utilizes more resources to provide better performance. Due to this, the implementation cost is increased.

The following describes each scheme utilized for protected ID in the network and demonstrates how the nodes' performance is determined across several rounds regarding the metrics. The BHO-DCNN-WSN provides better intrusion detection and convergence, but the system's efficiency is low. The major limitation of the A-DQN is that the routers in the network have a limited number of potentially exploitable services compared to the hosts. SL-GAT utilizes more resources to provide better performance. Due to this, the implementation cost is increased.

6.9 Ablation Analysis

Figure 6.14 illustrates the ablation study conducted by the developed DIO opt LSTM model, which discusses the model's performance by LSTM with optimization and LSTM without optimization. At 50 nodes, the model takes time to train without optimization 1.34ms and with optimization 0.41ms. Similarly, At 100 nodes, the LSTM without optimization takes 1.8 ms, while optimization takes 0.8 ms. Likewise, At 200 nodes, the LSTM without optimization takes 9.55 ms; with optimization, it takes 0.46 ms.

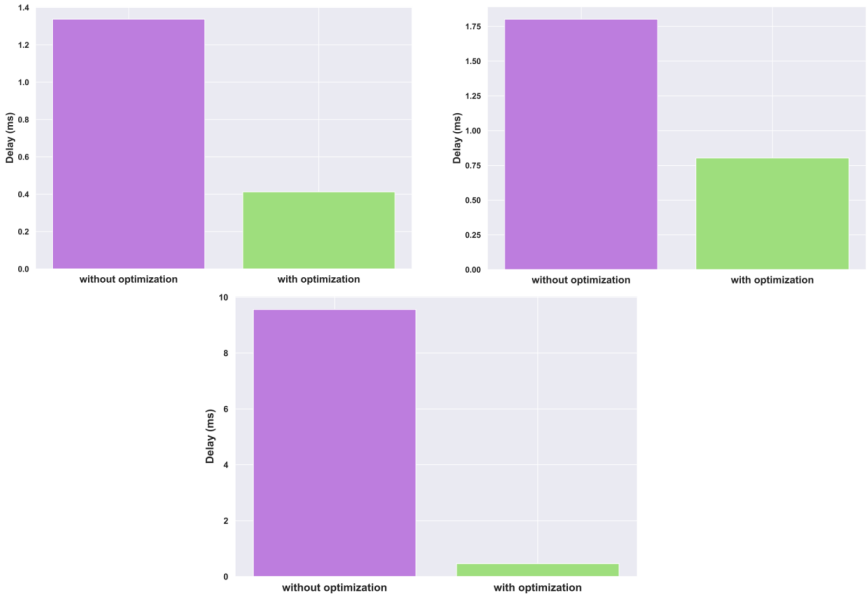


Fig 6.14: Ablation discussion of the proposed LSTM model

6.10 Summary

In this chapter, A self-configuring mobile agent-based intrusion detection system (MA-IDS) hybrid-optimized with Deep Long Short-Term Memory (Deep LSTM) networks offers an advanced solution for securing dynamic and complex networks like IoT and Wireless Sensor Networks (WSNs). This system features mobile agents that autonomously adapt to changing network conditions, improving their ability to detect intrusions in real-time. The integration of Deep LSTM networks enables the system to analyze and learn from time-series data, identifying complex patterns associated with both normal and malicious activities. This approach ensures adaptive security, scalability, and improved intrusion detection capabilities, making it highly suitable for modern network environments.

Table 6.1: Analysis of Literature Review

Method	Ref and year	Dataset	Advantage	Disadvantage	Achievement
Advanced mobile agent based ID system	2020 [53]	Global Datasets	Protect the connected medical equipment network	Costly in terms of computation and implementation.	Achieved accuracy of network 99.6 % and device-level intrusion detection 98.2 %
Mobile Agent for cluster based WSNs	2021 [54]	All the sensor node's dataset	Prevents the majority of attacks from occurring	Difficult to relocate mobile agents among all SNs	the detection efficiency is approximately 82 %
ML approaches	2020 [68]	NSL-KDD dataset	the attack detection rate enhanced and the false alarm rate reduced	Computational complexity is high	–
SL-GAT	2024 [71]	UNSW_NB15	provide better security to mobile agents	More resources are utilized by the system which causes system costs.	Achieved an accuracy of 98.5915%
BHO_DCNN	2024 [72]		Provides an enhanced rate of convergence	Need more hybrid optimization to improve the system performance	Normalized energy is 0.1622J, Throughput is 0.3125 %
A-DQN	2021 [66]	NSL-KDD and CICIDS2017	improves the scalability of the distributed network model	the router's potentially exploitable services are less than the hosts.	
An intelligent algorithm	2023 [73]		Provides guarantee for stability and security		Achieves low false alarm rates and high detection accuracy
Framework for DL based IDSes	2024 [74]	CIC-IDS2017 and CSE-CIC-IDS2018 Datasets	reduce the computational cost and time	Mostly this framework follows on labeled datasets	Achieve an average detection rate is more than 95 %
Ensembled Model	2023 [75]	CIDDoS2019	Reduced false alarm rates and also minimize computation and transmission costs	The solution may not be neutralizing completely	Achieves more accuracy and improved processing time.
Neuro Deep Learning	2022 [77]		Intrusion identification is improved and the rate of false caution and failure is diminished	unbalanced dataset	

Table 6.2: Comparative Discussion of the DIO opt LSTM

Method	**Alive Nodes**			**Delay(ms)**			**Normalized Energy**			**Throughput**		
	50	100	200	50	100	200	50	100	200	50	100	200
PSO-WSN	25	48	73	2.47	3.55	19.74	0.04	0.04	0.04	0.27	0.25	0.16
ABC-MDC	29	50	83	1.47	1.83	12.51	0.04	0.04	0.04	0.27	0.25	0.21
BCO-WSN	29	50	37	1.34	1.82	6.96	0.04	0.04	0.05	0.27	0.25	0.26
GWO-WSN	31	51	88	1.34	1.80	6.62	0.04	0.04	0.05	0.27	0.25	0.26
GWO-CSA-DSAE	31	51	88	1.34	1.80	6.62	0.04	0.04	0.05	0.26	0.25	0.26
SL-GAT	32	53	88	1.34	1.81	9.56	0.04	0.04	0.04	0.26	0.25	0.25
A-DQN	32	55	88	1.34	1.82	12.51	0.05	0.04	0.04	0.26	0.25	0.25
ESO-MABID	32	55	88	1.34	1.82	12.51	0.05	0.04	0.04	0.26	0.25	0.25
SSA-MABID	41	55	89	1.34	1.80	6.62	0.05	0.4	0.05	0.26	0.25	0.25
BHO-DCNN-WSN	44	82	137	0.54	1.40	0.69	0.16	0.15	0.15	0.27	0.25	0.26
DIO opt LSTM	46	87	146	0.41	0.80	0.46	0.16	0.15	0.15	0.76	0.85	0.89

Chapter 7

Enhancing Security with Dynamic Blow Filter and Blow Fish Security Protocol to Safeguard against Malicious Mobile Agents

This chapter introduces the use of a Dynamic Blow Filter and the Blow Fish Security Protocol to enhance the protection of mobile agent-based systems against malicious agents. The Dynamic Blow Filter provides real-time detection and blocking of threats by adapting to changing behavior patterns, while the Blow Fish Security Protocol offers strong encryption to secure data transmission. Together, these tools create a robust defense system, safeguarding against unauthorized access and ensuring the integrity of mobile agents in dynamic network environments.

Securing network agents is a complex and challenging task. A mobile agent is a newly performed innovative technique of cloud networking technology in the area of computer networking. The attack of malicious code execution in data communication between remote users is the primary problem affecting the agent. The design of agents presents numerous security challenges, which makes agent protection challenging. Nevertheless, recent studies have made progress in the area of agent code security. This paper presents an effective method for identifying the agent's issue and protecting it against malicious code execution.

