University of Sheffield

Adaptive Intrusion Detection System for 6LoWPAN

Aryan Mohammadi Pasikhani

Supervisors:

Prof. John A Clark
Dr Prosanta Gope

This dissertation is submitted
for the degree of *Doctor of Philosophy*

in the

Department of Computer Science
The University of Sheffield

June 20, 2022

Declaration

All sentences or passages quoted in this document from other people's work have been specifically acknowledged by clear cross-referencing to author, work and page(s). Any illustrations that are not the work of the author of this report have been used with the explicit permission of the originator and are specifically acknowledged. I understand that failure to do this amounts to plagiarism and will be considered grounds for failure.

Name: _____

Signature: _____

Date: _____

I would like to dedicate this thesis to my beloved Mum.

Abstract

Drastic reduction in the manufacturing cost of sensors and actuators has resulted in considerable growth in the number of smart objects. The so-called Internet of Things (IoT) blends the real and virtual environments and removes time and distance barriers. It is widely perceived as a major enabler for the efficient and effective provision of services across a range of sectors. Low power and lossy networks have grown in importance in recent years. A good deal of work has been carried out to provide routing with desirable characteristics over such networks. Of particular note is the Routing Protocol for Low Power and Lossy Networks, generally referred to as RPL. This is a flexible protocol that can provide routing for the needs of various applications (such as smart agricultural systems, smart-city, and smart-home environments). However, the protocol itself is subject to attack with severe consequences. Researchers have proposed different security infrastructures to mitigate harm to IoT networks. One of these is the Intrusion Detection System (IDS). An IDS is an essential component for network security and is widely adopted to reinforce the security of the Low Power and Lossy Network. IDSs for detecting RPL attacks must also cope with the often significant resource constraints that apply in such networks. Furthermore, due to the evolving nature of 6LoWPAN data streams, the performance of batch-trained/offline IDSs based on machine learning may degrade dramatically when computer network traffic changes.

In this thesis, we empirically investigated several machine learning (ML) algorithms (e.g. OZABagging, KNNADWIN, and One-Class Support Vector Machine), concept-drift detection algorithms (e.g. ADWIN, DDM, and EDDM), and reinforcement learning algorithms (Deep Q-Network and Double Deep Q-Network) to develop efficient, robust, and generalised IDSs for 6LoWPAN. For the first time, we propose an adaptive hybrid ML-based IDS to efficiently identify a wide range of RPL attacks in an evolving network environment. We propose an adversarial reinforcement learning framework to generate efficient and generalised incremental ML-based IDS agents for 6LoWPAN.

We apply our frameworks to networks under various numbers of nodes with varying levels of mobility and node maliciousness. To emulate different RPL attacks and measure the performance of the proposed schemes, we use the Tetcos Netsim simulator. The proposed schemes can detect various RPL attacks, including several intrusions unaddressed by current research. The outcomes of our experiments show that the proposed schemes are well suited to the resource-constrained environments of our target networks.

Acknowledgment

I want to express my deep and sincere gratitude to my supervisors, Professor John A Clark and Dr Prosanta Gope, who have been a constant source of knowledge, advice, inspiration, and support to achieve the accomplishments within this thesis.

I will be eternally grateful to my Mum and Dad for their unconditional love and for encouraging me all the way. My gratitude also goes to my brother for his encouragement during my studies.

I would like to thank my examiners, Dr Andrei Popescu and Dr Vassilios G. Vassilakis, for their insightful comments on improving this manuscript.

I would also like to thank my friends and colleagues in the Security of Advanced Systems Research Group at the University of Sheffield for making my PhD journey a great experience.

Contents

List of Figures

List of Tables

Nomenclature

Acronyms

$\hat{Q}$	Target action-value function
$\mathcal{A}$	Action Space
a	Agent action
c	Concept
D	Underlying data distribution
E	Quality signal
E_{min}	A minimum quality threshold
G	Discount Reward
m	Model $m \in \mathcal{M}$
Q	Action-value function
r	Reward
s	State
X	Feature set $X = \{x_1, x_2...x_N\}$
y	Ground-truth/label of observation X
6BR	6LoWPAN Border Router
6LoWPAN	IPv6 over Low-Power Wireless Personal Area Network
ADWIN	Adaptive Windowing method for concept-drift detection
ANIDS	Anomaly-based NIDS
BH	Blackhole attack
CD	Concept-drift Detection
CI	Clone Id attack
CIA	Confidentiality Integrity Availability
CIDS	Central IDS
CM	Central Manager
CPO	Control Packet Overhead
DA	DIS flooding Attack
DAG	Directed Acyclic Graph
DAO	Destination Advertisement Object
DAO-ACK	Destination Advertisement Object Acknowledgment
DDM	Drift Detection method for concept-drift detection
DDQN	Double Deep Q Networks
DI	DAO Inconsistency attack
DIO	DOADAG Information Object
DIS	DODAG Information Solicitation
DODAG	Destination Oriented Directed Acyclic Graph
DQN	Deep Q-Network
DR	Detection Rate
DS	DIO Suppression attack
E2E	End to End Delay
EDDM	Early Drift Detection method for concept-drift detection
ETX	Expected Transmission Count
FN	False Negative
FNR	False Negative Rate
FP	False Positive
FPR	False Positive Rate
GH	Grayhole attack
HDDM	Hierarchical Bayesian estimation of the Drift-Diffusion Model for concept-drift detection
HIDS	Host-based IDS
i.i.d	Independent and identically distributed

IDS	Intrusion Detection System
IoT	Internet of Things
IP	Internet Protocol
IPS	Intrusion Prevention System
IPv6	Internet Protocol version 6
IR	Increase Rank attack
KSWIN	Kolmogorov-Smirnov Windowing method for concept-drift detection
LBR	Low power and lossy Border Router
LLN	Low power and Lossy Network
LoWPAN	Low-Power Wireless Personal Network
LQ	Link Quality
LR	Local Repair Attack
ML	Machine Learning
MRHOF	Minimum Rank with Hysteresis OF
NIDS	Network-based IDS
OF	Objective Function
OF-EOR	Energy-Oriented Routing OF
OF0	Objective Function Zero
PAN	Personal Area Network
PDR	Packet Delivery Ratio
RA	Rank Attack
RL	Reinforcement Learning
ROC	Receiver operating characteristic
RPL	Routing Protocol for Low-Power and Lossy Networks
RSSI	Received Signal Strength Indication
SF	Selective Forwarding attack
SH	Sinkhole attack
SPoF	Single Point of Failure
TN	True Negative
TNR	True Negative Rate
TP	True Positive
TPR	True Positive Rate
UDP	User Datagram Protocol
VN	Version Number attack
WH	Wormhole attack
WP	Worst Parent attack
WSN	Wireless Sensor Network

Greek symbols

α	Learning rate
ΔR	Difference between expected and actual r
δ	Discrepancy between D_t and D_{t+1}
γ	Discount Factor
λ	Replay Memory
∇	Gradient descent
π	Policy
ψ	Q-learning target
$arg\max_a f(a)$	a value of 'a' at which $f(a)$ takes its maximal value
θ	Weights of NN
ε	Exploration probability

Chapter 1

Introduction

The Internet of Things (IoT) is a promising technology that is rapidly gaining ground in the scenario of modern wireless telecommunications. The IoT empowers ubiquitous connections amongst numerous Things (temperature, humidity, and turbidity sensors, controllers, storage, motors, valves etc) to sense, analyse, control and record information on aspects of a wide range of environments. Since the number of IoT devices increases each day and they are often deployed in hostile, unattended, and unfavourable conditions, securing them becomes a colossal challenge. In order to expand the connectivity of nodes in the Low-power and Lossy Network (LLN) and make them accessible through the Internet, the IETF has standardised the Routing Protocol for LLNs (RPL) [1]. The RPL is developed for IPv6 over Low-powered Wireless Personal Area Networks (6LoWPAN). The 6LoWPAN is a low cost and low-power communication network that connects resource-constrained actuators and wireless sensors by utilising a compressed IPv6 protocol for networking and IEEE 802.15.4 as a data-link and physical layer protocol. Conceivably every physical item can be connected to the 6LoWPAN. Furthermore, the inexpensive manufacturing processes and the communicating-actuating capabilities of the LLN nodes have made the 6LoWPAN both economically and technically desirable for various applications (e.g. industrial automation, shipment tracking, surveillance, and environmental controls [2]). Because of the global connectivity, resource constraints and RPL vulnerabilities, the 6LoWPAN is exposed to various routing threats internally (within the 6LoWPAN) and externally (through the Internet). The existing routing attacks (e.g. blackhole, grayhole, wormhole, DIS flooding) [3] cause the RPL to adopt a sub-optimal routing topology, isolate legitimate nodes, and cause significant overheads over the target network and nodes to endanger confidentiality, integrity and availability (CIA) of the 6LoWPAN streaming data.

In this regard, an efficient and effective Intrusion Detection System (IDS) is of utmost importance in identifying anomalous activities in the IoT networks. A Network-based Intrusion Detection System (NIDS) [3] analyses collected observations in the given network to identify abnormal behaviours. The IDS will generate false positives (classifying a legitimate activity as abnormal) and false negatives (classifying a malicious activity as safe). The IDS can be classified based on various axes, e.g. monitoring technique, source of data, and detection strategy [3]. The detection strategy of IDS can be classified as signature-based, anomaly-based, specification-based, and hybrid [3]. A signature-based IDS compares the network observation against predefined attack profiles (referred to as signatures). In contrast,

anomaly-based IDS profiles the network's legitimate activities and classifies observations as anomalous if they deviate from the expected profile. Signature-based IDS is known for causing low false positives; however, it cannot detect unforeseen intrusions, requires considerable storage space and may cause high false negatives. On the other hand, anomaly-based IDS is capable of identifying unforeseen intrusions but may cause high false-positives. To incorporate the strengths of each detection strategy researchers have developed hybrid IDSs for 6LoWPAN [3]. According to [3], existing IDSs mainly focus on detecting sinkhole (21%), grayhole (14%), blackhole (10%) and DIS flooding (10%) attacks while increase rank, DIO suppression, replay and worst parent attacks have received little or no attention.

Although the RPL was initially developed as a routing protocol for stationary LLN's, a broader usage of LLN nodes motivated researchers to develop mobility for RPL [3]. Only 13% of research has considered mobility in detecting RPL attacks [3]. It is vital to develop an IDS that can identify RPL attacks in the network environment with mobile nodes. Furthermore, because of the evolving nature of LLN environment (e.g. node mobility and application variation), the data distribution is non-stationary and shifting. Unpredictable and abnormal incidents evolve the network environment and induce shifts in the statistical distributions of data, known as concept drifts. The concept drift occurrences can degrade the performance of IDSs in LLNs and prevent them from accomplishing their primary tasks.

On the other hand, the streaming data in 6LoWPAN is imbalanced, i.e. the distribution of instances over the known classes is not equal. The class with abundant instances (legitimate activities) is called the majority class, whereas the class with a much fewer instances (malicious activities) is termed the minority class. In computer networks, the imbalance is a property of the problem domain, where the natural occurrence of one class dominates other classes. This is because the process that draws observations from the minority class has lower frequency. Imbalanced training data results in the development of biased models, specifically generating models that perform poorly on the minority class. This is a problem since the minority class is often more critical, with mis-classification of malicious activities as normal, indicating the IDS is simply failing at its primary tasks.

To the best of our knowledge, existing supervised and unsupervised IDS for 6LoWPAN in the literature are offline/batch-trained and mostly limited to stationary data environments [3]. Hence, the existing IDSs cannot identify or adjust to the changes in the network, such as the breakout of unforeseen intrusions or concept drifts. When implementing learning algorithms, one often faces the difficult problem of dealing with non-stationary environments whose dynamics evolve due to some unknown or not directly perceivable cause. In the evolving environment of 6LoWPAN, an IDS must analyse extensive, noisy, and imbalanced data. Therefore, a robust and generalised IDS is required to adapt to shifts in the streaming data and identify RPL attacks accurately.

1.1 Aims and Objectives

The Routing Protocol for LLNs (RPL) is vulnerable to various threats (such as sinkhole, blackhole, wormhole etc). Furthermore, the evolving data in LLN require a dynamic/robust means of updating the detection model in real-time. The 6LoWPAN has a non-stationary /dynamic data environment, where the network data distribution evolves on an unpredictable basis. To maintain detection performance, it is expected that the IDS modify its detection

model on a regular basis and incrementally adapt to unforeseen activities. Different IDSs have been proposed in the literature to detect existing RPL attacks in 6LoWPAN (will be discussed in Chapter 2). However, the existing IDSs for 6LoWPAN can address only stationary network environments and are unable to adapt to any changes in network configurations. Moreover, because of resource constraints in the LLNs, an IDS cannot explicitly store all the past traffic in a streaming environment to identify anomalous activities in low power and lossy nodes. In this regard, an incremental machine-learning-based IDS should be able to secure 6LoWPAN from any internal and external routing intrusions. Hence, our *first* formal hypothesis is:

***Hypothesis 1**: An adaptive heterogeneous hybrid IDS can identify various internal and external RPL attacks in 6LoWPAN. The adoption of concept-drift detection approaches can enable an incremental ML-based IDS to maintain and enhance its intrusion detection performance over time by adapting to unforeseen intrusions and data distributions.*

Since 6LoWPAN can have a large scale network topology, it is unclear whether a centralised IDS on the border router can detect RPL attacks when a malicious node[1] targets nodes at the lower level of the network hierarchy. In this regard, a passive decentralised reinforcement-learning-based (RL-based) IDS can be able to facilitate intrusion detection in 6LoWPAN. As a result, our *second* hypothesis is:

***Hypothesis 2**: An RL-based IDS framework can enhance the strength of distributed ML-based IDS in detecting RPL intrusions. An RL-based IDS will be able to accurately identify suspicious activities with the help of ML-based IDS agents.*

The streaming data in LLN require a dynamic means of updating the detection model on the fly. Moreover, since the IDS agents cannot accommodate the sheer amount of streaming data, the IDS needs to train on the succinct (more informative) data only. In this regard, the application of adversarial reinforcement learning can facilitate the generation of robust and generalised incremental ML-IDS. The developed IDS agents can make reasonable trade-offs between the variance and bias for detecting RPL attacks in evolving and imbalanced data environments of 6LoWPAN. Hence, our *third* hypothesis is:

***Hypothesis 3**: An adversarial RL-based IDS framework can generate efficient detectors using imbalanced training data. The integration of an adversarial RL environment and incremental machine-learning can facilitate the formation of resource-efficient, generalised and robust IDS detectors.*

1.2 Thesis Contributions

The major contributions of this thesis are as follows:

- Proposal and evaluation of an adaptive heterogeneous hybrid ML-IDS, maintaining its

[1] in this dissertation, we use malicious node and intruder interchangeably.

effectiveness against environmental change in different scaled 6LoWPANs.

- Proposal and evaluation of an RL-IDS framework to enhance the strength of distributed ML-IDSs in detecting internal and external RPL intrusions.
- Demonstration of a robust adversarial RL-based IDS framework that can generate resource-efficient adjustable detectors using imbalanced training data.
- Demonstration of an effective IDS that is capable of identifying and distinguishing a wide range of RPL attacks, including less researched ones; in this dissertation, we develop an IDS capable of detecting Increase Rank and DIO Suppression attacks for the first time.
- Construction of a set of features in a principled way that can facilitate the identification of RPL attacks.
- Demonstration of an IDS which is resilient against known and previously unseen RPL intrusions.
- Demonstration of an IDS which is capable of detecting intelligent *ML-based combinational intrusions* against 6LoWPAN.
- Presentation of the first application of reinforcement learning and incremental machine learning for IDS in 6LoWPAN.

1.2.1 Desirable properties

Below we identify various desirable properties (DPs) that could be expected of a high performing IDS in our target domain. These are based on our own views and those of other researchers [3; 4; 5]. In this context, this dissertation aims to achieve the following DPs.

- **DP1:** the IDS must be adaptive and generalised to enhance its model performance in the evolving data environment of 6LoWPAN.
- **DP2:** in case of a concept-drift, the IDS needs to adjust its detection model and tackle the joint issue of concept drift and class imbalances.
- **DP3:** due to the resource-constraint nature of the LLN's nodes, the IDS should not need excessive memory and computational resources whilst being able to identify routing attacks precisely.
- **DP4:** the IDS should be able to detect a wide range of RPL attacks. Published IDS schemes address only subsets of known RPL attacks and do not evaluate outside the chosen subsets.
- **DP5:** the IDS should be able to detect known and previously unseen intrusions.
- **DP6:** the IDS must be able to identify an ML-based adversary using combinational attack strategies.
- **DP7:** the IDS should be capable of identifying routing attacks in 6LoWPAN in the presence of varying numbers of mobile nodes.

1.3 Structure of the Thesis

The thesis is structured as follows:

- **Chapter 2: "Systematic Literature Review"** outlines the existing IDS architectures for 6LoWPAN and provides an IDS taxonomy. It explores the Routing Protocol for Low Power and Lossy Networks (RPL) and its existing threats, classifies relevant IDS techniques and identifies areas requiring further investigation. The literature review examines 103 papers.
- **Chapter 3: "Adaptive Hybrid Heterogeneous IDS"** presents an adaptive hybrid IDS to efficiently detect and identify a wide range of RPL attacks in non-stationary environments. We apply our proposed framework to networks under various levels of node mobility and node maliciousness. We empirically explore the use of several incremental machine learning (ML) and 'concept-drift detection' (e.g. ADWIN, DDM, and EDDM) algorithms to achieve our goals.
- **Chapter 4: "Reinforcement-Learning-based IDS"** proposes a Reinforcement Learning (RL) based IDS to detect various attacks on RPL in 6LoWPANs, including those which have been overlooked by the existing literature. The proposed scheme can detect previously unseen attacks and identify mobile intruders accurately. The scheme is well suited to the resource constrained environments of 6LoWPAN.
- **Chapter 5: "Adversarial Reinforcement-Learning-based IDS"** proposes the use of adversarial reinforcement learning, concept-drift detection, and incremental ML algorithms for the development of a robust and generalised IDS, and propose a novel approach to incorporate the salient information of imbalanced attack profiles into the resource-constrained intrusion detectors. The developed IDS can make reasonable trade-offs between variance and bias for detecting RPL attacks in the evolving and imbalanced data environment of 6LoWPAN. Since the 6LoWPAN is subjected to various known and unknown routing intrusions and has an evolving data environment, adaptivity and concept-drift detection play vital roles in developing a robust IDS. In this chapter for the *first* time, we propose an IDS to detect black-box and grey-box ML-based adversaries aiming to destabilise the 6LoWPAN.
- **Chapter 6: "Conclusion and future work"** first presents a summary of the dissertation. Subsequently, the contributions of this dissertation are discussed and potential future work identified.