A competent and practically applicable data model, the Dynamic Bloom filter (DBF), is the primary criterion for the agent code execution, providing evidence that the element is a function of set members. In the Java platform, the DBF creates the sub key to the arrays of agent codes that are broken up by the code breaker. The ASCII function then converts these sub keys into active keys, which are used in the encryption procedure. An elliptical curve key is used to generate the public key, which is required prior to the encryption process. The elliptical curve key (ECK) is a security measure that conceals the contents so that only authorized users can view the data in the form of a key or character. An asymmetric Blow fish algorithm has been used in the proposed protocol to perform encryption and decryption. The security of an agent from code execution is carried out by the proposed dynamic bloom filter and Blow fish algorithm. Data encryption and

decryption are performed using the 64-bit Blow fish encryption, which uses a key length that varies between 32-448 bits.

The goal of this paper is to improve agent security in terms of computational time when compared to the current AES and fragmentation protocol approach. The JADE platform, which is based on the Java programming language, is used exclusively to carry out the proposed work. The results demonstrate the effectiveness of the Blow fish algorithm and dynamic bloom filter in protecting the mobile agent against malicious code execution.

7.1 Introduction

In recent years, innovation and research on mobile agents have attained a higher response, which offers the implementation of new concepts in networking and distributed systems. The code in an agent is used to transmit data over the computer systems, which are hidden by some sort of programs written in common languages. The agent technology allows the application to send or receive codes to the remote user for execution. It can be retrieved dynamically from cloud sites or stored in data storage and output locally. Java is deemed to be a programming language for mobile codes and agent-based systems [87]. Based on this technology, creating distributed computing is possible by migrating programs and codes from one agent to another.

A mobile agent is a software-based agent that has the capability to transmit one host to another through its networking components. The capabilities of agents are to transmit data, codes, and some features like independent learning, which are effective for distributed systems. The major advantages of agents in networking are decreasing the network load, overcoming network latency, enclosing protocols, executing by self-determining and asynchronous, adapting dynamically in networking, being heterogeneous in nature, and fault tolerance. Finally, there is no killer app for mobile agents. Distributed applications are a challenge in technological advancement in the innovative world [88]. Enhancements in performance improvement to boost the efficiency aspects of agent-based technology have been implemented. The implemented technology offers different application features such as network flexibility, self-determination, capacity, etc.

However, the security and overhead issues are causing its global acceptance to decline. Operating systems use commercial trusted software applications for operation, and it is a very complex method of securing malicious data packets. A mobile agent is a program-based system that transfers from one system to another within the network environment. The intrusion links in the agents can block and attack certain collective agents[89]. The

defective codes are generally formed through network transmission in mobile agents. The agent infrastructure is classified into two modes of operation: communication and computation. The first is a home platform that creates the agents, and the second is a host platform used in agent code execution. The venomous codes can perform various network attacks on the agent when the agent is transmitted through the communication channel [90]. For malicious code execution, the agent's communication channel needs to be secured. In the migration of the agent from one line of transmission environment to another, the agent takes control of the execution environment, which makes it more sensitive to various attacks that are performed in the execution environment [91].

Generally, the security threat is categorized into three different sort of classes. They are data disclosure, DOS, and information exploitation. Two possible attacks influencing the mobile agent are intrusion agents and malevolent agencies. The agent intrusion is classified into two categories according to the target attacked. An affected agent can implement a DOS attack on other agents [6]. When transmitting large amounts of data over a network, malicious entities may target mobile agents currently residing on the network. The attacks that commonly occur are categorized into passive and active attacks. Contributions of this research are discussed below,

1. To protect mobile agents from malicious code execution attacks for enhancing the security of the agent.

2. To reduce the computational time required for agent security compared to existing AES and fragmentation protocol approaches.

3. To implement a dynamic bloom filter (DBF) for efficient agent code analysis.

4. To employ an elliptical curve key (ECK) and Blow Fish algorithm for robust encryption and decryption.

5. To assess the effectiveness of the proposed approach in terms of agent security and computational efficiency using the JADE platform.

The background process of this chapter is organized in the following sections. The recent research on this chapter is mentioned as related work and is explained in section 2, the methodology of the work is defined in section 3, the experimental research on the proposed work and result discussion is explained in section 4, and the final part of the chapter is the conclusion part explained in section 5.

7.2 Related Work

The mobile agent technique is a software-oriented agent system that offers to move between different locations. Developing an agent-based system, the system itself consists of a mobility framework [4]. The framework consists of all agent models, including the navigation model. The computational model of the agent consists of the calculative capabilities of a representative [92]. This model incorporates the data manipulation and primitives of thread control. The field of agents has numerous applications, methodologies, and thoughts, which has made it one of the more dynamic research areas in past years [5].

The agent model is a newer extension in distributed computing models. The software-oriented agent can send and retrieve data from a client to the web and circulate on the received hubs in the network [93]. The mobile agent further augments processing distributed systems in networking [94], [95]. It can be performed on those PCs to complete its task for the benefit of its user. The mobile agent state encourages it to work and, consequently, to transmit between one or more remote computing [96].

Security issues are becoming more noteworthy during unavoidable portable system registration. The data being utilized by adaptable distinctive sorts and altered vast scale conveyed applications associating over the wireless and wired system to transmit valuable administrations to venture, clients, settled, and flexible [97]. The security of an agent is a major issue for the agent, and research was carried out to implement security protocols[98]. In a distributed computing base, cloud clients often must depend on cloud benefit suppliers to transfer information into it [99].

It is still a concern for a remote computing client to gain trust in the security and consistent quality of remote computing administrations [100]. Cloud client private data in virtual machines situated on a respective domain makes it defenseless against numerous intrusions [101]. Recent research was carried out to create trust-based security infrastructure for remote computing that utilizes agents as the security protocol for obtaining helpful data from the computing system. By this method, the administrator or clients monitor information protection in the computing system [102].

All the system administrators provide protection to the hubs from intrusion or malicious host attacks. These users were programmed to steal nearby information or monitor the activity of the host [103], [104]. This research uses an obfuscation method to secure the network intrusion. The protection of code is not a part of the security instrument in networking, but a few functions define the quality that assumes a critical role in

source code security. In this research, mobile agent-based security has been implemented, bolsters adaptable and particular efforts to establish safety required by portable PCs and gadgets in appropriate frameworks [105]. This approach avoids privacy intrusion to operators from malicious attacks. The algorithmic function has been used to encrypt data from network intrusion, and a key generation technique is employed to develop a private key for each endpoint [106].

After the endpoint has transferred from the encrypted agent, they decode the information using the same algorithm for the decryption process, which also employs keys. For decryption, private keys are created from the initial key generation process, which, in turn, decrypts the information.

7.3 Proposed Dynamic Bloom Filter and Blow Fish Algorithm

The technique proposed in this paper includes Dynamic Bloom Filter and Blow Fish security protocol to avoid the threat of malicious code execution in the attacking environment. The agent code execution is formed during the data transfer between the mobile agents. Initially, the agents are created by the agent owner, and codes in the agent are broken into sets of blocks by a code breaker performed in the Java platform. The dynamic bloom filters the block to a set of arrays and generates the keys to the arrays, which are used for encryption. Each array creates 16 bits of a key, and each key is encrypted before it is sent to receiver nodes.

The encryption process needs a public and private key, the elliptic curve proposed in this paper is used for the generation of the keys. The data encryption and decryption are carried out by the Blow Fish algorithm, which uses a similar key for encryption and decryption. The decryption is reversible to the encryption process where it uses the same key for the decryption. By utilizing the private key from the elliptic curve, the data is decrypted by the receivers. After the encryption and decryption, the clone and checkpointing are carried out to validate the faulty nodes in the mobile agent.

The proposed method is experimented with in the JADE platform, and the output of the work is tabulated in terms of computational time and space complexity. Thus, the Dynamic Bloom Filter and Blow Fish algorithm attain higher efficiency in executing malicious nodes when compared with all the least possible methods in networking. Process Flow of Security Protocol against Malicious Mobile Agent is given in Figure 7.1.

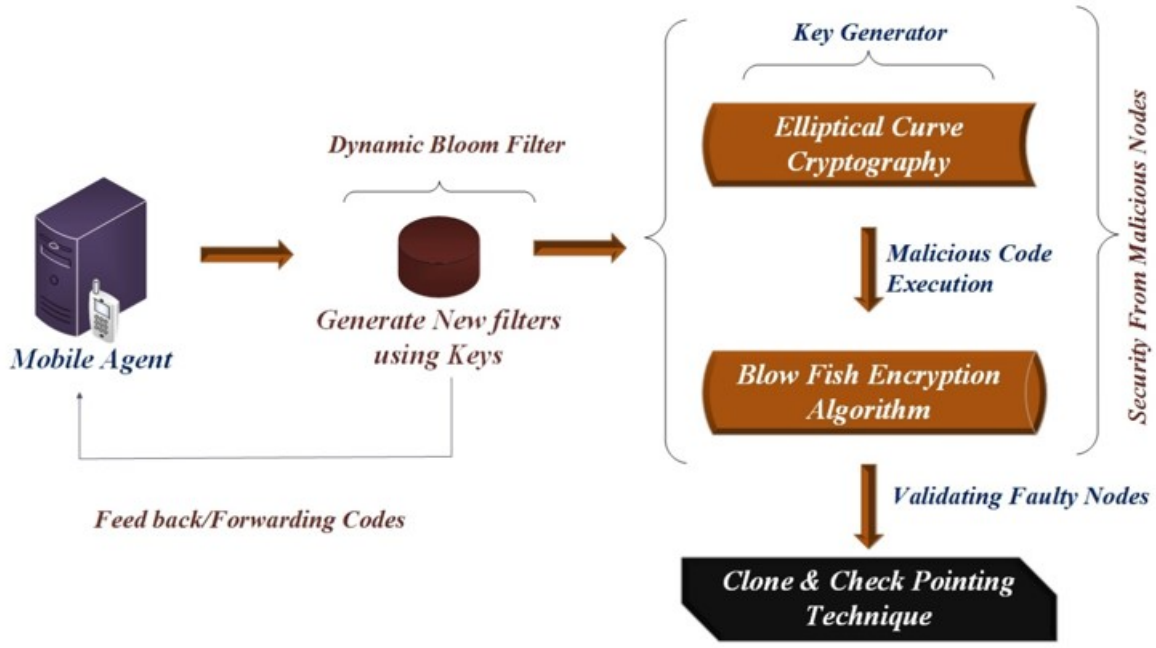


Fig 7.1: Process Flow of Security Protocol against Malicious Mobile Agent

7.3.1 Mobile Agent Technology

The mobile agent technique is a software abstraction that shifts across the network by categorizing the user to perform different tasks. The agents communicated in a specialized language since it is a computer system in varied environmental conditions designed to visualize the task and endpoints. Agents expand to various applications like networking, mobile computing, E-commerce, production, and research. The agent technology is an open system, so the code intrusions can easily attack it. This attacked host executes the codes in the agent, which often accesses the host resources. The accessed hosts in the agent attack the local agents to create the viruses and warm to the other agents.

The protection of an agent from the malicious host is harder than from the host protection. The host can access the agent codes and data if the agent executes the agents. Thus, the host makes manipulating and eliminating the agent codes easier. However, there is a possible way to secure the agents from attacks by limiting the private data transferring in the agents, secure routing, and use of algorithms. For the restriction of attack from the host, the agents have been encrypted/decrypted using various encryption algorithms. By employing these techniques, the agent's attacks can be restricted. Mobile agents are applicable in many areas, such as networking, data management, etc. It can reduce network traffic and consume fewer network resources for data computation.

7.3.1.1 Protecting the Mobile agent environment

Mobile agents are the specific class of agents that have the ability to execute the data transmitted from one host to another. The agent technology can reduce the traffic and overcome latencies in the network. Security is the factor regarding malicious code execution in the agent environment. The factors describing the securing agent environment are as follows.

7.3.1.2 Protecting the agent code

The agent environment is responsible for perfect execution and protection. The code execution is the main factor attacking the agent environment in the network security. In some conditions, the agent system and mobile code differ due to the execution of agent programs. The client and the code owner in the execution environment vary in functions. These were directed to problems regarding intrusion hosts and agent codes against different execution environments. The execution of the intrusion code is explained in the approaches below.

1. The problem is defined by refusing the agent code to the non-trusted host. This method determines whether the host is trust-based.

2. The problem is defined; measures have been taken, and the mobile agent feeds the code to open source so everyone can view it. The host execution is only undergone by trustworthy parties.

3. The next one is using specialized hardware to integrate agent codes. In this approach, every host is specialized in response hardware, which is a basic need for operation.

4. The final approach is securing the agent codes with controlled environments by employing proposed protocols in the agents before and after the execution process.

7.3.1.3 Security measures for Agents

Recent research carried out on agents has been implemented, and many of them are still in development. The security issues are mainly caused by malicious code execution in the networking. Still, somehow, the restriction of malicious code execution is carried out by utilizing the encryption techniques in the code execution. The solution can be established using adequate security protocols, and communication is protected on the same level.

7.3.1.4 Agent key generation

The key generation in the agent is to transmit encrypted code of information in the network. Elliptical curve-based encryption algorithms and techniques carry out the encryption. The same Blow Fish algorithms can decrypt these codes of information. Using this method, an agent's private information is secured and cannot be disclosed by everyone. This information is encrypted and only viewed by the environment when it is decrypted or conditions to be met. However, the data do not decrypt the information and no longer exist on the platforms.

7.3.1.5 Encrypted Data

Asymmetric elliptic curve technique is a public-key generating method used in the mobile agents to encrypt and decrypt the data by the owner to the network. The Blow Fish algorithm is proposed to encrypt the data in agents, and then it is added to the method, which results in small terms of size. By nature, the agent is restricted from accessing its own encrypted data until the trusted host has the decryption key.

7.4 Dynamic Bloom Filter (DBF)

The Dynamic Bloom Filter is a test function that validates the element in the member set. Elements are largely added to the set and considered to have a high probability of positives being false. The DBF can control the probability at a very low level by increasing the quantity as it raises the set level. The DBF has more stability than the bloom filter because of the unusual reconstruction and by directing the dynamic sets without an upper bound on primary sets. Based on the algorithmic function, the key number and the false positive probability can evaluate the filter sizes and the number of bits stored in each key.

The Dynamic Bloom Filter is constructed based on the s×m matrix with the standard bloom filter. The system parameters have filtered size 's', and hash function 'H' and many bloom filter slices are adjusted and allowed to grow DBF dynamically. This concept is purely established on the current BF. The element inserted in the bloom filter by each constituent is recorded. The Dynamic Bloom Filter does not come with the advantages of a bloom filter but also has the bloom filter features when dealing with dynamic sets. The bloom filters false probability increases aggressively with an increase in the set cardinality. The slow increase of the bloom filter is due to the expansion of capacity in an additional manner by the cardinality of dynamic sets.

7.5 Algorithmic function

The Dynamic Bloom Filter splits the agent code into 'm' bits of blocks and is set to '0'. The block element is hashed with the hash function 'H,' where the element is in an 's' array with uniform random distribution. The hash function 'H' is generally smaller than 'm' and is proportional to the number of elements added. The calculated filter false-positive rate decides the accuracies of k and the proportionality of m. For adding an element, it is fed to each of the k hash to form H arrays. The bits are set to 1 at these positions. To testify the element in the set fed in the hash function to form the 'H' array position. If the bits are not in position 0, it shows the element is not in the set and changes all the bits and sets to 1. If all the bits are 1 or by chance, it is set, which results in a false positive. The need for modeling different independent hash function is restricted for higher hash function 'H'. For getting a better hash function with larger output, if there is any relation between bits and hash. This type progresses several hashing by separating the outputs into multiple array bits. For larger 's' or 'H', the ability of the hash function is flexible with negotiation with an increase of false-positive rate.