1.4 Publications

Works in this thesis have appeared in the following publications:

1. **Intrusion Detection Systems in RPL-Based 6LoWPAN: A Systematic Literature Review**
   A. M. Pasikhani, J. A. Clark, P. Gope, and A. Alshahrani
   IEEE Sensors Journal, 2021

2. **Adaptive Hybrid Heterogeneous IDS for 6LoWPAN**
   A. M. Pasikhani, J. A. Clark, and P. Gope
   Submitted to IEEE Transactions on Dependable and Secure Computing.

3. **Reinforcement-Learning-Based IDS for 6LoWPAN**
   A. M. Pasikhani, J. A. Clark, and P. Gope
   20th IEEE International Conference on Trust, Security and Privacy in Computing and Communication (TrustCom2021)

4. **Adversarial RL-based IDS for Evolving Data Environment in 6LoWPAN**
   A. M. Pasikhani, J. A. Clark, and P. Gope
   Submitted to IEEE Transactions on Information Forensics and Security.

Chapter 2

Literature Survey

2.1 Introduction

The Internet of Things (IoT) provides a framework where vast numbers of devices can communicate with each other and so collaborate to provide services across many domains with increased efficiency and effectiveness. Such enhanced operation is underpinned by increased sophistication of information processing. It has become common to describe various nodes as 'smart' but in practice, the degree of smartness encapsulated in individual nodes varies hugely. However, the ease with which nodes may now communicate means that highly sophisticated system operation is possible, with responsibilities for different aspects of service provision being distributed across the network. Quite basic nodes now play a critical role in the provision of such sophisticated services. However, many of these 'things' in IoT based systems suffer from limited computational and energy resources.

According to Cisco [6], over 75 billion devices are expected to connect to the Internet by the year 2025. Due to the increased number of IoT devices, IPv4 does not apply in this domain and use of IPv6 is essential. Sensors collect and actuate a massive amount of data that requires precise analysis. However, because of resource limitations, Low Power and Lossy Networks (LLNs) have to transfer generated data to a device/server with sufficient computational resources for storing it and for conducting computation tasks, such as data analysis. Information can then be sent to the actuators to take identified actions. The IPv6 over Low-Power Wireless Personal Area Networks (6LoWPAN) was developed to provide a compact IPv6 to LLN nodes in Personal Area Networks (PANs) and enable nodes to interact over the Internet.

Making actuators, sensors, and devices connected to outside wired networks is a profitable business. However, the resource limitations of LLN nodes makes them vulnerable to internal and external malice and raises many security concerns. Most nodes in LLNs are battery-powered, lack heavy-computation capabilities, and inherit inadequate computation capabilities and limited storage capacity as a consequence of the need for low manufacturing costs. Although IoT devices have slightly better computational resources than Wireless Sensor Network (WSN) nodes and are heterogeneous in design [7; 8], the devices usually need to work in an unstable environment, where the neighbouring nodes may go on and off to conserve energy. Parent nodes may move outside their children's range and become unreachable; therefore, children need to find new parents in a timely manner. These require a scalable,

energy- and resource-efficient routing protocol. The Routing Protocol for LLNs (RPL) is designed to address these needs.

A substantial amount of research has addressed security concerns in LLNs through cryptography, intrusion detection systems (IDSs), intrusion prevention systems (IPSs), authentication and trust-based mitigation approaches, etc. IoT devices and traditional computers use some similar protocols in the application, transport, network, and physical layers, as discussed in Section 2.2. The constrained computational and energy resources of a LLN's devices are the primary barrier to adopting existing security mechanisms at IoT interfaces [9]. LLN devices generate huge volumes of data but lack the resources to store and process it further.

RPL plays a critical and widespread role in service provision in IoT systems. As a consequence, it is an obvious target for attack and a critical candidate for defence. One aspect of defence that must now be considered is how intrusions on RPL systems can be detected. Here we present a systematic review of the most influential approaches and methods providing IDS for RPL networks. We examine existing routing threats and their negative impacts. The strengths and weaknesses of IDS strategies and mitigation techniques are also addressed, as are evaluation methods. Our review classifies and provides a taxonomy of these IDS methods.

Several surveys and reviews, such as [10], [11], and [12], study RPL functionality, IoT threats, and some proposed mitigation methods. However, the existing studies provide only a light introduction to IDSs for RPL and do not cover the state-of-the-art attacks. They are not comparative and do not evaluate the pros and cons of each proposed method from different perspectives. Our survey provides a more rigorous characterisation of IDS strategies together with finer grained evaluation than is currently available in the literature. The increased evaluation detail will facilitate further research and help those considering IDS implementation.

We comment on a wide range of aspects: objectives of each proposed approach, the effectiveness of each proposed detection strategies, monitoring techniques, and to what degree each research achieved its goals. We also consider evaluation methods, configuration setups, and testbeds used by researchers. Our study focuses on the most high profile RPL mitigation techniques, concentrating on IDSs.

The rest of this chapter is organized as follows. In Section 2.1.1 we define a set of research questions to provide increased rigour and focus to our review. Section 2.2 describes the functioning and the potential vulnerabilities of RPL. Section 2.3 details discovered intrusions and vulnerabilities of RPL. Next, IDS in terms of the source of monitored data, detection strategies, response types, monitoring techniques, and evaluation methods are classified and described in Sections 2.4.1, 2.4.2, 2.4.3, 2.5, respectively. After providing a taxonomy for IDSs for RPL, Section 2.7 addresses the identified research questions and provides statistical evidence for the findings based on our investigation of 103 papers. Finally, Section 2.8 determines the existing research gaps.

2.1.1 Scope of this survey

Our survey is concerned with IDSs that target RPL networks. Salient features and methods of papers in this context are extracted and the characteristics are detailed. We primarily aim to address the following research questions:

Q 1. What types of routing threats exist in this domain and how have they been addressed so far?

Q 2. What is the impact of each attack in an LLN and to what extent do they damage 6LoWPAN?

Q 3. What are the technical performance objectives of the research in this field?

Q 4. How do the proposed approaches monitor the network to detect anomalies and to what extent are particular monitoring methods used by researchers?

Q 5. What IDS strategies exist to protect RPL networks and how widely is each used?

Q 6. What are the advantages and disadvantages of each proposed approach?

Q 7. What datasets or simulators are available for RPL networks, and to what extent are they employed by researchers? What configuration and setup of simulated and test networks (e.g. number of intruders, the distances between nodes, runtime duration) have researchers used to conduct their experiments?

Q 8. What evaluation methods are used to measure the performance of the proposed methods and to what extent have the researchers improved performance?

Q 9. What are open questions in this domain, and what vulnerabilities remained unaddressed?

2.1.2 Selection of studies

In this chapter the publications, literature, and methods are chosen with regards to their research scope, expert opinions, and the quality of the publication and its impact on the research published afterward. Web of Knowledge, Google Scholar, and IEEE Xplore search engines were used to discover and obtain the proposed papers between the years 2008 and 2021.

2.1.3 Relevant reviews

In [19], the authors review the security issues and mitigation techniques of the edge layer in IoT. The paper does not cover RPL attacks and mitigation techniques. A comprehensive review of trust-based IDS for RPL is given by [12]. In [14], the authors study the impacts of a few RPL attacks on 6LoWPAN and to what degree the built-in security mechanisms of RPL can resolve the negative impacts of such attacks. [15] and [11] review several RPL attacks and a few mitigation techniques. However, they study only a few proposed method and do so in limited detail. In [10], the authors concentrate on classifying IDS approaches. They do not provide an overview of RPL attacks and their adverse impact on the 6LoWPAN nor discuss the computational cost of each IDS approach on the LLN. In [16], the authors provide a review on the use of Machine Learning (ML) based security infrastructures to detect security vulnerabilities of IoT, but do not include RPL attacks. Table 2.1 shows the contribution, scope, and shortcomings of each related review and indicates how our work complements it.

2.2 Preliminaries

Several network protocols have been introduced that form connections between IoT devices and the various computers on the Internet, as shown in Fig. 2.1 [20]. Such protocols can be classified based on network coverage, energy overhead, and transmission rate. The 6LoWPAN

Table 2.1: *Related reviews*

Ref	Contributions	Scope	Areas needing to be further addressed	Improvements made by *our review*
[13]	Provides a detailed review of the RPL protocol, its related attacks, and some proposed mitigation strategies.	Introduces several mitigation approaches for RPL attacks. including some IDS proposals enhancing the security of RPL.	Provides an interesting description of RPL-based attacks, but omits in-depth analysis of IDS-specific designs and proposals.	The core of their review is similar to Q1, Q2, Q6, Q7 and Q9. However, we investigate and extract more relevant research questions such as Q3, Q4, Q5 and Q8.
[12]	Introduces RPL vulnerabilities and trust-based security mechanisms	IoT routing methods and existing vulnerabilities of such routing mechanisms.	The reviewed methods were designed mostly for WSNs. Their study does not cover any IDS approach.	A greater range of highlighted vulnerabilities are given in this chapter. Q3, Q4, Q5, Q6, Q7 and Q8 of our study provide additional analysis.
[14]	Discusses the negative impacts of some RPL attacks on the IoT network and to what degree the healing mechanism of RPL can counter them.	Defines negative impacts of some RPL attacks on 6LoWPAN. Indicates to what degree built-in RPL mechanisms can enhance security. Identifies the importance of developing RPL security mechanisms.	Introduces some RPL attacks. IDS is described briefly without detailed classification	RPL attacks and IDS methods are classified and expanded in our review. Q3, Q4, Q5, Q6, Q7 and Q8 address this.
[15]	Provides a comprehensive review of RPL attacks. The attacks are classified based on the damage they cause in the network	Introduces majority of RPL attacks and briefly describes their adverse impact.	Limited number of earlier RPL attacks. Very brief review over proposed mitigation methods.	The attacks are expanded on and discussed in more detail. Mitigation techniques are classified and introduced based on the cost they cause to the LLN. Q3, Q5, Q6, Q7, Q8 and Q9 augment the presented work.
[11]	Gives a brief survey of RPL attacks and their mitigation mechanisms. Includes some IDS research proposals.	RPL specific attacks. Intrusions on the 6LoWPAN adoption layer. IDS classification.	There is a need to thoroughly review RPL targeted attacks and mitigation methods and to present a detailed description of the IDSs.	Major RPL attacks and IDS proposals are covered. Research questions: Q3, Q4, Q6, Q7 and Q8 are not addressed.
[10]	Provides a classification of IDSs in IoT.	Classifies IDS detection methods, placement strategy, and evaluation techniques	A limited number of RPL attacks and IoT security issues are addressed. Also, monitoring methods are not studied comprehensively. Affects on LLN networks are not addressed	Study provides a more comprehensive classification and taxonomy of IDS. The answers of Q1, Q2 and Q4. enhance their shortcomings.
[16]	Reviews ML-based approaches for mitigating IoT attacks. Identifies limitations of ML in securing IoT infrastructures.	IoT vulnerabilities, mitigation using ML approaches, and computational costs of ML approaches.	Reviews IoT security in general and not RPL security specifically. Only a few IDS methods are reviewed briefly. RPL-based attacks not included.	Proposed ML-based IDSs are reviewed and their strengths and shortcomings identified. Q1, Q2, Q3 and Q5 of our study augment their work.
[17]	Surveys RPL and its existing vunerabilities. Review some proposed IDS and mitigation methods.	RPL vunerability and mitigation techniques.	Some proposed IDS are not adequately classified in their context. A comparison of proposed techniques is not made.	Q4 to Q7 address the indicated shortcomings.
[18]	Reviews 97 studies related to RPL, in various directions and domains. Provide some statstics on the reviewed papers.	RPL in general.	Briefly describes a few security methods of RPL and existing threats. Does not cover any IDS. The study is not security-focused and provides a very general overview of RPL security concerns and existing mitigation methods.	we provide different statistics for the proposed IDS for IoT. The unanswered questions, Q1, Q2, Q3, Q4, Q5, Q6 and Q8, would complement their work.

is the combination of IPv6 and Low-Power Wireless Personal Network (LoWPAN) protocols and is the adaption layer of IPv6 for 802.15.4 LLNs. It is a version of IPv6 optimised for LLNs and was mainly designed to provide Internet connections to resource-constrained nodes in PANs (Personal Area Networks). However, 6LoWPAN can be implemented over different platforms and is not restricted to radio links. The 6LoWPAN follows the IEEE802.15.4 standard and covers connection ranges of 10s of meters with ∼250Kb/s transmission rate. Implementing the original IPv6 is computationally expensive for LLNs. This adaption aims to provide mechanisms to reduce computational expense, such as address header compression, packet fragmentation, IPv6 neighbor discovery requirements. There is no default IoT stack layer in this domain; however, we can see the standards and protocols introduced for IoT from different perspectives. Fig. 2.1 illustrates the TCP/IP protocol layer and the most common stack layer described by researchers [20].

2.2.1 Routing Protocol for Low-power and Lossy Networks (RPL)

The Routing Protocol for low-Power and Lossy Networks (RPL) is designed to provide IPv6 communication among LLNs, i.e., IoT devices. LLNs include constrained devices with limited memory, processing power, and sometimes battery operated energy resources. Such devices have a lossy connection, typically supporting only low data rates that are usually unstable with relatively low packet delivery rates (PDRs). RPL was initially designed to work in an environment with static nodes in fixed locations; however, some mobility methods [21] were introduced that enable the participation of mobile nodes in the RPL network. In general, LLNs that use RPL are known for a meagre data rate; usually something below ∼250kbps [10], and also very high collision and dropped packet rates, which negatively impact the application throughput. RPL supports three kinds of communication: P2P (peer to peer also called point to point); P2MP (a central node to multiple points on network); and MP2P (from multiple nodes to a central server). It uses a distance vector routing protocol based on its Destination Oriented Directed Acyclic Graph (DODAG). Multiple RPL instances, each with a unique RPLinstanceid, can operate concurrently in the LLN. RPL is capable of constructing multiple paths back to the same destination and switches to alternative routes whenever default routes become corrupted (either intentionally or unintentionally). The protocol generates a directed acyclic graph (DAG) based on associated policies imposed by Low power and lossy Border Router (LBR). The LBR is the root of the DAG and usually has rank 1. This border router and the rest of the nodes interconnect in a hierarchical structure, which combines mesh and tree topologies referred to as a DODAG. The ranking system was designed to prevent and detect any probable routing loop in RPL. The rank enables LLN nodes to identify their parents and children. The RPL requires the nodes to store a list of parents, allowing a child node to switch to another parent easily should a current one become unavailable. The Rank in RPL is computed based on the distance from the 6BR with different metrics, such as Link Quality (LQ), Delay, Hop Counts, Connectivity etc, defined in the Objective Function (OF).

Although RPL is the most popular and standerdised routing protocol in IoT networks [22], several other protocols have also been developed to enhance routing in LLNs, namely CORPL (Cognitive RPL) [23], and CARP (Channel Aware Routing Protocol) [24]. The CORPL is an extended version of RPL and designed for cognitive networks. However, unlike RPL, it does not support storage management, and all nodes need to track forwarding records.

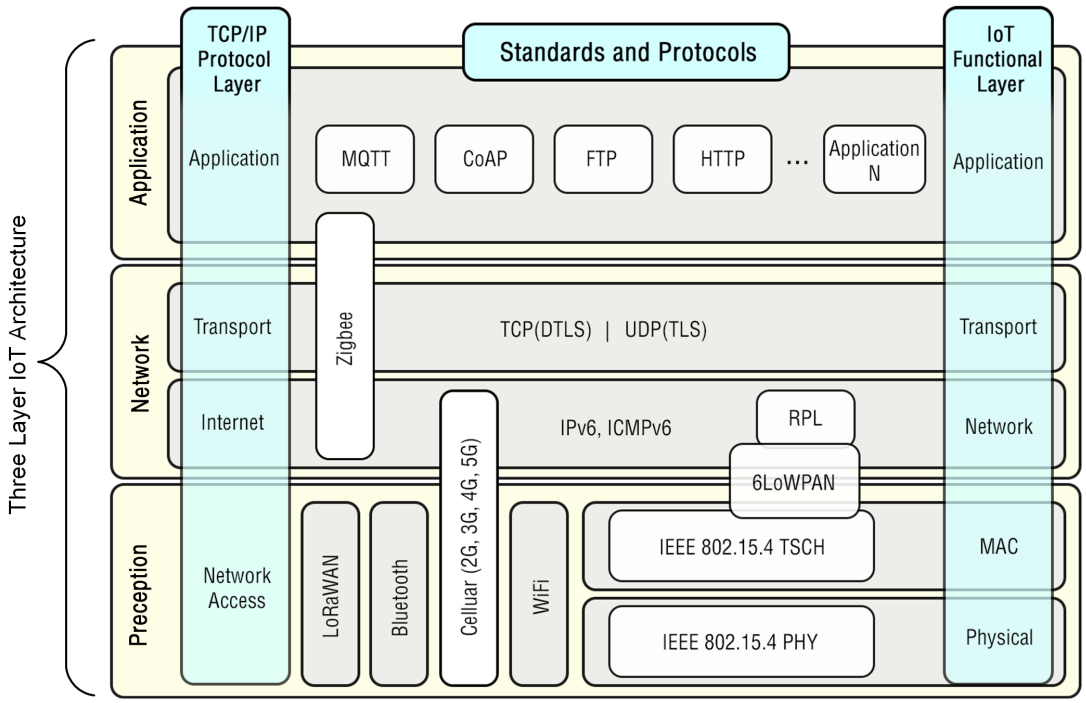


Figure 2.1: *IoT stack layer.*

(The parents are not the only ones responsible for this task.) The CORPL is designed for underwater communication scenarios, and unlike RPL, it does not support security and server technologies. Initially, no mobility was considered in the RPL network and all nodes were considered to be static. However, several researchers [22] have confirmed the possibility of placing mobile nodes in RPL.