7.5.1 Key Generation

The agent code breaks down into blocks; a set of byte arrays separates each block. Each byte array is separated into 's' bit's vectors. The block is used for the creation of a key, which is split into 16-byte arrays by using the hash function H = $h_1, h_2 \ldots . h_k$ with the range of $(1 \ldots . s)$. The generation of the key is explained below.

Let $A = (a_1, a_2 \ldots . a_n)$ of 'n ' number of elements, which are taken as subkeys. For generating the 16-bit key, the elements are initially taken in the range of (1-16). The hash function for the subkey formation for each block element is calculated using the following:

$$\left| \boldsymbol{h}_1(a) = P_1 \right| \left| \boldsymbol{h}_2(a) = P_2 \right| \ldots \left| \boldsymbol{h}_k(a) = P_k \right|$$

Initially, Bloom Filter splits the agent code into equal byte blocks of 16 bytes. Each byte block is characterized by a single-bit special character assembled to form a 16-bit key. Thus, the assembled subkey is like this,

$$Key = 5bc\#2r\$fe \;\&\; 1n@\Sigma 6\%$$

Table 7.1: Ordering Byte of Codes

Byte of Codes	Key Character	ASCII Value	Mod4
Byte 1	5	53	1
Byte 2	b	98	2
Byte 3	c	99	3
Byte 4	#	35	3
Byte 5	2	50	2
Byte 6	r	114	2
Byte 7	$	36	0
Byte 8	f	102	2
Byte 9	e	101	1
Byte 10	&	38	2
Byte 11	1	49	1
Byte 12	n	110	2
Byte 13	@	64	0
Byte 14	Σ	228	0
Byte 15	6	54	2
Byte 16	%	37	1

7.5.1.1 The probability of False Positives

The hash function chooses a position of the byte array at probability. Let ' s ' be the number of array bits in the byte block, set to 1. The hash function for the element insertion is defined by

$$1 - \frac{1}{S} \tag{1}$$

If the number of the hash function is H, The byte array probability of a bit is not set, and the hash function is,

$$\left(1 - \frac{1}{S}\right)^{H} \tag{2}$$

If the n number of elements is inserted and probability is set to 0 , then $\left(1 - \frac{1}{S}\right)^{Hn}$ and if it is 1, the $1 - \left(1 - \frac{1}{S}\right)^{Hn}$

For the element test, it is set; subsequently, the position of the array is measured by the hash function as 1 with the probability. Thus, the algorithm is made to zero error to confirm the element is in the set, which is defined as

$$\left(1-\left[1-\frac{1}{S}\right]^{Hn}\right)^{H} \approx \left(1-e^{-Hn/m}\right)^{H} \tag{3}$$

As a result, the probability of each bit is set, by the close comparative the probability of false positives decreases with an increase of ' s ' and ' n '.

By adding all the n items to the bloom filter, the fraction q of m bits are set to 0. By testing the element that is not in the set for the hash function k of array position and the bit probability is set to 1, then it is $1-x$. The bit probability of hash function ' H ' is set to 1 and $(1-x)^H$. The predicted outcome ' x ' is taken as probability, and the position of the array is not used in the hash function ' H ' for the n items, which can be derived by

$$E[x] = \left(1-\frac{1}{s}\right)^{Hn} \tag{4}$$

Thus, the probability of false positives is derived by

$$(1-E(x))^H = \left(1-\left[1-\frac{1}{s}\right]^{Hn}\right)^{H} \tag{5}$$

7.5.1.2 Optimal number of Hash function

$$H = \frac{s}{n}\ln 2 \tag{6}$$

Thus, the number of bits ' s ', inserted elements n and desired false-positive probability ' k ' can be measured by substituting the optimal hash function ' H ' in the given equation 7 below.

$$k = \left(1-e^{-(s/n\ln 2)\frac{n}{s}}\right)^{\frac{s}{n}\ln 2} \tag{7}$$

$$\ln k = -\frac{s}{n}(\ln 2)^2 \tag{8}$$

$$s = -\frac{n\ln k}{(\ln 2)^2} \tag{9}$$

7.6 Elliptical Curve Key-based Blow Fish Algorithm

In the proposed algorithm, the key is generated by the elliptical curve, which is a key generation process in encryption because the length of the key is the factor for the strength of encryption. The Elliptical Curve Key(ECK) can offer 160-bit key security at the same higher or lesser length as the 448-bit key generated by the Blow Fish algorithm. The key generation by an elliptical curve can provide less computation time to the Blow Fish algorithm.

7.6.1 Elliptical Curve Key (ECK)

The Elliptical Curve Key(ECK) is based on the elliptical curve property in the analytical structure. The Elliptical Curve Key(ECK) allows a user to select a numerical number as a private key, which is used to select a point on the curve. This paper uses the Elliptical Curve Key(ECK) based Blow Fish algorithm to create a public key for data encryption or decryption. The advantage of the elliptical curve is that the key value for encryption is small; storage can be reduced, limiting transmission needs. This technique is the same as the security used by the Advanced Encryption Standard (AES) and Fragmentation system, which uses the larger part of encryption modulus and key generation. The elliptic curves are used in the public-key cryptosystem. This key generation provides better protection against malicious intrusions than the traditional cryptosystem for the key size given. The fact is that security can be created by limiting the key size to keep the security. The reason for utilizing the cryptosystem is if an additional framework in the elliptical curve can crash to generate a new cryptosystem with advancing features. Selecting a base point or generator point in the Elliptic Curve (ECC) is the prime step for its security. The base point of ECC is given below,

$$E : c^2 = x^3 + px + q \tag{10}$$

Where, p and q are integers, by satisfying the condition is $4p^3 + 27q \neq 0 \bmod r$; and here ' r ' is the prime number. It includes a point called an infinity point.

7.6.2 Elliptical Curve Key Generation

The advantage of the Elliptical Curve Key is that the user can individually figure out the secret key from its private and public keys. The elliptical curve is an efficient method for executing keys to log messages securely. By utilizing the following equation to generate

the public key,

$$K = \eta^* P \tag{11}$$

Where ' η ' is the private key chosen as random in the range of 1 to $n-1$, P denotes the point curve on Elliptical Curve Key(Elliptical Curve Key(ECK)), and public key for the generation is denoted by K.

7.6.3 Blow Fish Algorithm

The Blow Fish algorithm is based on symmetric key algorithms that utilize the same key for encryption and decryption. A block cipher divides the information into a set of lengths during the encryption/decryption process. Blow Fish consists of a 64-bit block length of messages. The encryption algorithm is used for data protection. The encryption algorithm requires authentication and guarantees that the incoming messages will find their sources. Some algorithms restrict the outcomes from accessing the message content from reading. The Blow Fish encryption needs only 5Kb of memory for the processing. Encrypting a 64-bit message in a 32-bit processor requires approximately 12 clock cycles. Implementing longer messages requires a vast computation time in a linear manner. For example, for encrypting a 128-bit message, the calculation of the Blow Fish clock doubles. The Blow Fish can encrypt messages with keys up to 448 bits in length. The Blow Fish is a public domain cipher subjected to a particular encryption process that cannot be broken/crashed. The Blow Fish encryption is a secure encryption used in products like Splash ID that works on various types of processors in mobile phones and computers.

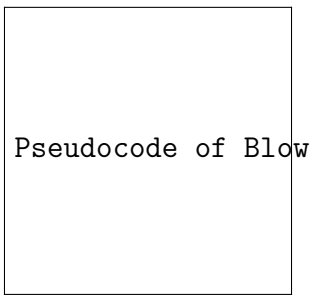


Fig 7.2: Pseudo code of Blow Fish Algorithm.

7.6.4 Blow Fish Encryption

The Blow Fish algorithm has the function to carry out iteration to 16 times of network. The iteration of the algorithm consists of various key modifications and data-based alternatives. The XOR operation is carried out with 32-bit data, and the encryption

algorithm is processed by the public key generated by the elliptical curve process. Initialize the Bloom filter, load the code to hash, and break the code into bit array blocks. Structure the blocks into arrays using ECC's private and public keys. Encryption creates 16 bits of keys, which are collapsed. These encrypted keys are sent to the receiver so the malicious codes cannot be executed.