2.2.2 DODAG

The RPL is capable of building several DODAG graphs [25], with identical roots in each graph characterised with different DODAG Id's. Each node is only permitted to join a single DODAG graph and be a child of a parent node at the same time; however, nodes with different roots and DODAGs can communicate with each other. A DODAG builds its graph in several steps, as represented in Fig. 2.2. The formation of the topology starts with the 6BR/Root, also referred to as the sink node. The root multicasts a DODAG Information Object (DIO) to all nodes in its neighbourhood to initiate the formation of a DODAG. A DIO packet carries essential information required by nodes to discover an RPL instance, learn configuration parameters, select a parent set, and maintain the DODAG graph.

Neighbouring nodes receiving the DIOs from the root choose the sender as the parent by replying with DAO (Destination Advertisement Object) messages. Next, the parent node may accept their request by sending DAO-ACK to each individual. The neighbouring nodes then calculate their ranks concerning the parents' rank value and other parameters and multicast a new DIO to the nodes in their neighbourhood for attracting potential children. Calculating the rank depends on several factors, such as the distance from the root, energy resource of the node etc. The node's rank identifies its position in the network topology, which is a

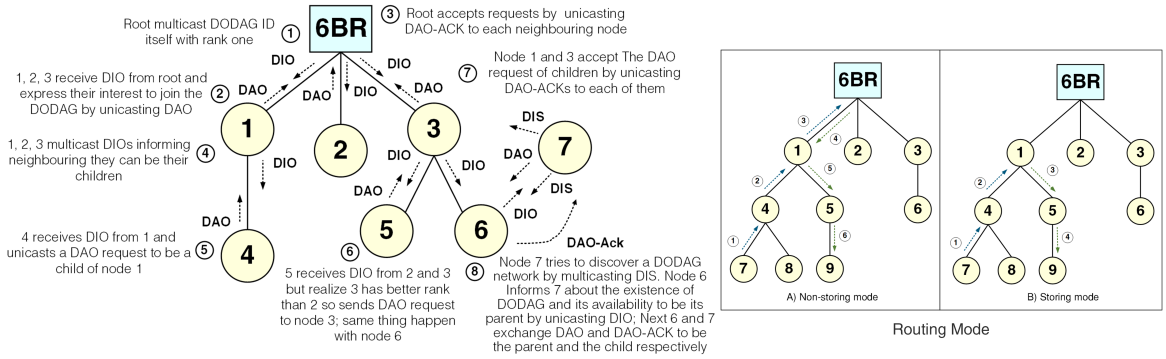


Figure 2.2: *DODAG graphing and RPL storing modes.*

top-down hierarchy. A child always has a higher, less valuable rank than its parents. IoT devices consider neighbours with a lower rank value as a parent. Optimal routes (parents, hops) in the DAG are obtained from metrics and constraints. In order to update the DAG, a DIO message is multicasted periodically according to the timer set by the border router (as part of the trickle algorithm).

Meanwhile, if any new node wants to join the DODAG, it will multicast DODAG Information Solicitation (DIS) requests to discover a DODAG network and listen for a DIO reply from a node in its neighbourhood. The DAO is intended to be used for creating a downward hierarchy. If a node loses connection with its parent, either it can wait for an incoming DIO message (taking 1-60 minutes) or send a DIS message [26].

If the parent node becomes unreachable or disappears, a couple of repair procedures are designed to avoid reconstructing the entire topology. The primary technique lets nodes send their packets through their neighbouring node with the same rank, and the second mechanism guides them to select another parent from the preferred parent set. IETF [1] also introduces a global repair mechanism to reconstruct the DODAG topology. Although such a mechanism can play an essential role in reviving an IoT network, it increases the vulnerability and enables malicious nodes to sabotage the network. Such attacks can exhaust battery-powered nodes, leading to shutdown.

The RPL IPv6 header option with the special flag 'O' indicates the intended packet direction, and 'R' notifies a rank error occurrence during packet forwarding between sender and receiver nodes. There are two downward routing modes, namely storing and non-storing mode, illustrated in Fig. 2.2. Each routing node is stateful in storing mode, and creates a downward routing table for its sub-DODAG to route incoming and outgoing traffic. In non-storing mode all nodes transfer their packets towards the border router/root, then the root node transfers the packet to the destination address.

2.2.3 Routing metrics

In the DODAG, the duty of configuring routing metrics, optimization objectives, rank calculation, and parent selection policy is defined by the OF (Objective Function) policy. The IETF proposed several OFs, using a variety of link attributes, for different applications and environmental conditions [27][28][29][30][31]. The OFs follow diverse policies with different goals. An OF may aim to enhance the packet end to end delay or preserve LLN nodes' energy

Table 2.2: *Objective functions*

Objective Functions	Routing Metric	Status	How does it work?
Objective Function Zero (OF0)	Hop count	Fully defined	It uses only the hop count as a routing metric. The preferred parent in OF0 is selected to be the neighbouring node with minimum rank. It adds the pre-determined value to the prior rank. The rank increases strictly from the node towards the sink monotonically. [36] The OF0 is the default OF of RPL
Minimum Rank With Hysteresis Objective Function (MRHOF)	ETX	Fully defined	The aim of this OF is to obtain the route that causes the least path cost. In this regard, either a path that satifies this need will be selected as the route, or hysteresis takes place as the second mechanism. The potential parent is nominated as a parent if its path cost - the threshold < the current parent path cost [37]. The Expected Transmission Count (ETX) is the default routing metric of MRHOF. The ETX pursues the discovery of routes that provide optimal end-to-end throughput.
ETX Objective Function (ETXOF) [38]	Hop count	Draft	This OF aims to discover the path providing a packet delivery with the minimum number of packet transmissions. The ETXOF is extensively implemented in a large number of IoT experiments, although it is an expired Internet draft.
Load Balancing Objective Function (LBOF)	Childset	Draft	In order to balance the number of children of each potential parent, and as a result, moderate the traffic in the network, the number of children is considered as a metric in this OF. This improves the packet delivery rate (PDR) and energy consumption in nodes.
Traffic Aware Objective Function (TAOF)	ETX + PTR	Draft	TAOF combines Packet Transmission Rate (PTR) and ETX to discover and nominate the path with minimum traffic toward the root.
LLQ OF	RSSI	Draft	This link quality depends on the distance between the nodes; it is based on the Received Signal Strength Indicator (RSSI) [39]

resources by avoiding routing through battery-powered nodes. Table 2.2 introduces existing, fully defined, or drafted OFs.

Energy calculation is not considered as an element in the routing path drawing of MRHOF and OF0. Several OFs have been proposed to address this limitation, namely Residual Energy OF, Energy Efficient and path Reliability Aware OF (ERAOF), Energy-Oriented Routing OF (OF-EOR), and Expected Lifetime OF (ELT). For brevity, this review paper does not review each OF. The reader is referred to [32] and [33] which provide comprehensive reviews of OFs and to [34] and [35] which study and analyse MRHOF and OF0 performance over several measures and LLN scenarios. Based on their findings, both OF0 and MRHOF cause long hops in a dense network with a large number of nodes, introducing an OF is essential.

2.2.4 Routing protocol vulnerabilities

The 6LoWPAN is vulnerable to various routing threats (such as Sinkhole, Version Number, Wormhole, etc) and does not have any concrete mechanism to ensure security in its routing protocol (RPL). [40] provides a comprehensive study and analysis of RPL performance in an extensive scale network. Studying RPL performance in a multi-hop network reveals the

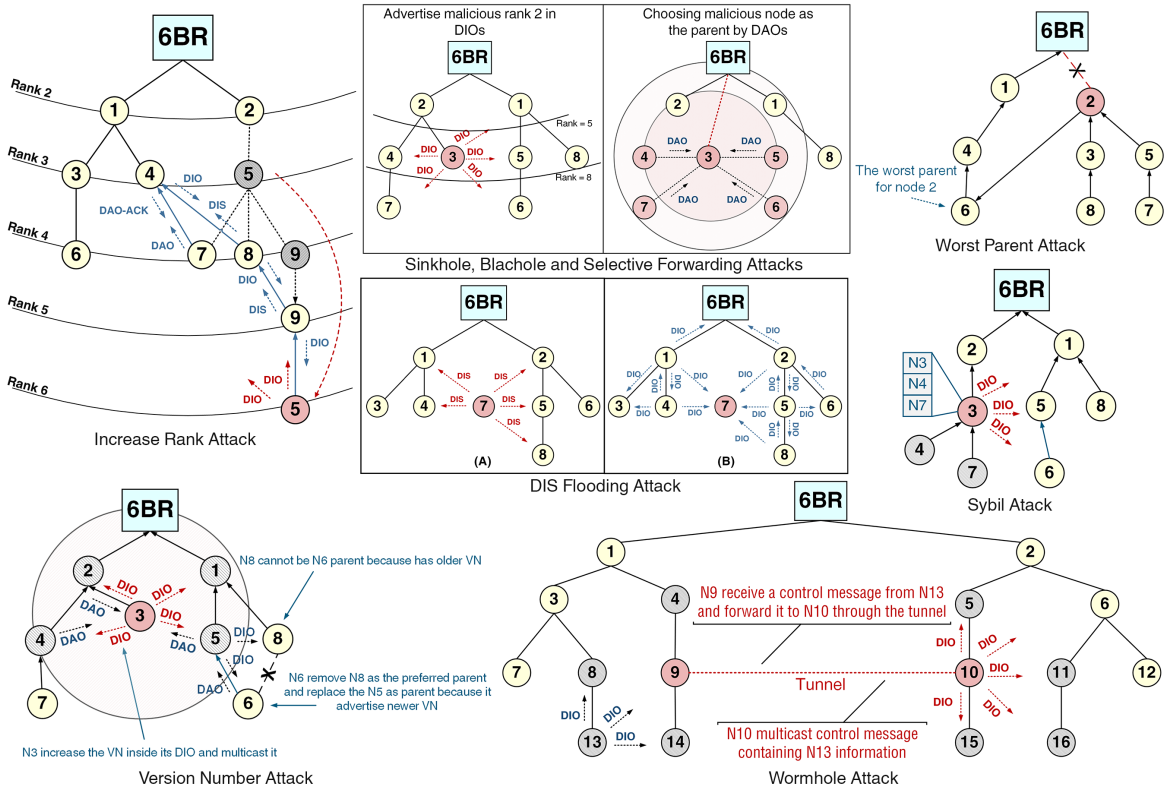


Figure 2.3: *Illustration of attacks in a RPL network .*

existence of link quality, energy exhaustion, information leakage, maintenance of routing information, integrity, and availability issues.

2.3 RPL Attacks

There are three types of RPL attacks, distinguishable by the harm they cause to the LLN [15], as illustrated in Fig. 2.3. If the attack is against the victim nodes' resources, then it falls into the resource attack category. This category consists of two subcategories, namely direct and indirect attacks. In a direct attack, the malicious node by itself establishes the attack while in the indirect category the intruder initiates the attack with the help of legitimate nodes. Both approaches aim to drain neighboring nodes' resources. However, detection of indirect attacks is more challenging because usually there is more than one attacker node present in the network and detection of their master, the primary intruder node, is harder since it does not target the LLN nodes directly. If the intruder aims to generate an unoptimised network topology, it is called a topology attack. Topology attacks divide into sub-optimisation and isolation subcategories. In the sub-optimisation attacks, the network diverges from optimal performance. In isolation attacks, targeted nodes become isolated from the network and cannot receive or transfer packets. The third category is against RPL network traffic, so it is called a traffic attack. This category divides into Eavesdropping (if the intruder node sniffs and analyses the network stream) and misappropriation (where the identity of other nodes

is stolen and used to advantage). In the following, we describe the prevalent RPL attacks in the framework mentioned above.

Intruders can obtain a malicious rank either by observing its neighbourhood and then advertising lower, more powerful, malicious rank values or by deceiving neighbouring nodes by manipulating the OF and persuading other nodes to assign a better rank to it. The OF manipulation enables the intruder to adapt to the changes in the network and work dynamically. This makes the detection of intruder nodes harder for the IDS [41]. Usually, the intruders combine attacks such as sinkhole and blackhole with selective forwarding in order to achieve their goals by making detection harder. Because such malicious activities aim to disturb the repair and routing mechanism of RPL, they are termed RPL attacks. Several researches [14], [42], [43], [44], [45], [41] analysed the adverse effect of different RPL attacks on the LLN. Watching each attack's symptoms over the LLNs can help researchers design a countermeasure by monitoring the affected parameters. Table 2.3 demonstrates the different negative impacts each RPL attack has on the network [14], [42], [43], [44], [45], [41].

Sinkhole (SH). A sinkhole attack is a sort of eavesdropping attack that sniffs the victims' data by persuading them to select the attacker node as the parent. This causes the construction of an unoptimised topology among the LLN nodes in the network. In Fig. 2.3, node 3 is the intruder and starts a sinkhole attack by multicasting DIO messages with a lower malicious rank of 2 while its legitimate rank is 7. As a result of this false advertisement, the neighbouring nodes express their interest in being children of the intruder by unicasting DAOs. Next, the victim node sends their packets to their parent, now the intruder node, to transfer the information to 6BR and other nodes. In the illustrated scenario, nodes 1 and 2 refuse the DIOs from node 3 because they already have rank 2, and node 8 is not affected by the malicious DIO message because it does not receive it. As a result of this falsified advertisement, the malicious node will more frequently be nominated as a preferred parent by its neighbours, while it does not provide a better performance based on the network Objective Function (OF). Algorithm 1 provides a pseducode of such an attack.

Blackhole (BH). The blackhole attack divides the LLN into isolated subnetworks, which cause an adverse effect on the network throughput. Similar to the SH attack, the intruder node attracts the neighbouring nodes by advertising a better malicious rank with DIOs, but instead of forwarding application packets, it drops all received information. A BH attack launched from a strategically chosen node can cause a massive loss in network traffic. A Selective-forwarding (SF) attack, also called a Grayhole attack, is a variant of this. Here the intruder selectively or randomly drops some of the packets and forwards the rest. Detecting this form of attack is more challenging.

Increase Rank (IR). This attack is against LLN nodes' resources and indirectly disrupts victim nodes to exhaust their computational and energy resources. It also causes a communication disruption in the LLN. The intruder initiates the attack by increasing its rank and multicasting DIO messages with the modified malicious higher (less valuable) rank value to its neighbouring nodes. By doing this, it forces the children to search and find a new parent to approach the border router. After causing significant network overhead, and when finally the children find a new parent, the intruder node reverts back to the previous rank or alternatively advertises a lower (better) rank to attract neighbouring nodes to reselect it as a parent. The process is illustrated in Algorithm 2; it repeats continuously until it exhausts the power resources of the victim nodes and forces them to shut down. In Fig. 2.3, the attacker

Algorithm 1: : Sinkhole, Black-hole, Selective_forwarding attacks

Initialisation
A: Attacker node
N: Neighbour list $\subset$ *legitimate LLN nodes*
B: a neighboring node $\in N$
P: Current packet
R: a lower more powerful rank, usually assigned as the sink node rank
Attack_type = {*Sinkhole, Black-hole, Selective_forwarding*}

Input: *"A"* receives *DIOs* from *A.N* and calculates *Min*(advertised_ranks)
Output: *"A"*obtains a lower, malicious rank and multi-casts it with *DIO* to all nodes, $\forall$ node $\in A.N$

if *(P is DIO)* $\wedge$ *(P.sender_id* $\in$ *A.N)* **then**
  if *P.sender_id* $\in$ *A.N* $\wedge$ *P.sender_id* $\neq$ *root_id* **then**
    if *DIO.rank* $\leq$ *A.malicious_rank* **then**
      A.malicious_rank $\longleftarrow$ *R*

if *(A.received(DIS from B))* $\vee$ *(A trickle_timer activated)* **then**
  A.multicast((DIO with malicious_rank) to node $\forall$*nodes* $\in$ *A.N)*
  B.receive(DIO from A)
  if *DIO.rank* $<$ *B.rank* **then**
    B nominate A as preferred_parent
  B unicast application packets to its preferred_parent, which is "A" now, in order to transfer it to the destination
  if *Attack_type is Sinkhole* **then**
    A collect packets from B and transfer it to next hop
  else if *Attack_type is Blackhole* **then**
    A collect packets from B then drop all of them
  else if *Attack_type is Selective_forwarding* **then**
    A collect packets from B and selectively or randomly drop some and transfer others to next hop

node 5, initiates an IR attack by increasing its rank to 6. As a result, its children nodes 7, 8, and 9 lose their connection to the border router and set about finding a new parent by multicasting DIS messages. Node 4 receives DISs from 7 and 8 and replies to their request by unicasting a DIO to them. Node 9 is not in the range of node 4 and sends DIOs periodically until nodes 7 and 8 join the DODAG and respond to its DIS with a DIO. When the intruder realizes that the victim has found a new route to the border router, it tries to re-attract them by advertising the original or better rank. This loop continues until all targets run out of power.