7.6.5 Data Decryption

The decryption is the same as the encryption process but reversible. The user can decrypt the code by reversing the encryption process. The decryption starts with retrieving the home user's private key and decodes the mobile agent's encrypted file.

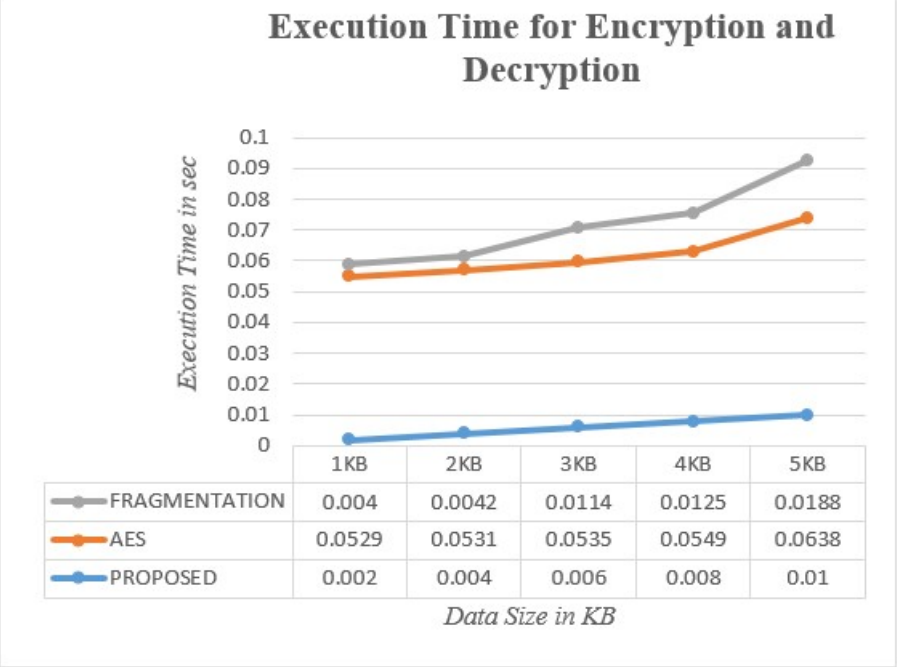
7.7 Experimentation and Result Discussion

The proposed approach, implemented and executed in the JADE platform, provides better optimization than the existing techniques. JADE is a Java framework implemented to execute important agent-based communication and networking tasks. JADE has many features, including the development of distributed systems. JADE can perform multi-agent implementation using graphical techniques, incorporating phases like debugging and deployment. The JADE operation is controlled wirelessly by using a remote GUI. The programming language used in JADE implementation is based on Java, with the least system requirements for running Java or a Java Development Kit (JDK) environment.

The user uses the mobile agent class to create JADE agents. JADE can implement distributed agents and storage for host of each running agent. The agent platform has various functional properties like tools for debugging, code mobility, agent contents, parallel agent execution, and so forth. In the proposed method, the mobile agent code is written in the Java platform by Executable. Class file. The extracted code file is split into different byte arrays by code breaker. Class in the Java platform. After breaking the code into an array, the bloom filter generates the sub key into a byte array. These sub-keys are rearranged using the ASCII function to generate 128-bit serial keys. The elliptical curve is used in the process to generate public by the keys generated from the bloom filter before the Blow Fish algorithm encrypts it, and the key generation process is repeated in the decryption process in the JADE platform.

In this chapter, the mobile agent code is broken up by a Dynamic Bloom Filter using the hash function, which splits the code into 16 bytes of array blocks, and each block creates 8 bits of the key. The keys are initially taken randomly, and by using ASCII

value, the keys are generated to the mix of characters. These keys are mod-operated and categorized into each array, and the key is collapsed. The elliptical curve splits up the key into a public and private key. After the key generation, these keys are encrypted using the Blow Fish algorithm. For the decryption process, the bloom filter was also employed to reassemble the key and decode the file for mobile users. Execution Time for Encryption and Decryption is shown in Figure 7.3, and Execution Time for Encryption is illustrated in Figure 7.4.



	1KB	2KB	3KB	4KB	5KB
FRAGMENTATION	0.004	0.0042	0.0114	0.0125	0.0188
AES	0.0529	0.0531	0.0535	0.0549	0.0638
PROPOSED	0.002	0.004	0.006	0.008	0.01

Fig 7.3: Execution Time for Encryption and Decryption

In Figure 7.5, the execution time of existing protocols with the proposed approach has been taken for comparison. The Blow Fish execution time for encryption of protocols is taken from the class file of 1Kb to 5Kb data size. From the time execution, the proposed technique has a lower execution time and is optimal than the existing protocols.

A space complexity comparison was made between the data used for encryption in various routing protocols and the proposed protocol. The memory utilized by the algorithm for data encryption is denoted as space complexity. In Figure 7.5, the graph is plotted to compare the existing protocol with the proposed technique utilizing the memory for encrypting the data, and the result shows that the method in this chapter uses less memory consumption than the existing protocols.

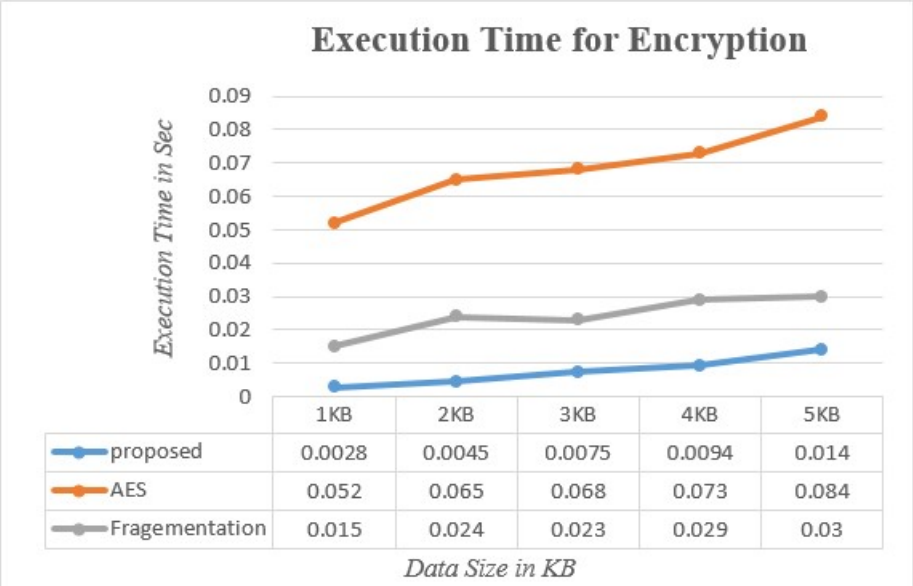


Fig 7.4: Execution Time for Encryption.

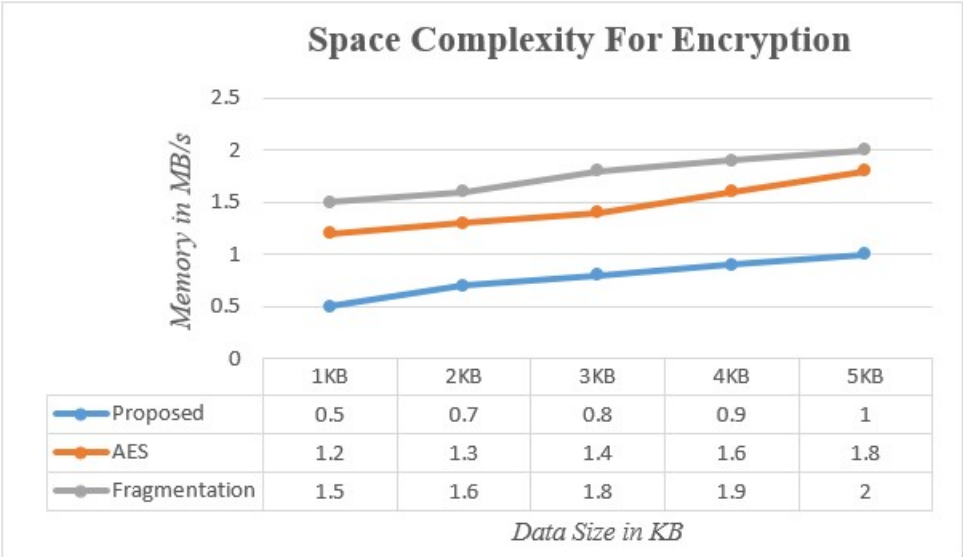


Fig 7.5: Space Complexity for Encryption.

7.8 Summary

In this Chapter, an efficient protection technique is developed to detect and secure mobile agents from malicious code execution. This proposed approach can effectively ensure the safety and security of clients' private information and take control measures of client data by controlling and shielding the information using security agents by the client side itself. The malicious hosts generally occur during the transmission of data from the mobile agent to the network. An efficient encryption and decryption protocol enhances the code execution. This mobile agent-based system helps in developing a trust-based system

for communication and transmission over a secure and reliable connection over the network. The security techniques discussed in this proposed method will be more efficient when compared with recent security measures in this field. The motive of this approach has been compared with the proposed security approach with recently established security protocols like AES and Fragmentation algorithms. Thus, the Dynamic Blow Fish and Blow Fish protocols are compared with the recent approaches AES and Fragmentation protocol regarding computational time for encryption and decryption. Moreover, the result clearly shows that the proposed approach is more efficient than the existing method, and it suits well in the place of securing the mobile agent from malicious code execution.

Chapter 8

Conclusion and Future Scope

Based on the methods discussed in your thesis report, here are the conclusions and future scope:

8.1 Conclusion

1. Comprehensive Understanding of Mobile Agent Security: The literature review and research methodologies have provided a deep insight into mobile agent security in wireless sensor networks (WSNs). This includes understanding their capabilities, inherent challenges like security during migration, dynamic adaptation, and resource utilization efficiency.

2. Identification of Security Gaps and Challenges: Significant gaps in existing research were identified, particularly the need for robust security mechanisms, adaptive communication protocols, and efficient resource management.

3. Innovative Solutions Developed: Various innovative solutions were proposed across different methodologies:

    - Ensemble Model for Attack Classification: A robust ensemble model was developed for automated attack classification in large-scale WSNs using mobile agent-based intrusion detection systems. This model demonstrated high accuracy and reliability in identifying attacks.

    - Border-Hunting Optimization with Deep CNN: The Border-Hunting Optimization integrated with Deep Convolutional Neural Networks (CNNs) showed enhanced intrusion detection capabilities by optimizing cluster head selection and improving detection accuracy across different network scenarios.

    - Dunnock Ibis Optimization LSTM for Self-Configuring Systems: The DIO-Opt LSTM model enhanced the security and reliability of mobile-agent-based WSNs by effectively detecting malicious nodes and optimizing cluster selection, thereby reducing communication costs and improving system efficiency.

- Efficient Protection Techniques: Developed techniques focused on securing mobile agents from malicious code execution during data transmission, emphasizing encryption protocols and trust-based communication frameworks.

8.2 Future Scope

1. Enhanced Security Protocols: Future research can focus on developing even more robust and adaptive security protocols that can dynamically respond to evolving threats in real time.
2. Integration with Artificial Intelligence and Machine Learning: Further integration of advanced AI and machine learning techniques could enhance anomaly detection and improve the responsiveness of intrusion detection systems.
3. Performance Optimization: Continued efforts in optimizing resource management, reducing latency, and improving energy efficiency will be crucial for deploying mobile agent systems at scale.
4. Real-World Applications: Extend research to validate proposed methodologies in diverse real-world applications such as agricultural monitoring, healthcare systems, and industrial IoT environments.
5. Comparison with Standard Protocols: Continuously benchmarking against existing security standards like Advanced Encryption Standard(AES) and fragmentation protocols to validate and improve upon the proposed methodologies.

By addressing these aspects, future research can contribute significantly to advancing the deployment and security of mobile agent-based intrusion detection systems in wireless sensor networks.

Bibliography

[1] Vasiu et al. Mobile agents in wireless devices. *Computer,IEEE*, 37(2):104–105, 2004.

[2] Khan et al. The state-of-the-art wireless body area sensor networks: A survey. *International Journal of Distributed Sensor Networks*, 14(4):1550147718768994, 2018.

[3] Priyanka et al. Security issues in mobile agents. *International Journal of Computer Applications*, 11, 12 2010.

[4] Srivastava et al. Trust analysis of execution platform for self protected mobile code. In *2015 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, pages 1904–1909. IEEE, 2015.

[5] Wei et al. Security analysis of an enhanced mobile agent device for RFID privacy protection. *IETE Technical Review*, 32(3):183–187, 2015.

[6] Zhang et al. A novel multi-agent-based collaborative virtual manufacturing environment integrated with edge computing technique. *Energies, MDPI*, 12(14):2815, 2019.

[7] Nimbalkar et al. Mobile agent: Load balanced process migration in linux environments. In *2015 International Conference on Computing Communication Control and Automation*, pages 561–564. IEEE, 2015.

[8] Dada et al. Performance evaluation of aglets and jade mobile agent using encryption and decryption time. *Radioelectronics and Computer Science, Kharkov National University of Radio Electronics*, 35(4):16–20, 2010.

[9] Bhawana Sharma and Arun Gangwar. Mobile agents: Objective, platforms and architecture (the cutting edge of wireless technology). *International Journal of Engineering and Management Research (IJEMR)*, 2(2):10–13, 2012.

[10] Shoham et al. *Multiagent systems: Algorithmic, game-theoretic, and logical foundations*. Cambridge University Press, 2008.

[11] Bieszczad et al. Mobile agents for network management. *IEEE Communications Surveys*, 1(1):2–9, 1998.

[12] Grover et al. Agent based dynamic load balancing in cloud computing. In *2013 International Conference on Human Computer Interactions (ICHCI)*, pages 1–6, 2013.

[13] Lin et al. Balancing energy consumption with mobile agents in wireless sensor networks. *Future Generation Computer Systems, Elsevier*, 28(2):446–456, 2012.

[14] Aiello et al. Using mobile agents as enabling technology for wireless sensor networks.

In *2008 Second International Conference on Sensor Technologies and Applications (sensorcomm 2008)*, pages 549–554. IEEE, 2008.

[15] Zhang et al. Mobile agent routing algorithm in wireless sensor networks. In *Advances in Computer Science and Information Engineering: Volume 2*, pages 105–113. Springer, 2012.

[16] El Fissaoui et al. A survey on mobile agent itinerary planning for information fusion in wireless sensor networks. *Archives of Computational Methods in Engineering,Springer*, 28:1323–1334, 2021.

[17] S. Venkatesan et al. Advanced mobile agent security models for code integrity and malicious availability check. *Journal of Network and Computer Applications, Elsevier*, 33(6):661–671, 2010.

[18] Baumann et al. Mole: A mobile agent system. *Software: Practice and Experience, Wiley Online Library*, 32(6):575–603, 2002.

[19] Ssekibuule and Richard. Mobile agent security against malicious platforms. *Cybernetics and Systems: An International Journal, Taylor & Francis*, 41(7):522–534, 2010.

[20] Bellavista et al. A survey of context data distribution for mobile ubiquitous systems. *ACM computing surveys (CSUR)*, 44(4):1–45, 2012.

[21] Mottola et al. Programming wireless sensor networks: Fundamental concepts and state of the art. *ACM Computing Surveys (CSUR)*, 43(3):1–51, 2011.

[22] La Polla et al. A survey on security for mobile devices. *IEEE communications surveys & tutorials*, 15(1):446–471, 2012.