Wormhole (WH). This attack aims to disturb and obstruct the RPL topology by causing victim nodes to create unoptimised routes with regards to a falsified OF. This happens when two or more widely spaced attacker nodes, connected through a private channel or tunnel, over a wired or wireless connection established with the help of a powerful antenna mounted on the intruder nodes, dominate two parts of the network with their broad radio coverage. Algorithm 3 demonistrates the implementation of the Wormhole attack. Consider Fig. 2.3. Here the attack is initiated by the intruder node 9, transferring the collected control packets from its neighbourhood to its accessory, the intruder node 10, placed on another part

Algorithm 2: : Increase rank attack

Initialisation
A: Attacker node
N: Neighbour list
P: Current packet
R_1*: is the initial, legitimate rank*
R_2*: is a high, less valuable rank*

Input: *"A"* increases its rank to much higher rank and multi-casts it with *DIO*
Output: *"A"* receives a *DIO* containing a lower rank from a neighbouring node, then decreases its rank and multi-casts it with *DIO*

```
if (P = DIO) ∧ (P.sender_id = A.id) then
    if A.rank = R1 then
        A.rank ⟸ R2
    else if A.rank ≠ R1 then
        A.rank ⟸ R1
    A.Multicast(DIO to A.N)
if (P is DIO) ∧ (P.sender_id ∈ A.N) then
    if (P.Rank < A.Rank) ∧ (P.sender_id ≠ rootnode_id) then
        A.rank ⟸ R1
        A.Multicast(DIO, ∀ nodes ∈ A.N )
```

of the RPL network; the node 10 multicasts the received control packets and confuses the nearby nodes by making them believe that the generators of control packets (i.e. node 4, 8, 13 and 14) are in their neighbourhood. This action encourages the victims to add the initiators to their neighbour lists. The WH attack causes unoptimised route construction in the network topology and greatly increases network overheads. Several studies [44], [45] analyse the negative impact of the WH attack on LLNs. The WH attack can also be considered as an external intrusion if the attacker creates a tunnel between a node inside the RPL network and a device outside the LLN. The manipulated or malicious node inside the RPL network can be equipped with a more powerful antenna to transfer collected data to an external device outside the network. Due to the fact that in such attack, attackers use a private channel for transferring data, and the border router is not involved in transferring data, detection of an external wormhole attack is more sophisticated. As far as we are aware, there are no IDS proposals to detect external WH attack.

DIS Flooding (DF). This attack aims to exhaust target nodes' resources by generating a large amount of traffic in the victim network. This also disrupts communication among the LLN nodes. The DF attack significantly increases control packet overheads and energy consumption, and causes routing disruption. In the flooding attack, the intruder node can be placed inside or outside the network, and in the most extreme scenario, it succeeds in exhausting all targets' resources. As explained earlier in Section 2.2.2, a new node or the node that has lost its connection with its preferred parent and nodes in its parent list, uses a DIS message to discover a DODAG network in the RPL routing protocol. As in Algorithm 4, the intruder abuses the vulnerability of this method and multicasts a DIS message to the neighbouring node then listens for a DIO reply; this action repeats in order to drain resources and cause a considerable number of collisions in the network. Fig. 2.3 illustrates a DIS flooding attack scenario. In the scenario (A), node 7 is assigned as the intruder

Algorithm 3: : Wormhole attack

Initialisation
A_1, A_2 :*Attacker 1 and 2*
N: Neighbour list
B: a neighboring node $\in N$
P: Current packet
Control_Packet = {*DIO, DAO, DIS, DAO-Ack*}
Input: *B Multi-casts Control_Packet to nodes,* $\forall$ *node* $\in B.N$
Output: $A_1 or A_2$ *transfer the received Control_Packet from B to its counterpart*
if *(P is Control_Packet)* **then**
  if $P.source\_id \in A_1.N$ **then**
    A_1.transfer(P to A_2)
    A_2.multicast(P, $\forall$ node $\in A_2$.N)
  else if $P.source\_id \in A_2.N$ **then**
    A_2.transfer(P to A_1)
    A_1.multicast(P, $\forall$ node $\in A_1$.N)

and establishes the flooding attack by multicasting DISs to its neighbor list. As a result, its neighbours reply to its request by multicasting DIOs not only to node 7 but to their neighbourlist (B). This RPL attack causes network congestion and the saturation of the LLN nodes. It increases control packet overheads considerably. In the multicast DIS Flooding attack, the victim node (the receiver node here) will reset its trickle timer and multicast its DIO message when its receives a multicasted DIS message from the intruder. In a unicast DIS flooding attack, the receiver node of unicasted DIS message unicasts a DIO to the intruder without resetting its trickle timer. Since it is not required to be part of the DODAG to send DIS control packets, an intruder can initiate the DIS flooding attack outside of the network [15].

Clone Id (CI) & Sybil Attacks. Both of these attacks are inherited from WSNs. In the Clone ID attack, the intruder node clones or takes the identity (MAC address, IP address, rank, etc.) of a victim node, then multicasts or unicasts packets to its neighbors to disrupt the network and threaten confidentiality and integrity of the targeted node data. On the other hand, in the Sybil attack, the intruder aims to disturb a vast number of nodes by stealing the identity of several nodes. The intruder then multicasts and unicasts the control packets of targeted nodes simultaneously. The placement of intruder node(s) in a Sybil attack affects the degree of negative impact on the network; this has been studied in [46], [47], [48]. The intruder node(s) can manipulate data by bonding to an area and disturbing a smaller quantity of nodes by stealing their identity and collecting their data; the process is presented in Algorithm 5. They are also capable of scaling the attack domain by influencing nodes in different locations to impact a larger proportion of network. The aim in this kind of placement is to damage the routing mechanism and make the ranking system or OF ineffective. Detection of distributed attack scenarios is harder if malicious nodes are mobile. The study [48], describes various types of Sybil attacks.

Algorithm 4: DIS Flooding attack

Initialisation
A: Attacker node
N: Neighbour list
I: Current node id
B: a neighboring node $\in N$
V: Victim list
P: Current_packet
Attack_type = {*Unicast DIS Flooding, Multicast DIS Flooding*}
Control_Packet = {*DIO, DAO, DIS, DAO-Ack*}

Input: *"A"* uni-casts or multi-casts *DIS* to node(s), $\forall$ nodes $\in A.N$
Output: *"B"* uni-casts or multi-casts *DIO* message
if *A.Attack_type is Unicast DIS Flooding* **then**
    A.unicast(DIS $\Longrightarrow$ *B,* $B \in A.N$*)*
    B.unicast(DIO $\Longrightarrow$ *A)*
else if *A.Attack_type is Multicast DIS Flooding* **then**
    A.Multicast(DIS, $\forall B \in A.N$*)*
    for $\forall B \in$ `A.N` **do**
        B.Multicast(DIO, $\forall node \in B.N$*)*

Algorithm 5: Sybil Attack

Initialisation *A: Attacker node N: Neighbour list P: Current packet L: Target List S: sender node Control_Packet* = {*DIO, DIS* } *Attack_Types* = {*Sybil, Clone_Id*}

Input: Control_packet initiated by victim node(s)
Output: *"A"* steals victim nodes credentials, then uni-casts or multi-casts control packets with their identities
if *(P* $\in$ *Control_Packet)* $\wedge$ *(P.sender_id* $\in$ *A.N)* **then**
    //attacker can select the victim(s) selectively or target its children
    if *(S.node_id* $\in$ *A.children_list)* **then**
        if *(Attack_Type = Sybil)* **then**
            if *(S.node_id* $\in$ *A.L)* **then**
                A.clone(S.credential)
            A.Multicast(Control_Packet $\Longrightarrow$ *A.N)* $\vee$ *A.Unicast(Control_Packet* $\Longrightarrow$ *A.N[node_id])*
        else if *(L.length = 1* $\wedge$ *Attack_Type = Clone_Id)* **then**
            A.clone(S.credential)
            A.Multicast(Control_Packet $\Longrightarrow$ *A.N)* $\vee$ *A.Unicast(Control_Packet* $\Longrightarrow$ *A.N[node_id])*

Worst Parent Attack (WP). As in Algorithm 6, the intruder selects the worst parent to transfer data [41] while multicasting its actual rank with DIOs. The idea behind this attack is to cause lengthy end-to-end delays and create an unoptimized path from children nodes to the 6BR. Detection of this attack is more challenging than other attacks because the intruder does not show any abnormal attitude through multicasting control packets; however, if the intruder decides to impact larger nodes and attract more children by advertising lower rank, detection becomes more feasible. No current study covers this attack.

DODAG Inconsistency. The intruder can either multicast malicious control packets with enabled 'O' and 'R' flags in the opposite direction or can alter the received packet from a

Algorithm 6: Worst Parent Attack

Initialisation *A: Attacker node N: Neighbour list P: Current packet Control_Packet = {DIO, DIS, DAO, DAO-Ack }*

Input: *"A"* discovers the neighbouring node with the highest, least valuable rank, providing the worst *OF*

Output: *"A"* selects the discovered worst parent as the preferred parent to reduce routing performance

if *(P = DIO) ∧ (P.sender_id ∈ A.N)* **then**

  if *(A.preferred_parent[rank] < Node.Rank, ∀ Nodes ∈ A.N)* **then**

    A.preferred_parent = Node.id

Algorithm 7: Version Number attack

Initialisation
A: Attacker node
N: Neighbour list
I: Current node id
B: a node
P: Current packet
M_VN: Malicious Version Number
Control_Packet: {DIO, DAO, DIS, DAO-Ack}

Input: *"A"* collects DIO from the root or a neighboring node and reads the current version number

Output: *"A"* increases the current version number and puts it in *(M_VN)* and advertise it through multi-casting DIO

if *(P = DIO ∧ P.destination_id = A.id)* **then**

  if *(P.version_number = 1) ∨ (P.version_number ≤ M_VN)* **then**

    A.DIO[version_number] ⟸ M_VN
    A.multicast(DIO, ∀B ∈ A.N)
    M_VN ⟸ M_VN ++

  else if *(P.version_number > M_VN)* **then**

    A trigger repair mechanism of DODAG and recalculate its rank
    M_VN ⟸P.DIO[version_number] ++
    A.DIO[version_number] ⟸ M_VN
    A.multicast(DIO)

neighboring node and enable its 'O' and 'R' flags before forwarding it to the destination. This causes significant control packet and energy overheads and increases packet delivery time. If the intruder decides to manipulate the received packet before transferring it, the detection becomes harder. It causes isolation in the network because the receiver node always drops the packet and initiates the repair mechanism.

Version Number Attack (VN). Since there is no built-in security mechanism to ensure that only the root node can modify the value of a DODAG version number in a LLN, the intruder can abuse this vulnerability to cause an adverse impact on the functionality of a DODAG. In a VN attack (Algorithm 7), the intruder node incrementally increases the repair mechanism value and then advertises it through its DIO message. This encourages LLN nodes to enable the global repair procedure and recalculate their routing paths more

frequently. VN causes significant energy overheads in LLN nodes while exhausting their computational resources. This attack can become much more sophisticated if the intruder node is far from the 6BR, i.e., in the lower level of the DODAG hierarchy. In [42], [43], the authors analyse the negative impact of the VN attack on LLNs.

Ranks Attack (RA). This attack is harder to be detected because the intruder node does not initiate any malicious packet or manipulate any legitimate packet. It disregards the rank error initiated by its neighbouring or child node [41]. The DODAG allows only an increase of the rank in a downward direction and decreases in upward direction, as illustrated in Fig. 2.2. The nodes have to check the rank condition when sending and receiving packets. If LLN nodes find any error in this procedure, they have to enable the rank-error bit defined in the RPL protocol and inform neighboring nodes about inconsistency in the network; this prevents the formation of a loop in the network. In the rank attack, the intruder does not enable the rank-error bit when it discovers a rank error. This attack is difficult to detect because the intruder does not display any abnormal behavior (e.g., it satisfies all protocol conventions, except honesty). In the long run, this malicious behavior causes the formation of a loop in the network, damaging the network topology. Moreover, it isolates the nodes with a rank error in the network and results a massive number of error packets and inconsistencies in the routing mechanism. It [41] analyses the impact of the rank attack on LLNs.

Local Repair Attack (LR). The intruder initiates this attack by sending a repair packet to the node in its neighbourlist while there is no inconsistency or error in the network [49]. This results in computational exhaustion of LLN nodes and an increase in control packet overheads in the network, because victim nodes have to recalculate their routes to the malicious node.

Replay Attack. The intruder records legitimate control packets, such as DIO, DAO, DIS, generated by its neighboring nodes, and then later it unicasts or multicasts the collected packets. This causes inconsistency and creation of expired routing paths in the network because some configurations in advertised control packets are outdated and cause the network to function erroneously. Algorithm 8 represents such an attack. Because the intruder forwards the collected control packets from legitimate nodes, built-in security mechanisms of RPL and the use of cryptography cannot prevent it [13].

Even RPL secure mode and cryptography cannot secure the LLN against such intrusions because knowing the keys is not required for an intruder to replay collected packets. The consequence of this attack is discussed in Table 2.3. The intruder replays the application packet in the replay attack for the WSN platform, while in the RPL, the intruder replays control messages only.

DIO Suppression attack (DS). The authors of [50] study the DIO suppression vulnerability of RPL and analyse its adverse impacts on LLNs. In the DS attack, the intruder advertises a DIO frequently in order to slow down the DIO message process. Neighbouring nodes of the attacker consider the received DIO consistent after collecting enough similar DIO messages from the malicious node. This leads victim nodes to suppress their DIO multicasting process, which in turn leads to the isolation of some LLN nodes since they cannot discover their neighbouring nodes, and some routes that are providing better OF will remain undiscovered. A study and analysis of the consequences of a DIO suppression attack in LLN can be found in [51]. It also proposes a mitigation method [51].

DAO Inconsistency attack (DI). In RPL, the forwarding-error flag is designed to

Algorithm 8: Replay Attack

Initialisation *A: Attacker node N: Neighbour list P: Current packet L: Target List R: List of recorded control packets Control_Packet = {DIO, DIS, DAO, DAO-Ack }*

Input: *"A" records Control_Packet initiated by "L"*
Output: *"A" multi-casts R*
if *(P ∈ Control_Packet) ∧ (P.sender_id ∈ A.N)* **then**
  //attacker can select the victim(s) selectively or target its children
  if *(P.sender_id ∈ A.L)* **then**
    Add(R ⟸ P)
  if *(Attack_triggertimer.status = Activated)* **then**
    A.multicast(R, ∀ nodes ∈ A.N)

indicate that the stored path in the routing table of the parent is no longer valid and needs to be removed. This is done by enabling 'F' flag in the option header of the received packet and replaying it to the parent. In RPL storing-mode, the intruder exploits the vulnerability of this mechanism to initiate a DAO inconsistency attack. Upon receiving a packet sourced from an ancestor of the intruder node, the intruder enables the 'F' flag of the received packet and replays it to its parent to claim that the indicated downward route in the packet is no longer available. This misleads the parent into removing the legitimate downward route from its routing table. As a result, the parent node also has to inform its parents that the destination node is no longer available when it receives a packet that wants to use the expired route. The authors of [52] study the impact of this attack on LLNs.

2.4 Intrusion Detection Systems

Security infrastructures such as encryption may perform well in securing 6LoWPAN against external intrusions but they are computationally expensive [52; 105; 100] for LLN nodes and cannot make RPL resilient in the face of internal malicious activities [106; 60]. However, Intrusion Detection Systems (IDSs) show outstanding performance with acceptable energy overhead for detection of internal and external intrusions. The structure of IDS for 6LoWPAN can be classified along several axes, namely the source of monitoring data, analysis type, detection strategy, monitoring technique, the form of response, and detection time. Next, each criterion is discussed in detail and the relevant proposed methods categorised. Fig. 2.6 gives a taxonomy of IDS for RPL and Table 2.6 shows the IDS approaches employed by researchers.

Before classifying IDSs, we define what an IDS is. In recent years we have seen inconsistency in the definitions of IDS in RPL. The IDS is the software or hardware designed to monitor and analyze the events taking place inside the host machine, or packets sniffed through the network traffic, in order to discover any suspicious activities and raise an alarm. An IDS does not have any mitigation duty. On the other hand, an Intrusion Prevention System (IPS) can work with an IDS to detect suspicious behavior.

Although IPS can autonomously prevent intrusions, security administrators sometimes may prefer to implement IDS rather than IPS. Moreover, detecting suspicious activities (via an IDS) or mitigating the effects of an intrusion (using an IPS) it is often desired simply to understand the situation better. For example, administrators like to discover the aim

Table 2.3: *RPL attacks and their impacts on LLNs*

Attack name	**Type**	Against									**Researches**
		D1	**D2**	**D3**	**D4**	**D5**	**D6**	**D7**	**D8**	**D9**	
Sinkhole	Insider	N	L	M	L	M	Y	L	N	C/I	[49], [53], [54], [55], [22], [56], [57], [58], [59], [60], [61], [62], [63], [64], [65], [66], [67], [68]
Blackhole	Insider	N	H	M	L	M	Y	H	Y	C/I/A	[69], [70], [59], [62], [71], [72], [73], [74], [75]
Selective forwarding	Insider	N	M	M	L	M	Y	M	N	C/I/A	[76], [54], [22], [58], [59], [60], [77], [62], [63], [71], [78], [74]
Wormhole	Insider/Outsider	N	M	H	H	M	Y	L	N	C/I	[79], [56], [80], [81], [82], [83]
Rank attack	Insider	Y	H	M	M	M	Y	H	Y	A	[49], [25], [61]
Version number	Insider	Y	M	H	H	H	Y	H	Y	C/I/A	[84], [85], [57], [62], [86], [87]
Increase rank	Insider	Y	M	H	H	H	Y	H	N	A	-
DIS flooding (Unicast)	Insider/Outsider	N	N	M	H	H	N	H	N	A	[88], [89], [90], [85], [56], [57], [59], [91], [87], [92]
DIS flooding (Multicast)	Insider/Outsider	N	N	H	H	H	N	H	N	A	[88], [89], [90], [85], [56], [57], [59], [91], [87], [92]
DoS	Outsider	N	N	H	H	H	N	H	N	A	[93], [94], [95]
Neighbour attack	Insider	Y	L	M	M	L	Y	L	N	C/I/A	[49], [90]
Local Repair	Insider	N	H	H	N	H	N	M	N	C/I	[25], [49], [59], [96]
OF Manipulation	Insider	N	L	N	N	L	Y	L	N	C/I	[61]
DIO suppression	Insider	N	L	M	M	H	Y	M	Y	A	-
DAO Inconsistency	Insider	N	H	M	L	H	Y	M	Y	C/I/A	[97], [52], [98]
DODAG Inconsistency	Insider	N	N	H	H	H	Y	H	Y	I/A	[70], [99], [100]
Sybil	Insider	N	H	H	M	M	Y	M	Y	C/I	[46], [101], [102], [103], [59], [62], [48], [83]
Clone Id	Insider/Outsider	N	L	L	M	L	Y	L	Y	C/I	[59], [64], [82]
Replay attack	Insider	N	H	M	M	M	H	H	Y	C/I/A	[51], [104]
Worst Parent Attacks	Insider	Y	M	N	N	N	Y	H	N	C/I	-
Sniffing	Insider/Outsider	N	L	M	L	L	Y	L	N	C	-

ROUTING LOOP (D1), NEGATIVE IMPACT ON PDR (D2), CONTROL PACKET OVERHEAD (D3), NETWORK CONGESTION (D4), DRAIN BATTERY (D5), UNOPTIMISED ROUTE (D6), DELAY(D7), NODE ISOLATION (D8), CONFIDENTIALITY (C) INTERGRITY (I) AVAILIBILITY (A) (D9), LOW (L), MEDIUM (M), HIGH (H), YES (Y) , NO (N)

and identity of intruders by tracing the path of attackers seeking information. This may be achieved, for example, by using a honeypot or a variety of situational awareness tools.

2.4.1 Source of monitoring data

The source of data for monitoring can be defined with regards to the type of intrusion the IDS aims to counter. It may aim to secure the IoT network against attacks manipulating the content of the application layer, such as SQL injection, bruteforce, or side-channel attacks. In this case, monitoring audit-logs, system events of the client machine, or in some scenarios, the payload contents of network packets after decryption plays a vital role in detecting intrusions. On the other hand, the attacks that alter the parameters of a legitimate network packet or generate malicious packets require the appliance of network security infrastructure such as an IDS to monitor and analyse network traffic. Therefore, the IDS obtains network-packets and audit-logs of the host machine, or both, for monitoring purposes.

Network-based IDS (NIDS): Since RPL is in the network layer of the IoT stack, detecting RPL attacks requires analysing network packets. The NIDS analyses the flow of network traffic in the LLN. Researchers commonly use NIDS for detecting RPL based intrusions. However, NIDS cannot analyse the encrypted contents of packets' payloads without possessing the encryption key. The NIDS monitors the network traffic either through several monitoring agents placed among LLN nodes, or each LLN node is required to participate in the monitoring task, as discussed in Section 2.5. The advantages and disadvantages of NIDS in 6LoWPAN are given below. NIDS are widely used by researchers in this domain because they can monitor 6LoWPAN on a large scale. NIDS operates in hidden mode, also called ghost-mode, and is concealed from the eyes of intruders; therefore, attackers cannot probe them in order to compromise them [59].

Additionally, NIDS can function in passive mode and cause less energy and computational overhead for LLN nodes. This also leads to less disruption in network traffic and less congestion and dropped packets. A strategically placed probe can monitor an extensive network. However, centralized NIDS are very likely to face difficulties in dealing with volumes of incoming data from an extensive scale network, especially if the assigned monitoring node has resource constrained LLN devices. They may miss incoming attacks during periods of high traffic. Secondly, NIDS cannot analyse the encrypted content in sniffed packets' payloads. Finally, the network communication between the central IDS and the sensors in the active decentralized, hybrid IDS generates a very significant control packet overhead, leading to network congestion.

Table 2.4: *Strengths of Monitoring techniques and detection strategies in IDS*

Strength	Monitoring Techniques					Architecture		
	ACIDS	**ADIDS**	**AHIDS**	**PCIDS**	**PDIDS**	**NIDS**	**HIDS**	**Hybrid**
Provide host machine information	✓	✓	✓	-	-	-	✓	✓
Work in promiscuous mode	-	-	-	✓	✓	✓	-	-
Global analysis	✓	-	✓	✓	✓	✓	-	✓
Local analysis, lower level of hierarchy	-	✓	✓	✓	✓	✓	-	✓
Provide the participating IDS agents with main power line and better computational resources	✓	-	-	✓	-	-	-	-

ACIDS: Active Centralised IDS, ADIDS: Active Decentralised IDS, AHIDS: Active Hybrid IDS
PCIDS: Passive Centralised IDS, PDIDS: Passive Decentralised IDS

Host-based IDS (HIDS): The HIDS, in its traditional meaning, is designed to monitor and analyse not only the network inputs and outputs of the host machine but also the internal system events that are taking place inside the host machine. It monitors system

Table 2.5: *Weaknesses of Monitoring techniques and detection strategies in IDS*

Weakness	Monitoring Techniques					Architecture		
	ACIDS	**ADIDS**	**AHIDS**	**PCIDS**	**PDIDS**	**NIDS**	**HIDS**	**Hybrid**
Consume LLN's Computational resources	✓	✓	✓	-	-	-	✓	✓
Drain LLNs' Energy resources	✓	✓	✓	-	-	-	✓	✓
Integrity problem	✓	✓	✓	-	-	-	✓	✓
Require additional equipment and increase financial cost	-	-	-	-	✓	✓	-	✓
Increase control packet overhead dramatically	✓	✓	✓	-	-	-	-	✓
Vulnerable to external intrusions	-	✓	-	-	-	-	✓	-
Cannot monitor nodes in lower level of hierarchy	✓	-	-	✓	-	N/C	✓	-
Suffer from Single Point of Failure (SPoF)	✓	-	-	✓	-	-	✓	-

ACIDS: Active Centralised IDS, ADIDS: Active Decentralised IDS, AHIDS: Active Hybrid IDS
PCIDS: Passive Centralised IDS, PDIDS: Passive Decentralised IDS

logs and events to identify suspicious activities. Because HIDSs are hosted in LLN nodes, they may place very significant demands on the computational and energy resources of the host machine. As mentioned earlier, there is no use of HIDS in its traditional definition for detecting RPL attacks; all researchers employ NIDS to counter such attacks. However, the use of HIDS is essential, especially for attacks manipulating IoT in the application layer and when analysing encrypted content of packet's payload is required. In the RPL domain the proposed IDS is sometimes called host-based by researchers if LLN nodes are required to send their device information in terms of geographical location, RSSI (Received Signal Strength Indication), routing table, neighboring node information etc, to an IDS or an internal IDS of the node. Researchers typically categorize this IDS as being one of active monitoring.

Hybrid IDS: The IDS is called hybrid, in terms of the data source, if both HIDS and NIDS security mechanisms are incorporated in a network, among LLN nodes, to monitor the network events that are taking place from different perspectives. Although this technique provides the IDS management system with a broader monitoring oversight of the 6LoWPAN and secures the network against a more extensive range of malicious activities threatening different stack layers of IoT, there is no research that covers this type of IDS yet.

2.4.2 Detection strategy

There are two main approaches to the analysis of events for detecting attacks [107]: detection of malicious signatures and detection of anomalies. Signature detection is broadly employed by most security software companies in the market. Anomaly-based IDS has attracted researchers over many years. A third approach, specification-based IDS, compares behaviours against reference behaviours defined more formally, e.g. by protocol specifications. Below we describe each of the proposed methods.

Signature-based: The signature-based IDS, also known as misuse-based, compares the collected data against the already stored signatures of malicious software to identify abnormal activities. This type of IDS relies on stored patterns of known intrusions, collected by experts through real-world experience, and empirical or simulation experiments. A low False Positive Rate (FPR) is a major strength of misuse-based IDS, while this detection strategy is unable to detect unknown intrusions. Intrusions are the ones not stored in the system profile/database,

because it is either a zero-day attack or one whose signature is otherwise not yet included. So this category of IDS only performs well over known intrusions and shows poor performance over unknown attacks. Continual updating of the database is needed. Also, signature-based IDS demands significant storage space, which is scarce in IoT nodes; researchers usually place such IDS in the 6BR or at the edge.

Anomaly-based: Unlike misuse-based IDS, an anomaly-based IDS adapts to normal activities and highlights any deviation from the system's normal behavioural profile. This enables anomaly IDS to detect unknown attacks. It does so through statistical, knowledge-based, or machine-learning methods. However, such IDS is known for having a considerable FPR; that is because lots of normal activities are not considered or have been missed in the profile-building/training phase. This form of IDS constructs a profile of normal activities across nodes in the LLN. The anomaly-based IDS requires less storage compared to the misuse-based IDS, but consumes more processing power, especially in the training period [22]. Additionally, determining what is normal requires a comprehensive dataset of legitimate activities and also requires a long adaptation period.

Specification-based: This IDS uses a defined notion of normal behavior and highlights any deviation from it. However, and unlike the anomaly-based approach, expert manual assistance is typically required to define the specification of the normal profile. (This may take the form of a protocol specification for example.) This strategy is widely used by researchers on account of its small storage requirement and reasonable FPR and FNR performance and requiring no training period. Furthermore, according to [49], this approach is well-suited to detecting topology or rank-based attacks in RPL networks. However, specification-based IDS cannot update its normal profile when the network topology changes or when there is an increase or decrease in the number of nodes. Manual updates to the specification will be needed.

Hybrid: To circumvent the shortcomings of the detection strategies mentioned above, researchers have sought to combine the detection strategies to produce hybrid IDSs to monitor the network. A hybrid IDS typically provides a better detection rate and performance at the expense of greater resource (computation and energy) consumption. Researchers seek practical trade-offs between accuracy and LLN node resource exhaustion.

2.4.3 Response

An IDS generally seeks to detect intrusions. Once detected, a decision needs to be made as to how to respond. We can categorize responses into two major groups:

Passive response: Here, the security administrator or the system users will be informed about the occurrence of abnormal activities. No automated corrective action is taken as a result. The 'response' must be manually invoked.

Active response: Here, the response is automatic and takes place when specific categories of attacks are noticed. Active intrusion detection systems log and notify the security administrator in the same way as passive ones do, but they can also take extra actions to counter the intrusion. For instance, they may alter Access Control Lists (ACLs) on a firewall in order to terminate malevolent traffic, block processes on the server subject to the intrusion, or guide the intruder to a trap or "safe environment" created by security administrators.

Table 2.6: *State-of-the-art ids techniques*

Paper	ACIDS	ADIDS	AHIDS	PCIDS	PDIDS	NIDS	HIDS	Hybrid	Misuse	Anomaly	Specification	Hybrid	Response
[92]	-	-	-	-	✓	✓	-	-	✓	-	-	-	P
[25]	-	✓	-	-	-	✓	-	-	-	-	✓	-	P
[108]	-	-	-	✓	-	✓	-	-	-	-	✓	-	AC
[49]	-	✓	-	-	-	✓	-	-	-	-	✓	-	P
[84]	-	-	-	-	✓	✓	-	-	-	-	✓	-	AC
[53]	-	-	✓	-	-	✓	-	-	-	✓	-	-	AC
[79]	-	-	✓	-	-	-	-	-	-	✓	-	-	P
[76]	-	-	✓	-	-	✓	-	-	-	✓	-	-	AC
[93]	-	-	✓	-	-	✓	-	-	-	-	-	✓	P
[69]	-	✓	-	-	-	✓	-	-	-	-	✓	-	AC
[101]	-	-	✓	-	-	✓	-	-	-	-	threshold	-	AC (N/A)
[102]	-	-	-	-	-	-	-	-	-	-	-	-	-
[70]	-	✓	-	-	-	✓	-	-	-	-	threshold	-	AP
[89]	-	✓	-	-	-	✓	-	-	-	-	threshold	-	AP
[97]	-	✓	-	-	-	✓	-	-	-	-	threshold	-	AP
[90]	-	✓	-	-	-	✓	-	-	-	threshold	-	-	P
[54]	-	-	✓	-	-	✓	-	-	-	-	-	✓	P
[55]	-	✓	-	-	-	✓	-	-	-	-	threshold	-	AC
[22]	-	-	✓	-	-	✓	-	-	-	-	-	✓	AC
[8]	-	-	-	✓	-	✓	-	-	-	-	-	✓	P
[85]	-	-	-	-	✓	✓	-	-	✓	-	-	-	AC
[56]	-	-	-	✓	-	✓	-	-	-	-	-	✓	P
[57]	N/A	N/A	N/A	N/A	N/A	✓	-	-	-	✓	-	-	P
[94]	-	-	-	-	✓	✓	-	-	✓	-	-	-	P
[99]	-	-	-	-	✓	✓	-	-	-	-	✓	-	P
[80]	-	-	✓	-	-	✓	-	-	-	-	threshold	-	P
[58]	-	✓	-	-	-	✓	-	-	-	-	✓	-	AC
[109]	-	-	✓	-	-	-	✓	-	-	-	-	✓	-
[103]	✓	-	-	-	-	✓	-	-	-	-	Trust-based	-	AC
[110]	✓	-	-	-	-	✓	-	-	-	-	Trust-based	-	P
[100]	-	✓	-	-	-	✓	-	-	-	-	threshold	-	AC
[52]	-	✓	-	-	-	✓	-	-	-	-	threshold	-	AC
[59]	-	-	-	✓	-	✓	-	-	✓	-	-	-	P
[60]	-	-	✓	-	-	✓	-	-	-	-	threshold	ü	AC
[61]	-	-	✓	-	-	✓	-	-	-	-	✓	-	AC
[77]	✓	-	-	-	-	✓	-	-	-	-	Trust-based	-	P
[87]	-	-	-	✓	-	✓	-	-	-	-	Rule-based	-	P
[81]	✓	-	-	-	-	✓	-	-	-	-	-	✓	AC
[88]	-	-	-	✓	-	-	✓	-	-	✓	-	-	P
[111]	-	-	-	✓	-	✓	-	-	-	✓	-	-	AC
[112]	-	✓	-	-	-	✓	-	-	-	-	-	✓	P
[105]	✓	-	-	-	-	-	✓	-	-	✓	-	-	AC
[113]	-	-	✓	-	-	✓	-	-	✓	-	-	-	P
[62]	✓	-	-	-	-	✓	-	-	-	-	-	✓	P
[71]	-	-	✓	-	-	✓	-	-	-	-	Trust-based	-	AC
[63]	✓	-	-	-	-	✓	-	-	-	-	-	✓	P
[64]	✓	-	-	-	-	✓	-	-	-	-	threshold	-	P
[114]	-	-	✓	-	-	-	-	✓	-	-	-	✓	P
[115]	-	-	-	-	✓	✓	-	-	-	-	-	✓	AP
[48]	-	-	✓	-	-	✓	-	-	-	-	threshold	-	AC
[96]	-	✓	-	-	-	✓	-	-	-	-	✓	-	P
[116]	-	✓	-	-	-	✓	-	-	-	✓	-	-	AP
[72]	-	-	✓	-	-	✓	-	-	-	-	✓	-	AC
[73]	-	-	-	✓	-	✓	-	-	-	-	-	✓	P
[91]	-	-	-	-	✓	✓	-	-	-	-	threshold	-	AC
[65]	✓	-	-	-	-	✓	-	-	-	-	✓	-	AC
[66]	✓	-	-	-	-	✓	-	-	-	-	✓	-	AC
[82]	✓	-	-	-	-	✓	-	-	-	threshold	-	-	AP
[117]	-	-	✓	-	-	✓	-	-	-	-	-	✓	AC
[67]	-	-	-	-	✓	✓	-	-	-	-	✓	-	AC
[68]	✓	-	-	-	-	✓	-	-	-	-	✓	-	P
[78]	✓	-	-	-	-	✓	-	-	-	-	✓	-	AC
[118]	-	-	✓	-	-	✓	-	-	-	-	✓	-	P
[98]	-	-	✓	-	-	✓	-	-	-	✓	-	-	AC
[95]	-	-	✓	-	-	✓	-	-	-	✓	-	-	P
[86]	✓	-	-	-	-	✓	-	-	-	-	✓	-	AC
[74]	✓	-	-	-	-	✓	-	-	-	-	threshold	-	P
[75]	-	-	✓	-	-	✓	-	-	-	-	✓	-	AC
[51]	-	✓	-	-	-	✓	-	-	-	-	✓	-	AC
[83]	-	-	✓	-	-	✓	-	-	-	-	✓	-	P
[104]	-	✓	-	-	-	✓	-	-	-	-	✓	-	AC

RESPOND TYPES: ACTIVE PROACTIVE (AP), ACTIVE CORRECTIVE (AC), PASSIVE (P)