[23] Conti et al. Emergent properties: detection of the node-capture attack in mobile wireless sensor networks. In *Proceedings of the first ACM conference on Wireless network security*, pages 214–219, 2008.

[24] Almomani et al. WSN-DS: A dataset for intrusion detection systems in wireless sensor networks. *Journal of Sensors, Hindawi*, 2016, 2016.

[25] Odesile et al. Distributed intrusion detection using mobile agents in wireless body area networks. In *2017 Seventh International Conference on Emerging Security Technologies (EST)*, pages 144–149. IEEE, 2017.

[26] Kulik et al. Negotiation-based protocols for disseminating information in wireless sensor networks. *Wireless networks, Springer*, 8(2):169–185, 2002.

[27] Amri and Mohammad Hossein. *Secure-Spin with Hashing to Support Mobility and Security in Wireless Sensor Network*. PhD thesis, Universiti Teknologi Malaysia, 2013.

[28] Jitender Grover et al. Reliable spin in wireless sensor network. In *Proceedings of 3rd International Conference on Reliability, Infocom Technologies and Optimization*,

pages 1–6. IEEE, 2014.

[29] Tseng et al. Location tracking in a wireless sensor network by mobile agents and its data fusion strategies. *The Computer Journal,OUP*, 47(4):448–460, 2004.

[30] Yoneki et al. A survey of wireless sensor network technologies. *UCAM-CL-TR-646*, 2005.

[31] Chen et al. Mobile agent based wireless sensor networks. *J. Comput.*, 1(1):14–21, 2006.

[32] Shen et al. Mobile agent based middleware using publish/subscribe mechanism in wireless sensor networks. In *2009 International Conference on Communication Software and Networks*, pages 111–115. IEEE, 2009.

[33] Abiona et al. Mobile agent based authentication for wireless network security. In *2008 International Wireless Communications and Mobile Computing Conference*, pages 1075–1080. IEEE, 2008.

[34] Aiello et al. Tinymaps: A lightweight java-based mobile agent system for wireless sensor networks. In *Intelligent Distributed Computing V*, pages 161–170. Springer, 2011.

[35] Lingaraj et al. Eagilla: An enhanced mobile agent middleware for wireless sensor networks. *Alexandria engineering journal,Elsevier*, 57(3):1197–1204, 2018.

[36] Khanum et al. Mobile agent based hierarchical intrusion detection system in wireless sensor networks. *International Journal of Computer Science Issues (IJCSI),Citeseer*, 9(1):101, 2012.

[37] Thamilarasu et al. Autonomous mobile agent based intrusion detection framework in wireless body area networks. In *2015 IEEE 16th international symposium on a world of wireless, mobile and multimedia networks (WoWMoM)*, pages 1–3. IEEE, 2015.

[38] Finogeev et al. Information attacks and security in wireless sensor networks of industrial SCADA systems. *Journal of Industrial Information Integration, Elsevier*, 5:6–16, 2017.

[39] Raman et al. Comparative analysis of delay minimizing and energy conserving algorithms in wireless sensor network. In *2018 2nd International Conference on Trends in Electronics and Informatics (ICOEI)*, pages 808–812. IEEE, 2018.

[40] Jianjian et al. A novel intrusion detection system based on iabrbfsvm for wireless sensor networks. *Procedia computer science,Elsevier*, 131:1113–1121, 2018.

[41] Alshaikh et al. Machine learning for detecting stuck pipe incidents: Data analytics and models evaluation. In *International petroleum technology conference*. OnePetro, 2019.

[42] Jasminder Sandhu et al. Enhancing dependability of wireless sensor network under

flooding attack: a machine learning perspective. *International Journal of Ad Hoc and Ubiquitous Computing,Inderscience Publishers (IEL)*, 33(2):73–89, 2020.

[43] Davahli et al. Hybridizing genetic algorithm and grey wolf optimizer to advance an intelligent and lightweight intrusion detection system for iot wireless networks. *Journal of Ambient Intelligence and Humanized Computing,Springer*, 11:5581–5609, 2020.

[44] Chaudhary et al. Energy-efficient and secured mobile agent itinerary approach in wireless sensor network. In *Innovations in Electrical and Electronic Engineering: Proceedings of ICEEE 2020*, pages 693–711. Springer, 2021.

[45] Rincy N et al. Design and development of an efficient network intrusion detection system using machine learning techniques. *Wireless Communications and Mobile Computing, Hindawi Limited*, 2021:1–35, 2021.

[46] Aranganathan et al. An efficient secure detection and prevention of malevolent nodes with lightweight surprise check scheme using trusted mobile agents in mobile ad-hoc networks. *Journal of Ambient Intelligence and Humanized Computing ,Springer*, 10:3493–3503, 2019.

[47] Thamilarasu et al. An intrusion detection system for internet of medical things. *IEEE Access*, 8:181560–181576, 2020.

[48] Selva et al. Intelligent network intrusion prevention feature collection and classification algorithms. *Algorithms, MDPI*, 14(8):224, 2021.

[49] El Fissaoui et al. Multi-mobile agent itinerary planning-based energy and fault aware data aggregation in wireless sensor networks. *EURASIP Journal on Wireless Communications and Networking*, 2018:1–11, 2018.

[50] Riyaz et al. A deep learning approach for effective intrusion detection in wireless networks using CNN. *Soft Computing, Springer*, 24:17265–17278, 2020.

[51] Lohani et al. Energy efficient data aggregation in mobile agent based wireless sensor network. *Wireless Personal Communications, Springer*, 89:1165–1176, 2016.

[52] Yang et al. Wireless network intrusion detection based on improved convolutional neural network. *IEEE Access*, 7:64366–64374, 2019.

[53] Geethapriya Thamilarasu et al. An intrusion detection system for Internet of medical things. *IEEE Access*, 8:181560–181576, 2020.

[54] Gandhimathi et al. Mobile malicious node detection using mobile agent in cluster-based wireless sensor networks. *Wireless Personal Communications , Springer*, 117:1209–1222, 2021.

[55] Seyedali Mirjalili. How effective is the grey wolf optimizer in training multi-layer perceptrons. *Applied intelligence ,Springer*, 43:150–161, 2015.

[56] Dutta et al. Border collie optimization. *IEEE Access*, 8:109177–109197, 2020.

[57] Wang et al. An intelligent data gathering schema with data fusion supported for mobile sink in wireless sensor networks. *International Journal of Distributed Sensor Networks, SAGE*, 15(3):1550147719839581, 2019.

[58] Wang et al. An enhanced pegasis algorithm with mobile sink support for wireless sensor networks. *Wireless Communications and Mobile Computing , Hindawi Limited*, 2018:1–9, 2018.

[59] Xiong et al. Robust dynamic network traffic partitioning against malicious attacks. *Journal of Network and Computer Applications,Elsevier*, 87:20–31, 2017.

[60] Chen et al. Citywide traffic flow prediction based on multiple gated spatio-temporal convolutional neural networks. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 14(4):1–23, 2020.

[61] Tcydenova et al. Detection of adversarial attacks in AI-based intrusion detection systems using explainable ai. *Human-Centric Comput Inform Sci, HCCIS*, 11, 2021.

[62] Mirjalili et al. Grey wolf optimizer. *Advances in engineering software ,Elsevier*, 69:46–61, 2014.

[63] Xu et al. Multi-agent modeling and jamming-aware routing protocols for movable-jammer-affected WSNs. *Sensors, MDPI*, 23(8):3846, 2023.

[64] Jatti et al. Sinkhole attack detection and prevention using agent based algorithm. *Journal of University of Shanghai for Science and Technology*, 23(5):526–544, 2021.

[65] Farooq et al. Multi-mobile agent trust framework for mitigating internal attacks and augmenting RPL security. *Sensors,MDPI*, 22(12):4539, 2022.

[66] Sethi et al. Attention based multi-agent intrusion detection systems using reinforcement learning. *Journal of Information Security and Applications,Elsevier*, 61:102923, 2021.

[67] Deng et al. Retracted article: mobile network intrusion detection for iot system based on transfer learning algorithm. *Cluster Computing,Springe*, 22(Suppl 4):9889–9904, 2019.