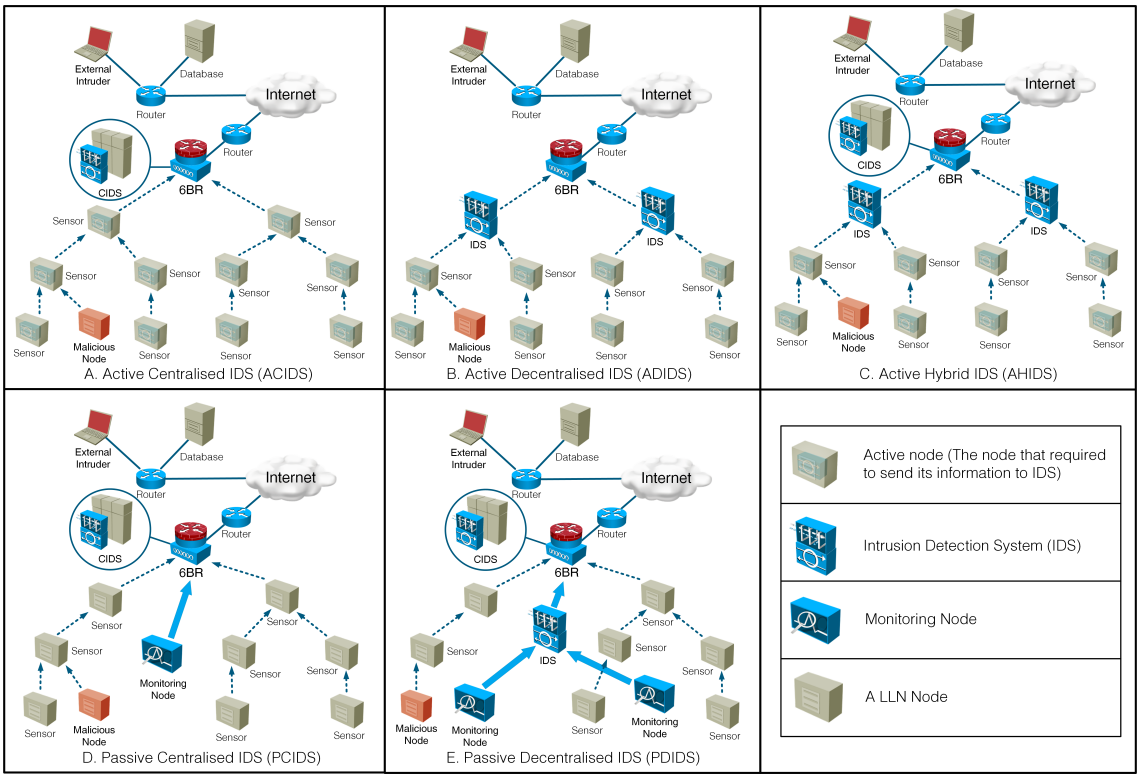


Figure 2.4: *Monitoring techniques.*

2.4.4 Machine-learning-based IDSs

Another division of IDSs is based on Machine Learning (ML) techniques, where ML algorithms are applied to develop intrusion detection engines. ML-based IDS (ML-IDS) can be classified according to the type of supervision they receive through the training phase (supervised, unsupervised, semi-supervised, and reinforcement learning). Supervised learning methods aim to discover the relation among observations (independent variables) and the ground truth (dependent variable). The formed relations are represented in a structure called a model. The ML-based IDS model intends to provide an understanding of malicious behaviour. The majority of the existing learning methods are based on inductive learning. The implicit assumption of the inductive learning approach is to generate/acquire a generalised model which can be applied to future unseen observations. More formally, given observation (X, Y) where $X = x_1, x_2 \ldots x_n$ is a vector of independent features and Y is the dependent variable (ground truth), a supervised classifier aims to develop a predictor $f(x)$ to predict y_t given X_t, or $y_t = f(X_t)$. The correctness of ground truth has a direct impact on the classifier accuracy. Various supervised ML algorithms are employed to develop ML-IDSs in the literature [119]; namely Decision Tree (DT), Random Forest (RF), Logistic Regression (LR), K-Nearest Neighbours (KNN), and Support Vector Machines (SVM). On the other hand, unsupervised learning extracts relations among the independent variables X to draw its model without having access to a pre-specified, dependent attribute. The unsupervised learning approach is based on data interpretation, focusing on extracting the hidden relations within unlabelled data. Clustering is regarded as an unsupervised learning;

that uncovers hidden patterns in data by grouping similar observations into clusters. In the semi-supervised learning, the classifier has partial access to the ground truth; for instance in One-class SVM the classifier only trains over legitimate activities [120] to identify outliers. Reinforcement learning develops an rational agent capable of interacting with its environment and consequently take optimal actions. An agent learns to take the optimal action through receiving a feedback signal (referred to as a positive or negative rewards) from its environment. We can further classify the ML-IDS into two categories i.e incremental/online or batch-trained/offline. Next we describe each of these categories in detail.

Batch-trained ML-IDS

In batch learning (also referred as offline learning), the predictor model should train over a given finite batch of training data (where the data distribution is unknown, yet stationary) and can only be used when the training is 'completed'. This learning approach is known to be very time consuming and computationally intensive. In this learning approach, the training is carried out in offline mode, after data collection. After the training phase, the classifier runs the developed model, and it does not have any automatic learning procedure (it is offline). On the break out of new concept (new intrusions or new network traffic patterns), the developed classifier must train over the entire training data (including new observations and old data) to update its model. Next, it needs to discard the old model and execute the recently developed classifier. Although this method can work efficiently for various ML tasks, for the streaming data environment of 6LoWPAN (with continuous flow of data), we need more reactive classifiers that can adjust to new threats on the fly. Furthermore, in a resource-constrained environment, we need more computationally resource-efficient methods that strategically use memory space, CPU, and network resources to learn a significant volume of computer network traffic autonomously. Batch learning methods cannot help with this.

Incremental/Online ML-IDS

In offline/batch learning, the model can be used after the training phase is completed. Whereas in online/incremental learning, the model does not have access to the complete data stream; hence the model needs to create a prediction model incrementally. It becomes crucial to be able to extract essential information from vast, fast-paced datastreams; highlighting the need for data stream mining. In 6LoWPAN, IDS observes a considerable (unbounded) volume of data as a continuous flow, hence it is required to be adaptive to change rapidly and autonomously [121]. Hence, an incremental model is essential in this domain. Once an incremental ML classifier has learned new observations, it can discard them from its memory [122]. This mechanism can save significant amount of memory space for IDS. In incremental learning, the classifier trains over observations sequentially (individually or in small batches). Every training round means to be computationally efficient to facilitate system adaptation to new data (concepts) on the fly. The incremental classifiers update their model batch-by-batch on the arrival of new data. As a result, they are capable of processing the evolving data streams.

Concept Drift: The streaming data classification task involves the classification of data in a streaming environment, where data is not available to the system beforehand and will be observed sequentially (one-by-one). Hence the classifier needs to manage the incoming

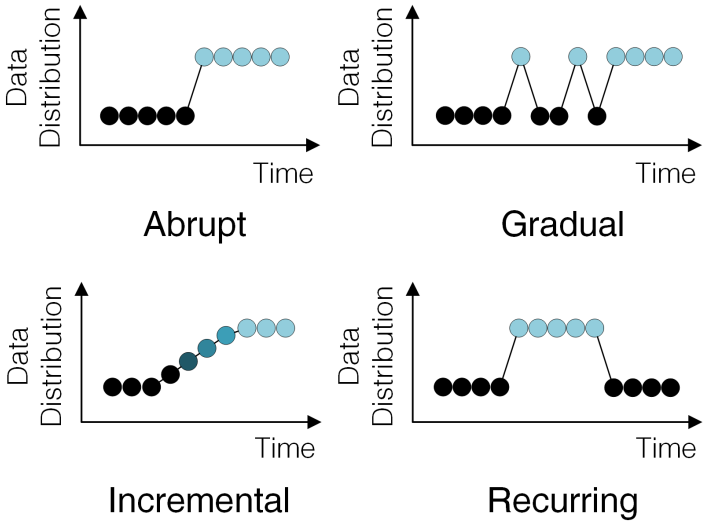


Figure 2.5: *Concept drifts.*

observations on the fly. Aside from the complexity imposed by the streaming data, there is another issue of unanticipated changes in data distribution. This phenomenon is known as concept-drift [121]. In offline learning, the ML-based classifier trains over a finite number of instances in dataset $\mathcal{D}$, where the $\mathcal{D}$ distribution is stationary and unknown to the system. The rate of concept drift is unknown to the system and can be abrupt, incremental, gradual or recurring [121], shown in Fig. 2.5. The abrupt drift is a sudden change in data distribution. In the gradual drift, the previous window instances appear infrequently compared to recently observed instances in the prevailing window. In the incremental drift, concepts evolve slowly. In contrast to incremental drift, the concepts systematically recur in the recurring concept drift. Identifying the concept drift rate is vital for any online (adaptive) IDS model. Adaptive IDS model refers to an IDS with a learning mechanism that can update its detection model online (in real-time) during IDS operation to react to concept-drifts [121].

Concept-drift Detection: In [121], concept-drift detection approaches are classified into three categories: Statistical-based; Windows-based; and Sequential-Analysis-based approaches. The statistical-based algorithms (e.g. drift detection method and early drift detection method) analyse statistical parameters (e.g. the mean and the standard deviation) of the prediction results to identify concept drift occurrence. In contrast to statistical-based algorithms, windows-based algorithms (e.g. adaptive windowing and drift detection methods based on Hoeffding's bound) employ a reference window for summarising the past observations and a sliding window for summarising the latest observations. A significant difference between distributions of these windows shows concept drift occurrence. There is yet another concept-drift detection approach referred to as Sequential Analysis based approach, where a detector incrementally assesses prediction outcomes and identify concept drift when a pre-defined threshold is reached.

Reinforcement Learning: Reinforcement learning (RL) [123] is a framework to solve sequential decision-making tasks under uncertainty to maximise a long term benefit through trial and error. RL lies between supervised and unsupervised learning approaches. The

learner is at the centre of an RL problem, which is called the agent in RL terminology. We mostly have a single agent in the RL framework; however, if there is more than one agent, the RL approach is called multi-agent RL (a.k.a MARL), where agents may have collaborative or competitive relations. An agent aims to learn what action to take in the world that it lives in, known as the environment. The environment is everything surrounding the agent. A set of observations representing the situation in the environment is known as a state. The agent observes the state and takes action based on its observation. The agent interacts with the environment by taking action. As a result, the agent receives a reward signal from the environment, and the environment transitions into a new state. The reward is the environment's feedback regarding the agent's action. The agent aims to acquire a long-term and highest reward. In other words, the agent's objective is to get the best sequence of actions that optimise the expected sum of the future rewards without knowing the dynamics of the environment. In RL, exploration plays a vital role in making the agent find the most efficient answer for a given task. By this, the agent can achieve the ultimate reward at the end. In this regard, the agent and the environment form a cyclic path of information in which the agent takes action, and the environment returns a feedback signal. The process of reaching the ultimate reward can be formalised using Markov Decision Processes (MDP), where we can model the transition probabilities for each state. There are two standard solutions to achieve objective functions, Q-learning and policy learning. The former learns based on the action-value function, while the latter uses the policy function that maps the best action and the corresponding state.

2.5 Monitoring Techniques

Earlier IDSs dedicated a solitary monitoring node to analyse and watch the events either in a hosted device or a specific network. This is called Centralised IDS (CIDS). In the RPL network the CIDS is usually placed at the 6BR because it incurs lower energy and computational overheads compared with LLN nodes. CIDS is prone to highly sophisticated and distributed intrusions and Single Point of Failure (SPoF). That is because the computational power of 6BR may sometimes be overwhelmed and cause a considerable proportion of incoming network traffic not to be analysed. To address CIDS issues the Distributed IDS (DIDS) carries out data monitoring and/or IDS detection tasks at several locations. Although DIDS is a better candidate for computer networks, demanding 6LoWPAN network nodes participate in monitoring and detection tasks causes very significant network overheads. Researchers have considered different placements of DIDS in LLNs to balance the number of agents in a way that covers a reasonable number of nodes. In the most computationally expensive scenarios the monitoring and detection duty is spread across all nodes. One of the most effective distributed placements of IDSs, a clustering placement that divides the LLN into clusters with cluster heads with various combinations of tasks among nodes and heads, is discussed in detail in [124]. The placement of IDSs and their monitoring nodes plays an essential role in reducing network overheads, saving energy resources, reducing FAR, and increasing the detection rate of attacks. Fig. 2.4 depicts existing monitoring techniques for IDS in LLN.

Although the 6BR has sufficient hardware resources to carry out heavy computation and host a comprehensive IDS, communication between LLN nodes and the 6BR results in very significant overheads on the network. Placing IDS agents on the sensors can reduce the control

packet overhead associated with network monitoring. However, such LLN sensors have limited resources and IDS computation may drain their computational resources (processing, storage, ROM, and energy). Placing IDS agents across dedicated nodes can reduce monitoring traffic, provide us with more processing capacity, and enable the IDS to monitor a wider area.

The IDS can be placed at various locations in the IoT network, such as sinknode/6BR, predefined devices, or all nodes. Nodes that host IDS can have different responsibilities. In the distributed IDS the nodes can be responsible for monitoring neighbouring sensors. A node that is responsible for monitoring its neighbours is usually referred to as a watchdog. The centralised IDS is placed at an individual node and works alone. In an IoT network it typically is placed on the border router or a dedicated host. Since the border router is the bridge between LLNs and outside world placing the IDS in the 6BR allows monitoring and analysis of the internal and external traffic to the 6LoWPAN network [22].

Nevertheless, analysing traffic between LLNs and the Internet that traverses the border router is not enough to secure the network because it cannot watch the activities that are taking place among the nodes unless they are near the 6BR. Additionally, the centralised IDS may have difficulty monitoring compromised nodes.The IDS monitoring technique divides into two categories called Active and Passive monitoring, whether the LLN node participates in the monitoring tasks or not. Table 2.4 and Table 2.5 give the pros and cons of each monitoring technique.

2.5.1 Active monitoring

In this kind of monitoring, the LLN nodes are responsible for monitoring tasks. The monitoring tasks can be transferring packets or gathering monitored information, and analysing them. This monitoring technique divides into three subcategories: centralised, decentralised or hybrid.

Centralised monitoring: In active centralised monitoring, a single central unit is responsible for analysing and judging the collected packets. Meanwhile, the rest of the nodes need to monitor, capture, and store the data and transfer them to the Central Manager unit (CM). The CM node aggregates received data and analyses it. Usually, the CM has better computational hardware resources than other nodes in the RPL network. It can be a local server or manifest itself as a cloud-based service. This type of IDS works well over small scale networks. However, in larger-scale networks the CM is more likely to face route congestion and suffer from significant overheads and SPoF.

Decentralised monitoring: This type of monitoring is similar to a centralised approach where each node still has responsibility for packet collection and transportation. However, unlike active centralised IDS, the distributed nodes are usually router nodes or cluster heads and need to perform decision making tasks. Therefore, there would be reduced load on the LLN nodes in the network compared with centralised monitoring. Although decentralised monitoring conserves nodes' hardware resources better than a centralised one, it still places significant computational and energy demands on LLN resources.

Hybrid monitoring: In an active hybrid approach, both the CM and distributed nodes share responsibility for monitoring and decision-making in the network. However, LLN nodes must still collect and transfer their information to IDS agent nodes and so there may be computational exhaustion of LLN nodes' resources.

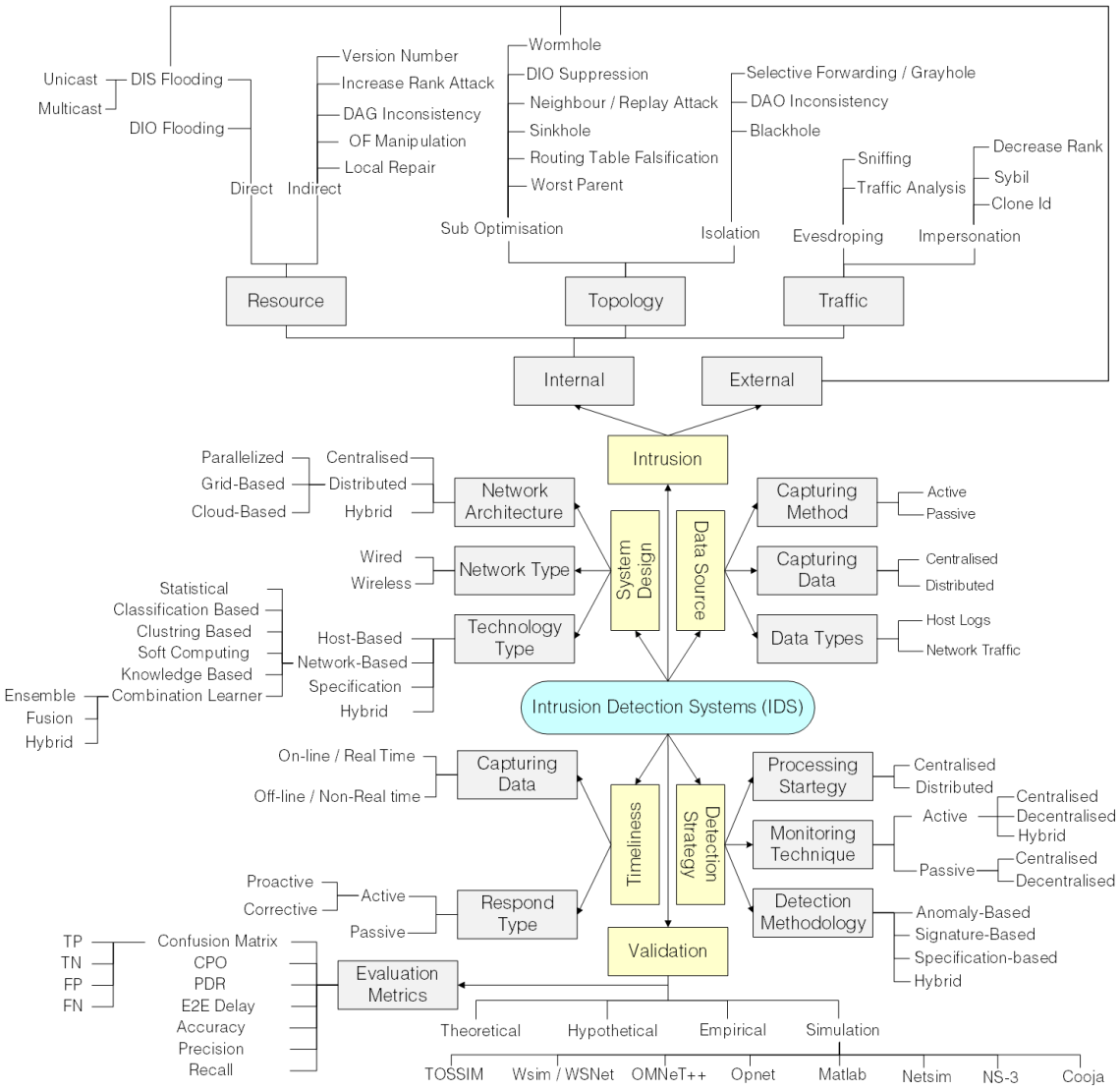


Figure 2.6: *IDS taxonomy.*

2.5.2 Passive monitoring

In this approach, monitoring nodes (sniffers) are assigned in the 6LoWPAN to sniff and collect control packets from their neighborhood. They are responsible for collecting information about nodes and events occurring in the target network. Passive monitoring employs centralised and decentralised approaches, as described below.

Centralised: In this approach, the monitoring nodes passively listen to the communications in the network, then collect the data before sending it to the sink node, which is responsible for analysing and decision-making. PCIDS is capable of conducting a more in-depth analysis of the collected data remotely, on the edge or cloud, where more computational resources are available; however, this results in a delay for attack detection.

Decentralised: In this approach, the central management unit and several monitoring

nodes are responsible for monitoring tasks like data aggregation and analysis. Several monitoring nodes can be placed in the network to do data collection, and aggregation tasks. The sniffers can be involved in sending the collected data from their neighbouring nodes to the monitoring nodes. Next, monitoring nodes can perform data aggregation before forwarding information to the sink node for deeper analysis. In this way, the target node gets analysed from both local and global perspectives.

2.6 Evaluation

2.6.1 Evaluation approaches

Researchers use different IDS evaluation approaches [125], as discussed below.

Simulation: This strategy is the most widely used approach for IDS evaluation in this domain. Here, researchers either validate their method against a dataset generated through simulation of several normal and attack scenarios or implement their proposed algorithm in the simulator and validate its performance at run-time using different evaluation metrics. There are several pros and cons to using this evaluation method. The main advantage of simulation is its low cost, low implementation effort, and compressed experimental time (i.e. simulated time is far quicker than real-time). The majority of simulators in this field are open-source and implementing a large number of nodes does not significantly increase the project cost. Moreover, the time taken to implement and test ideas can be drastically decreased compared to empirical approaches. However, simulation outcomes may be less reliable than those of empirical evaluation.

Empirical: This approach collects the evidence through an experimental network setup. It is considered as the most reliable approach for evaluating any proposed system. However due to high economic costs, effort and time barriers, the implementation and use of an extensive heterogeneous IoT network in a wide geographical area for a long duration is not feasible using this evaluation method.

Theoretical: a solid theoretical argument to support research outcomes. This includes relating a model to attribute properties it is intended to represent [125].

Hypothetical: This evaluation strategy is used when the applicability of the proposed method in practise is not clearly specified.

No evaluation: There are several researches in this field that did not provide any evidence for their proposed methods. This is the most unreliable approach for evaluation.

2.6.2 Evaluation metrics

Researchers use several metrics to measure the performance of their proposed methods [126]. It is common to measure the accuracy and effectiveness of the proposed IDS in classifying malicious and normal packets. One of the most comprehensive ways to calculate the performance of a classifier is the confusion matrix, illustrated in Fig. 2.7. It summarises four aspects of binary classification: the numbers of True Positives (TP), True Negatives (TN), False Positives (FP), and False Negatives (FN). Each source event is classified as either an attack or normal. The positive is the intrusion class, and negative is the normal one. Most studies seek to minimize the False Positive Rate (FPR) and False Negative Rate (FNR). Both false classification of malicious activity as normal (FN) and the false classification of normal

packets as malicious (FP) incur costs. In contrast, the correct classification of intrusions (TP) and normal activities (TN) incurs no cost other than the costs of deploying the IDS (C0) (Fig 2.7, B). Reducing the FPR is generally considered easier than reducing the FNR. Of course, the FNR is significantly sensitive to inability to detect unknown intrusions.

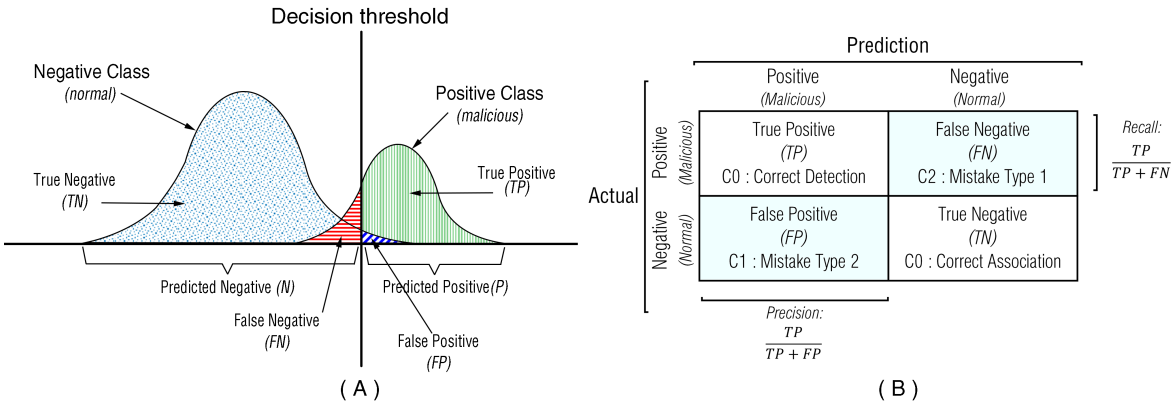


Figure 2.7: *Decision threshold and confusion matrix.*

In Fig. 2.7, FNs and FPs have different negative consequences. A considerable number of FPs causes system management to waste time and can lead to loss of confidence. A high FN indicates that the IDS is failing to perform the primary task it was designed for. The FN rate, equation 2.1, is usually higher than the FP rate [127], equation 2.2. This is because normal instances (Majority class) usually significantly outnumber malicious ones (Minority classes); hence, the classifier can classify normal instances more accurately, while classifying malicious instances is more challenging because of the lack of training instances.

$$FNR = \frac{FN}{FN + TP} \tag{2.1}$$

$$FPR = \frac{FP}{FP + TN} \tag{2.2}$$

An ROC curve (or Receiver Operating Characteristic curve) is a plot that summarises the performance of a binary classification model on the positive class. The x-axis indicates the FPR and the y-axis indicates the TPR.

The Packet Delivery Ratio (PDR), equation 2.3, is the ratio between the total number of application packets received ($P_{received}$) by the final destination nodes and the total number of application packets sent (P_{sent}) by senders.

$$PDR = \frac{\sum_{i=1}^{n} P_{received_i}}{\sum_{j=1}^{n} P_{sent_j}} \tag{2.3}$$

The Detection Rate (DR) or Recall, equation 2.4, is another widely used metric in this field. It declares how and in what measure the IDS succeeds in detecting the attacks.

$$Recall = DR = TPR = \frac{TP}{TP + FN} \tag{2.4}$$

The Control Packet Overhead (CPO) is the total number of DODAG control packets (DIO, DAO, and DIS) initiated by each node, equation 2.5. In order to calculate the power consumption of a node, researchers use equation 2.6, which is the sum of total energy consumed by the machine and the network (Energy consumption) divided by the elapsed time in seconds.

$$CPO = \sum_{i=1}^{n} (DODAGControlPacket)_i \tag{2.5}$$

$$Power\ Consumption = \frac{\text{Energy consumed(mJ)}}{\text{Time(s)}} \tag{2.6}$$

The End to End (E2E) delay gives the average time elapsed when transferring a packet from a source to its destination, equation 2.7.

$$E2E\ Delay = \frac{\sum_{i=1}^{n} d_i}{\text{n}} \tag{2.7}$$

Accuracy, given in Equation 2.8, is the fraction of all events that are correctly classified (either as malicious or normal). Precision, given in Equation 2.9, is the fraction of all positive classifications (i.e. alarms) that are correct. Precision is focused on positive classifications whilst accuracy considers both positive and negative classifications. F1-measure, given in Equation 2.10, combines precision and recall into a single measure that captures both properties and provides the classifier performance. It provides a way to display precision and recall concerns with a single score.

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{2.8}$$

$$Precision = \frac{TP}{TP + FP} \tag{2.9}$$

$$F1\text{-}measure\ (\beta = 1) = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{2.10}$$

In a streaming data environment, the number of instances for each class may evolve and change. The Kappa [128; 121] statistic is a measure for evaluating the prediction performance of incremental classifiers. The kappa-statistic is computed as Equation 2.11, where Θ is the accuracy rate of an intelligent classifier and Θ_r is the accuracy rate of a random classifier, which randomly permutes the predictions of the intelligent classifier. The kappa-statistic takes values between 0 and 1, where 0 indicates that the achieved accuracy is random.

$$Kappa = \frac{\Theta - \Theta_r}{1 - \Theta_r} \tag{2.11}$$

2.7 Discussion

Our study reviews 103 papers in order to answer the questions posed in Section 2.1.1. The results provided by researchers are considered as the basis for evaluating and comparing their proposed methods. Justifying the correctness and trustworthiness of the provided results

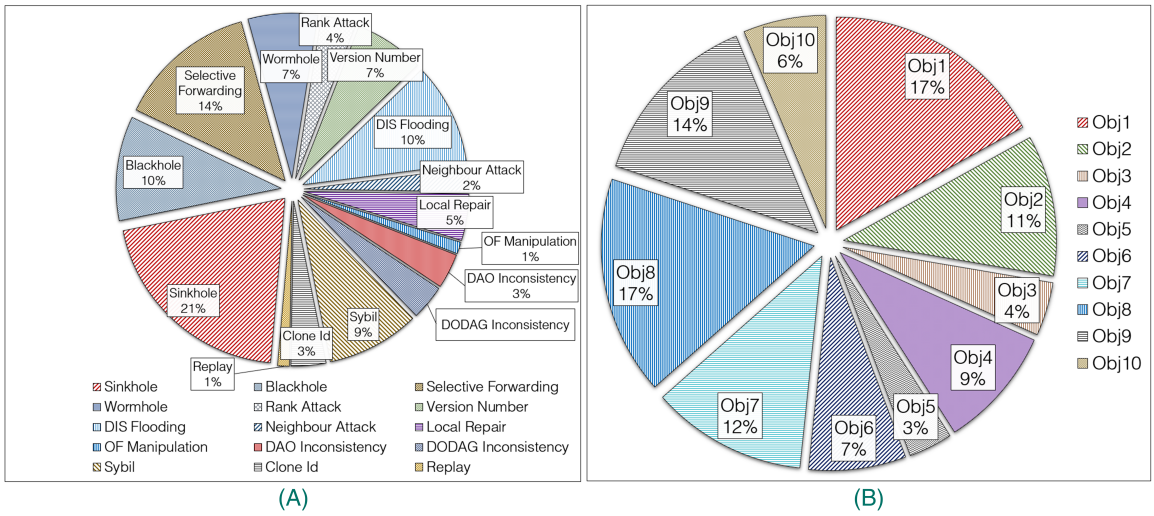


Figure 2.8: *Detected attacks (A) and research objectives (B) proportions.*

claimed by authors is not the aim of this study. Table 2.7 and , 2.8 summarise the results provided in each piece of research using the evaluation metrics discussed in section 2.6.2. Studying and analysing Tables 2.3, 2.6, 2.7, and 2.8 help us to answer each question in turn.

2.7.1 To what extent are RPL attacks addressed so far (Q 1)

In Section 2.4, we introduced and described a comprehensive set of known RPL attacks. Fig. 2.8.A illustrates to what extent each RPL attack has been addressed so far, based on Table 2.7 and 2.8 data. The extracted information shows that the proposals mostly concentrate on addressing sinkhole, selective forwarding, DIS flooding, and blackhole attacks, with 21%, 14%, 10%, and 10% of papers, respectively. The rest of the attacks constitute less than half of the researches' attention, 45% in total. There are two explanations; either the dominant attacks are the most disruptive malicious activities that are harming LLN, or the less considered attacks are less easily detected. Hence, there is a significant need for research to detect all intrusions or concentrate more on those receiving little attention. Our survey did not find any study proposing an IDS to detect Worst Parent, External Wormhole, OF Manipulation Attacks. Also, very few propose IDS to detect Replay, DODAG inconsistency, DAO inconsistency, Neighbour attacks, and Rank attacks. No comprehensive study in this field detect all types of RPL attack. Because some RPL attacks are similar in nature, the ideal IDS should be able not only to detect the occurrence of attacks but also identify the type of attack accurately and identify intruder nodes correctly.

2.7.2 Negative impact of each RPL attack (Q2)

Studying the proposed methods enables us to determine to what degree each RPL attacks cause abnormality in 6LoWPAN. Discovering the adverse impact level of each RPL attack requires an in-depth analysis of each intrusion over several LLN scenarios, which is accomplished by the already reviewed researches mentioned in Section 2.3. Table 2.3 shows the negative impact of each attack from different perspectives that are scaled with regards to

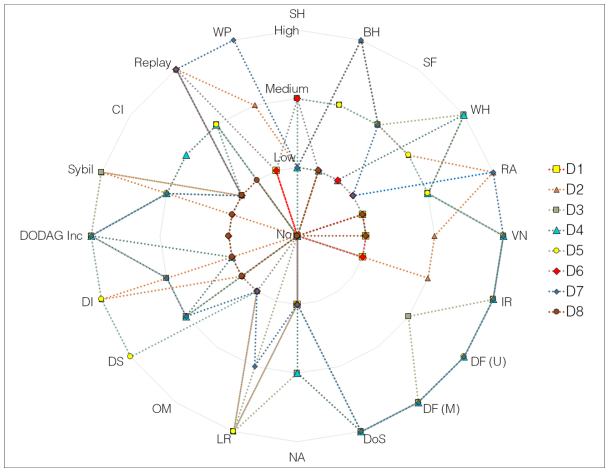


Figure 2.9: *The adverse effects of RPL attacks on LLN*

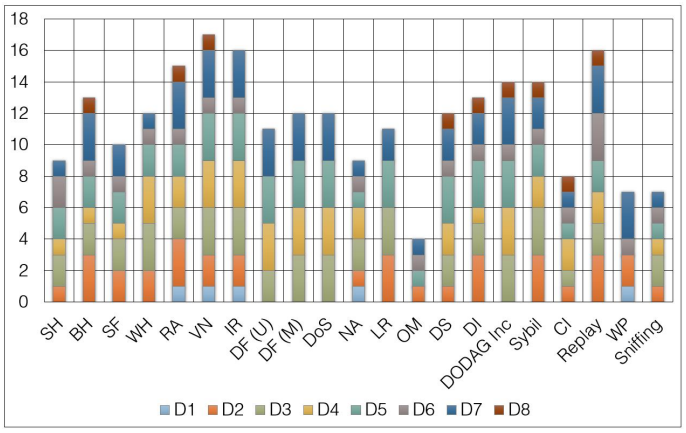


Figure 2.10: *Negative impact level of each RPL attack.*

terminology used in the studied paper. This answers Q2 and helps researchers to concentrate more on the most destructive malicious activities. Table 2.3, Fig. 2.9 and Fig. 2.10 reveal that the version number attack is the most disruptive intrusion on the LLN, while OF manipulation and worst parent attacks cause the least disruption (which makes their detection harder). Analysing the affected parameters would help researchers to detect such attacks. Each RPL attack manipulates and harms the target network in various aspects with different strengths; therefore, precise and accurate algorithms are required for IDS to not only detect the occurrence of attack but also to classify the type of intrusion and to distinguish the intruder node correctly.

2.7.3 Technical performance objectives (Q 3)

The primary stated objectives of the reviewed IDS approaches are to achieve the high TPR/Detection Rate (Obj1), low energy consumption overheads (Obj2), low Control Packet Overhead (Obj3), low FAR(Obj4), ability to protect networks that have mobile nodes (Obj5),

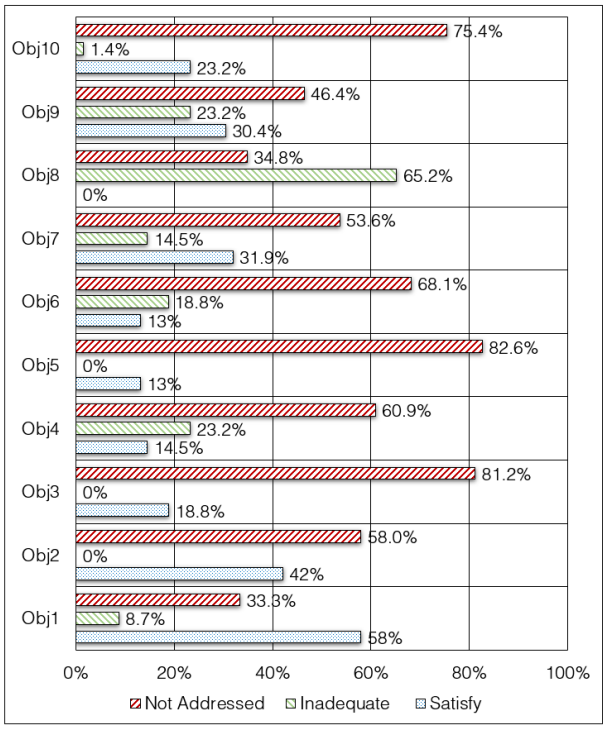


Figure 2.11: *Research objectives.*

provide mitigation for multiple attacks (Obj6), evaluation over many and heterogeneous networks (Obj7), resilience against unknown intrusions (Obj8), providing intrusion prevention mechanisms (Obj9), and high PDR (Obj10). There are multiple objectives that can be used for evaluation purposes. Many pieces of research address only one or a small number of these and provide no information on the others. We cannot assume that the implemented systems perform well on objectives that have not been formally evaluated, and so the practical applicability of such IDS in real-world networks is doubtful. Often researchers call their method comprehensive in all terms without providing sufficient evidence to prove their claim. Table 2.9 and Fig. 2.8.B show the objective of each study based on statements and the results provided in their paper. In order to discover the minimum, maximum, first quartile, mean, median, third quartile value of each detection and monitoring technique used by researchers, we analyse the provided results in Tables 2.7 and 2.8 and extract essential information shown in Fig. 2.12 and Fig. 2.13. With regards to extracted data we illustrate to what extent researchers satisfy the objectives, in Fig. 2.11. Enhancing the detection rate (Objective 1) is the primary aim of 17% of the reviewed papers. Although 58% of researches satisfied the requirement of this objective, 8.7% could not fully answer this need, and 33.3% did not address this essential requirement of IDS. This study discovers that 65.2% of proposed methods would be able to detect unknown intrusions (Objective 8) with the attention of 23% of papers.