[68] Islabudeen et al. A smart approach for intrusion detection and prevention system in mobile ad hoc networks against security attacks. *Wireless Personal Communications,Springer*, 112:193–224, 2020.

[69] Ajay Kumar et al. Intrusion detection and prevention system for an iot environment. *Digital Communications and Networks, Elsevier*, 8(4):540–551, 2022.

[70] Dbouk et al. A novel ad-hoc mobile edge cloud offering security services through intelligent resource-aware offloading. *IEEE Transactions on Network and Service Management*, 16(4):1665–1680, 2019.

[71] Kushwah et al. A novel mobile agent-based intrusion detection framework for network security using SL-GAT and PP-FQCC. *Educational Administration: Theory*

and Practice, 30(5):5051–5062, 2024.

[72] Prabhjot Kaur et al. Border-hunting optimization for mobile agent-based intrusion detection with deep convolutional neural network. *Concurrency and Computation: Practice and Experience,Wiley Online Library*, 36(1):e7876, 2024.

[73] Zhihua Xia et al. STR: Secure computation on additive shares using the share-transform-reveal strategy. *IEEE Transactions on Computers*, 2021.

[74] Soltani et al. A multi-agent adaptive deep learning framework for online intrusion detection. *Cybersecurity, Springer*, 7(1):9, 2024.

[75] Abu Bakar et al. An intelligent agent-based detection system for DDOS attacks using automatic feature extraction and selection. *Sensors,MDPI*, 23(6):3333, 2023.

[76] Venkatesan et al. Advanced mobile agent security models for code integrity and malicious availability check. *Journal of Network and Computer Applications, Elsevier*, 33(6):661–671, 2010.

[77] Venkateswaran et al. An efficient neuro deep learning intrusion detection system for mobile adhoc networks. *EAI Endorsed Transactions on Scalable Information Systems*, 9(6), 2022.

[78] Chen et al. Egret swarm optimization algorithm: an evolutionary computation approach for model free optimization. *Biomimetics, MDPI*, 7(4):144, 2022.

[79] Saidala et al. Northern bald ibis optimization algorithm: theory and application. In *2018 IEEE 17th International Conference on Cognitive Informatics & Cognitive Computing (ICCI* CC)*, pages 541–551. IEEE, 2018.

[80] Tamil Nadu. Optimized data routing using PSO in WSN. 2019.

[81] Hajisalem et al. A hybrid intrusion detection system based on ABC-AFS algorithm for misuse and anomaly detection. *Computer Networks, Elsevier*, 136:37–50, 2018.

[82] Hussan et al. DDOS attack detection in iot environment using optimized elman recurrent neural networks based on chaotic bacterial colony optimization. *Cluster Computing, Springer*, pages 1–22, 2023.

[83] Hamidouche et al. An efficient clustering strategy avoiding buffer overflow in iot sensors: A bio-inspired based approach. *IEEE Access*, 7:156733–156751, 2019.

[84] Keserwani et al. An optimal intrusion detection system using GWO-CSA-DSAE model. *Cyber-Physical Systems,Taylor & Francis*, 7(4):197–220, 2021.

[85] Ahmad et al. Robust intrusion detection for resilience enhancement of industrial control systems: An extended state observer approach. *IEEE Transactions on Industry Applications*, 2023.

[86] Mingjun et al. A network intrusion detection method based on SSA-BRF. *Journal of Hebei University (Natural Science Edition)*, 42(5):552, 2022.

[87] Niklas Borselius. Mobile agent security. *Electronics & Communication Engineering*

Journal, IET, 14(5):211–218, 2002.

[88] Karnik et al. Design issues in mobile agent programming systems. *IEEE concurrency*, 6(3):52–61, 1998.

[89] Greenberg et al. Mobile agents and security. *IEEE Communications magazine*, 36(7):76–85, 1998.

[90] Bellavista et al. Mobile agent middleware for mobile computing. *Computer,IEEE*, 34(3):73–81, 2001.

[91] Fawzi Breidi. *Lightweight intrusion detection for mobile devices.(c2012)*. PhD thesis, Lebanese American University, 2012.

[92] Saleem et al. Secure transfer of environmental data to enhance human decision accuracy. *Computers in Human Behavior, Elsevier*, 51:632–639, 2015.

[93] Gupta et al. Energy and trust aware mobile agent migration protocol for data aggregation in wireless sensor networks. *Journal of Network and Computer Applications, Elsevier*, 41:300–311, 2014.

[94] Dong et al. Mobile agent-based energy-aware and user-centric data collection in wireless sensor networks. *Computer networks, Elsevier*, 74:58–70, 2014.

[95] Brahmkstri et al. Ontology based multi-agent intrusion detection system for web service attacks using self learning. In *Networks and Communications (NetCom2013) Proceedings of the Fifth International Conference on Networks & Communications*, pages 265–274. Springer, 2014.

[96] Adi Narayana Reddy et al. An enhanced probabilistic encryption algorithm for secured data transmission. In *International Conference on Computing and Communication Systems*, pages 284–290. Springer, 2011.

[97] Gope et al. Enhanced secure mutual authentication and key agreement scheme preserving user anonymity in global mobile networks. *Wireless Personal Communications,Springer*, 82:2231–2245, 2015.

[98] He et al. Anonymous two-factor authentication for consumer roaming service in global mobility networks. *IEEE Transactions on Consumer Electronics*, 59(4):811–817, 2013.

[99] Heydari et al. An efficient password-based authenticated key exchange protocol with provable security for mobile client–client networks. *Wireless Personal Communications, Springer*, 88:337–356, 2016.

[100] Rajwinder Singh and Mayank Dave. Antecedence graph approach to checkpointing for fault tolerance in mobile agent systems. *IEEE Transactions on Computers*, 62(2):247–258, 2011.

[101] Geetha et al. Implementation of trust and reputation management for free-roaming mobile agent security. *IEEE Systems Journal*, 9(2):556–566, 2014.

[102] Nguessan et al. Framework for security and privacy management for mobile middleware based on tuple. *IEEE Latin America Transactions*, 13(8):2757–2762, 2015.

[103] Aydın et al. A hybrid intrusion detection system design for computer network security. *Computers & Electrical Engineering, Elsevier*, 35(3):517–526, 2009.

[104] Sander et al. Protecting mobile agents against malicious. *Mobile Agents and Security*, 1419:44, 2003.

[105] Esparza et al. Punishing malicious hosts with the cryptographic traces approach. *New Generation Computing ,Springer*, 24:351–376, 2006.

[106] Subashini et al. A survey on security issues in service delivery models of cloud computing. *Journal of network and computer applications, Elsevier*, 34(1):1–11, 2011.

List of Publications

1. Prabhjot Kaur, Shalini Batra and Prashant Singh Rana, *"Border-Hunting Optimization for Mobile Agent-Based Intrusion Detection With Deep Convolutional Neural Network"*, Concurrency and Computation: Practice and Experience, Wiley, 36(1):e7876, 2023. [SCIE, IF 1.5, Q3]

2. Prabhjot Kaur and Prashant Singh Rana, *"Self-Configuring Mobile-Agent based Intrusion Detection using Hybrid optimized with Deep LSTM"*, Knowledge Based Systems, Elsevier, 304(1):112316-332, 2024. [SCIE, IF 7.2, Q1]

3. Prabhjot Kaur and Prashant Singh Rana, *"Using a Mobile Agent-Based Intrusion Detection System, an Ensemble Model for Automated Attack Classification in Large-Scale Wireless Sensor Networks"*, Wireless Networks, Springer, 2024. [SCIE, IF 2.1, Q3, Under Minor Review]

4. Prabhjot Kaur and Prashant Singh Rana, *"Enhancing Security with Dynamic Blow Filter and Blow Fish Security Protocol to Safeguard against Malicious Mobile Agents"*, Microprocessors and Microsystems, Elsevier, 2024. [SCIE, IF 1.9, Q3, Under Review]