2.7.4 Monitoring techniques implementation proportion (Q 4)

76.8% of the proposed methods use active monitoring systems and the remaining 23.2% use passive monitoring, using Fig. 2.14.A. Active monitoring techniques enable the researcher to

Table 2.7: *State-of-the-art research outcomes on ids in low power and lossy network (LLN)*

Paper	PDR	Power Consumption	Accuracy	Precision	Recall / TPR	ROC	FPR	FNR	CPO
[92]	-	-	-	-	N/C	-	-	-	N/C
[25]	-	-	-	-	-	-	-	-	-
[108]	-	2,65,256 mJ	-	-	80∼93%	-	12%	-	-
[49]	-	6.3% increase	-	-	100	-	5.92∼6.78	2.64∼6.78	-
[53]	79%∼95%	-	-	-	92%	-	∼3%	8%	-
[84]	-	-	-	-	-	-	1∼20%	-	-
[79]	-	329 mJ, Energy Overhead provided	-	-	AD:94% IND:87%	-	-	-	193
[76]	-	-	-	-	100%	-	-	-	High
[93]	-	-	-	-	∼90%	-	5%	-	-
[69]	∼40% Packet loss	-	-	-	-	-	-	-	-
[101]		0.009∼0.041 joules	-	-	53∼100%	-	-	-	-
[70]	∼99% packet delivery	1.25∼1.35 Jouls 10%∼40% saving compare to attack	-	-	-	-	-	-	13∼55% saving compared to attack
[89]	-	23%∼Consume more than normal	-	-	-	-	-	-	67%∼overhead
[97]	∼100%	11∼94% more power consumption	-	-	-	-	-	-	25%∼ overhead
[90]	-	-	-	-	96%∼ 100%	-	0.2∼ 5.85	-	-
[54]	-	-	89.84 ∼ 97.38%	-	76.19 ∼ 96.02%	-	2.08 ∼ 5.92%	-	-
[55]	41∼95%	∼0.05J consumed in 30 ms	-	-	-	-	-	-	4.7%∼
[22]	-	∼30% Energy overhead	-	-	SH: 79%, SF: 81%	-	High [49]	High [49]	-
[8]	-	-	100%	-	91%	-	-	-	-
[85]	-	-	-	-	-	-	-	-	-
[56]	-	5840mJ	∼100%	∼100%	93.3∼99.4%	-	0.002∼0.058	-	-
[57]	-	-	-	-	-	-	-	-	-
[94]	-	-	-	-	-	-	-	-	-
[99]	-	140∼1250 mJ	-	-	-	-	-	-	-
[80]	-	-	-	-	41∼76%	-	-	-	-
[58]	-	-	-	-	50∼80%and 70∼96%	-	-	-	-
[109]	-	-	-	-	-	-	-	-	-
[103]	-	-	-	-	-	-	-	-	-
[110]	-	-	-	-	70∼93.3% (N/A) 150 attacker nodes	-	28∼62%	3∼20%	-
[100]	99%	∼50% save energy	-	-	-	-	-	-	20∼50% reduced
[52]	70∼90% - up to ∼20% packet loss	6%∼ Energy overhead	-	-	-	-	-	-	-
[59]	-	-	94.50%	-	-	98%	-	-	-
[60]	-	1500000∼1600000mJ	-	-	96.30%	-	6.10%	-	-
[61]	-	save 4%∼ less than normal RPL	60∼100%	-	100%	-	0∼10%	-	22.2%∼ over UDP but ∼29.4% less over DIOs
[77]	-	-	-	-	N/C	N/C	6∼10%	4∼10%	-
[87]	-	-	96∼99.4%	-	94∼100%	-	0.2∼1%	-	-
[81]	-	-	-	-	71∼75%	-	-	-	-
[88]	-	-	-	-	99%	∼1%	-	-	-
[111]	-	Consumed 46% of energy resources	99.44 ∼ 99.97%	-	-	-	-	-	-
[112]	-	2000∼2900mJ	89∼100%	-	-	-	-	-	-
[105]	-	10.45% Energy Overhead	97.23%	-	96.52%	-	2.06%	3.48%	-
[113]	-	-	96.03%	99.51%	95.17%	1.18%	0.98	-	0.13 ∼ 0.42
[62]	-	-	-	-	TP+TN: 87.08	0.97	-	-	-
[71]	30% packet loss	-	-	-	N/C	-	-	-	-
[63]	-	-	-	-	80.95 ∼ 97.88%	-	1.96 ∼ 5.92%	-	-
[64]	-	0.1876	-	-	20 ∼ 100%	-	-	-	-
[114]	-	-	-	-	-	-	-	-	-
[115]	-	-	-	-	-	-	-	-	-
[48]	85 ∼ 97%	0.07∼0.15J	95%	-	93.5∼97.4%	-	-	-	0.5∼3.5
[96]	-	-	-	-	94.4∼95.7%	-	0.46∼0.89%	-	-
[116]	3∼5% packet loss	10∼17J	-	-	80.1∼90%	-	-	-	-
[72]	85∼96%	-	-	-	-	-	-	-	-
[73]	-	-	-	-	100%	-	-	-	-
[91]	-	-	-	-	83∼100%	-	∼11%	-	-
[65]	-	∼20% Energy Overhead provided	62.3∼100%	-	60∼97%	-	∼36%	∼29%	-
[66]	-	Reduces energy consumption to 381J	-	-	-	-	-	-	-
[82]	-	-	-	-	-	-	-	-	-
[117]	-	∼0.04 mW	-	-	90∼100%	-	-	-	-
[67]	-	N/C	-	-	-	-	-	-	-
[68]	-	-	-	-	∼ 100 %	-	0% ∼ 15%	-	-
[78]	Increased (N/C)	∼2 Jouls Lower than normal RPL	-	-	∼ 5% higher than TSD	-	-	-	-
[118]	-	Consume lower power than neighbour & cluster-based	-	-	94%	-	Less than 10%	-	-
[98]	-	N/C	88∼94%	-	-	-	9∼50%	-	-
[95]	-	-	-	-	-	-	-	-	-
[86]	86%	Reduced up to 63%	100%	-	100%	-	0%	0%	Reduced up to 71%
[74]	-	-	-	-	-	-	-	-	Low 0.06%
[75]	60∼80%	0.81∼1.1 mW	-	-	98.50%	-	∼3.7%	3.70%	-
[51]	50∼90%	-	-	-	60∼94%	-	-	-	-
[83]	95∼100%	0.7∼1.4 mW	-	-	97∼100%	-	∼0%	-	-
[104]	-	-	65∼94%	-	-	-	-	-	-

ATTACK DETECTION (AD), INTRUDER NODE DETECTION (IND) PACKET OVERHEAD (PO), NOT ANSWERED (N/A), NOT CLEAR (N/C)

Table 2.8: *State-of-the-art research experiment setup on ids in low power and lossy network (LLN)*

Paper	Evaluation	Node Quantity	Intruder Quantity	Nodes distance	Mobility	Duration	Delay
[92]	Empirical, PenTest (Metasploit) + ebbits	5	3	-	No	30 test, each test 1 minute	-
[25]	Not Validated	-	-	-	No	-	-
[108]	Simulation (Cooja)	100	2	-	-	30 minutes	∼18%
[49]	Simulation (Cooja)	100	1	∼50m, 600m × 600m	No	30 minutes	-
[53]	Simulation (Cooja)	50	10∼15 or 20∼30%	30∼40m	Yes	25 minutes, (Avg 35 runs)	-
[84]	Simulation (Cooja)	20	1	-	No	10 minutes, (3 runs)	-
[79]	Simulation (Cooja)	8∼24	1∼2	-	No	30 minutes	-
[76]	Simulation (Cooja)	8∼40	3	∼50m	Yes	60 minutes	-
[93]	Simulation (NS-2)	25	1∼5	50 m × 50 m terrain	No	-	-
[69]	Simulation (Cooja)	26	3	∼50m	No	60 minutes	-
[101]	Simulation (OMNeT++)	20	1∼3	∼30m	No	1000 seconds	-
[70]	Simulation (Cooja)	10	1	-	No	60 minutes	-
[89]	Simulation (Cooja)	8∼16	1	10∼60m, 200 m × 200m	Yes	5∼15 minutes	-
[97]	Simulation (Cooja)	50	3	10 m, 100m ×100m	No	30 minutes	-
[90]	Simulation (Cooja)	20∼40	1∼20% & 30%	20∼ 40m, 100m ×100m	No	30 minutes	-
[54]	Simulation (C#), the simulator is not available	5∼50	1∼5	∼10m, 100m x100m	No	20 minutes	-
[55]	Simulation (NS-2)	150	1 (N/A)	∼100m, 500m × 500m	No	-	-
[22]	Simulation (Cooja)	8∼64	1∼4	100m ×100m	No	5∼30 minutes(Avg 10 runs)	-
[8]	Empirical	11	1 (N/A)	-	No	-	-
[85]	Not Validated	-	-	-	No	-	-
[56]	Simulation (Cooja)	8	1∼3	-	No	10∼30 minutes	-
[57]	Simulation (Cooja)	12	1	-	No	-	-
[94]	Hypothetical (No Result)	10 (N/A)	1 (N/A)	-	No	-	-
[99]	Simulation (Cooja)	2∼10	1	15∼50m	No	8 hours (6 runs)	-
[80]	Simulation (Cooja)	8∼24	∼2 (12 attacks)	-	No	15 ∼ 60 minutes	-
[58]	Simulation (Matlab)	10∼60	1 (N/A)	1000m ×1000m -random	No	-	-
[109]	Empirical	-	-	-	-	-	-
[103]	Simulation (Cooja)	50	2∼10	∼50m, 300m ×300m	Yes	330 seconds	-
[110]	Simulation (Matlab)	1000	∼300	100m ×100m, random	No	-	-
[100]	Simulation (Cooja)	10	1	-	No	60 minutes (5 times)	-
[52]	Simulation (OMNeT++)	50	1	∼30m, 150m×150m	No	50000 seconds	-
[59]	Simulation (NetSim)	-	-	-	No	-	-
[60]	Simulation (Matlab)	100	1	∼200m (random), 1000m × 1000m	No	-	-
[61]	Simulation (Cooja)	11∼32	1∼4	∼50m	No	30 minutes	-
[77]	Simulation (Cooja)	50	2	∼50m	No	60 minutes	-
[87]	Simulation (Cooja)	50	∼10%	-	No	20 minutes	-
[81]	N/A, C++	10∼200	∼2 (N/A)	-	No	-	-
[88]	Empirical	11	1	-	No	60 minutes	-
[111]	Empirical	100	N/C	-	No	90 minutes	-
[112]	Simulation (TOSSIM)	50∼300	10∼40%	∼15m, 300m × 300m	No	900 seconds	-
[105]	Simulation (C++)	-	1 (N/A)	-	-	-	-
[113]	KDD dataset	-	-	-	No	-	-
[62]	Simulation (Cooja)	11	1	-	No	30 seconds (8 times)	-

Paper	Evaluation	Node Quantity	Intruder Quantity	Nodes distance	Mobility	Duration	Delay
[71]	Simulation (Cooja)	30	3	∼50m, 70 m × 70 m	No	3600 seconds	-
[63]	Simulation (C#), NSL-KDD	10	2	∼10m, 100 m × 100 m	No	30 minutes	-
[64]	Simulation (Cooja)	8∼24	∼2	-	No	5∼50 minutes (5 times)	-
[114]	Hypothetical (Cooja)	7	1	-	No	-	-
[115]	Not Validated	-	-	-	-	-	-
[48]	Simulation (Cooja)	80	1, clone 1∼60% of nodes	∼50m, 300 m × 300 m	Yes	3000 seconds	-
[96]	Simulation (Cooja)	100	1 (N/A)	∼50m, 300 m × 300 m	No	30 minutes	-
[116]	Simulation (Cooja)	30∼70	1 (N/A)	∼50m, 100 m × 100 m	No	300 seconds (10 times)	1.5∼3
[72]	Simulation (Cooja)	20∼50	10∼40%	∼50m, 100 m × 100 m	No	-	-
[73]	Simulation (Matlab)	35	1	∼50m, 100 m × 100 m	No	1000 seconds (10 times)	-
[91]	Simulation (Cooja)	38∼47	1∼6	-	No	25 minutes (5 times)	-
[65]	Simulation (Cooja)	50	2∼8	∼30m, 200 m × 200 m	Yes	600 seconds	2∼25 seconds
[66]	Simulation (Cooja)	100∼200	40%	1000m × 1000m	No	-	detection delay 2215ms
[82]	Simulation (Cooja)	-	-	-	Yes	-	41.1 ∼ 63.3%
[117]	Simulation (Cooja)	4∼8	2	-	No	-	-
[67]	Simulation (Cooja)	12	1	25m	No	-	Minimized delay (N/C)
[68]	Simulation (Cooja)	32	1∼4	100m	No	30 minutes	-
[78]	Simulation (OMNeT++)	100	30	30 m, 200m ×200m	No	30000 seconds	-
[118]	Simulation (N/C)	40	N/C	100m × 100m (random)	No	800 seconds	-
[98]	Simulation (Cooja)	10∼100	10∼30%	50 m	No	20 minutes	-
[95]	Simulation (Cooja)	5	N/C	N/C	No	N/C	-
[86]	Simulation (Cooja)	36	1	50 m, 150m × 150m (Grid & Random)	No	50 minutes	87%
[74]	Simulation (Cooja)	8	1	-	No	30 minutes	-
[75]	Simulation (Cooja)	16∼36	∼3	50 m	No	180 minutes(100 times)	∼1190 ms
[51]	Simulation (Cooja)	∼18	∼4	150m × 150m	Yes	1800 seconds	0.25 ∼ 1.27 second
[83]	Simulation (Cooja)	21∼101	2∼10%	25m, 800m × 800m	No	5 ∼ 30 minutes	-
[104]	Simulation (Cooja)	16	4	150m × 150m	Yes	1800 seconds	-

reduce the financial cost of the network, as there is no need for extra equipment and sniffers to probe the LLN. Furthermore, LLN nodes can provide host machine configuration and other information, which is not available in passive monitoring techniques. However, assigning the monitoring tasks to the LLN constrained nodes increases network traffic overhead as nodes need to transfer their information to centralised or decentralised IDSs. The passive monitoring technique can reduce the active monitoring shortcomings while providing IDS with less detail about LLN nodes. Moreover, even though passive monitoring can provide a comprehensive view of the monitored network [129], the use of separate network communication (i.e. a collection of probes) may increase overhead costs and restricts their benefit to the small-scale and controlled network. As a result, and as seen from Fig 2.14, most proposals use active monitoring. Researchers in [99] use a different channel for IDS agents' communication to reduce the negative impacts of IDS communication on LLN in passive monitoring techniques. Table 2.6 shows what monitoring technique is used in each research, and Tables 2.4 and 2.5

give each monitoring technique's strengths and weaknesses.

2.7.5 Proposed IDS strategies (Q 5)

Sections 2.4 and 2.5 discussed existing IDS techniques and monitoring methods in the literature. Fig. 2.14.B, and Fig. 2.14.C show the proportions of each IDS strategy built on by the reviewed papers and the proportion of their response types, respectively. We can see that the majority, 54%, of the introduced methods use a specification-based detection strategy to detect RPL attacks, 21% are hybrid, 17% are anomaly-based, and the remaining 8% are signature-based IDS. This noticeable difference in proportions is because specification-based IDS uses less storage space and consumes less computational resource than misuse-based and anomaly-based detection approaches. However, such IDSs are inflexible and do not adapt automatically to attacks, as stated in section 2.4.2.

2.7.6 Shortcomings of proposed methods (Q 6)

Studying Fig. 2.11, Fig. 2.12, and Fig. 2.13 shows the vulnerabilities and shortcomings of each IDS strategy in LLN. Further study is required to address the shortcomings stated in Table 2.5. From Fig. 2.12 and Fig. 2.13, We can see that researchers obtain the least FPR and the best TPR using misuse detection techniques with 1% and 95.2% FPR and TPR, respectively. However, none of the proposed misused-based IDS, which constitute 8% of all proposed methods, provides any evaluation of FNR, which is an essential metric, especially for evaluating signature-based IDS. Anomaly IDS provides the researchers with the least FNR of 5.8% and better detection rate than the specification-based method, on average, 92.3% TPR. However, as we expect in section 2.4.2, the FPR of anomaly-based and specification-based methods was higher than signature-based IDS, with 5% and 12.2% FPR, respectively. Although 21% of researchers attempt to minimise the FPR and receive the optimum TPR by developing a hybrid IDS, this detection strategy provides researchers with 88.4% TPR and 6.8% FPR on average. The illustrated results in Fig. 2.12 and Fig. 2.13, and the proportion of detection strategies in Fig. 2.14, reveal the need for further investigation into hybrid detection strategies to boost the performance of IDS in the 6LoWPAN.

2.7.7 Datasets and simulators used by researchers (Q 7)

There are several well-known intrusion datasets developed through simulations, Capture The Flag (CTF) competitions, or empirical lab experiments. These have been used by researchers to train and test their proposed methods to detect various types of attacks. Some very well known datasets are KDD 99, NSL-KDD, Defcon, and CDX. However, none of the mentioned datasets include either 6LoWPAN traffic or any RPL-based attacks. This is because they were not generated through IoT simulation or empirical experiments and mostly include application layer intrusions. The lack of an recognised, reliable dataset compelled researchers in this field to evaluate their proposed methods through simulation or empirical experiments. Table 2.10 introduces several simulators that exist in this field and are widely used by researchers for simulation and evaluation purposes. Some of these simulators are employed more than others. 73% of the proposed researches in this domain have used the Cooja simulator for simulation and evaluation purposes. The authors of [130] compared different simulators for

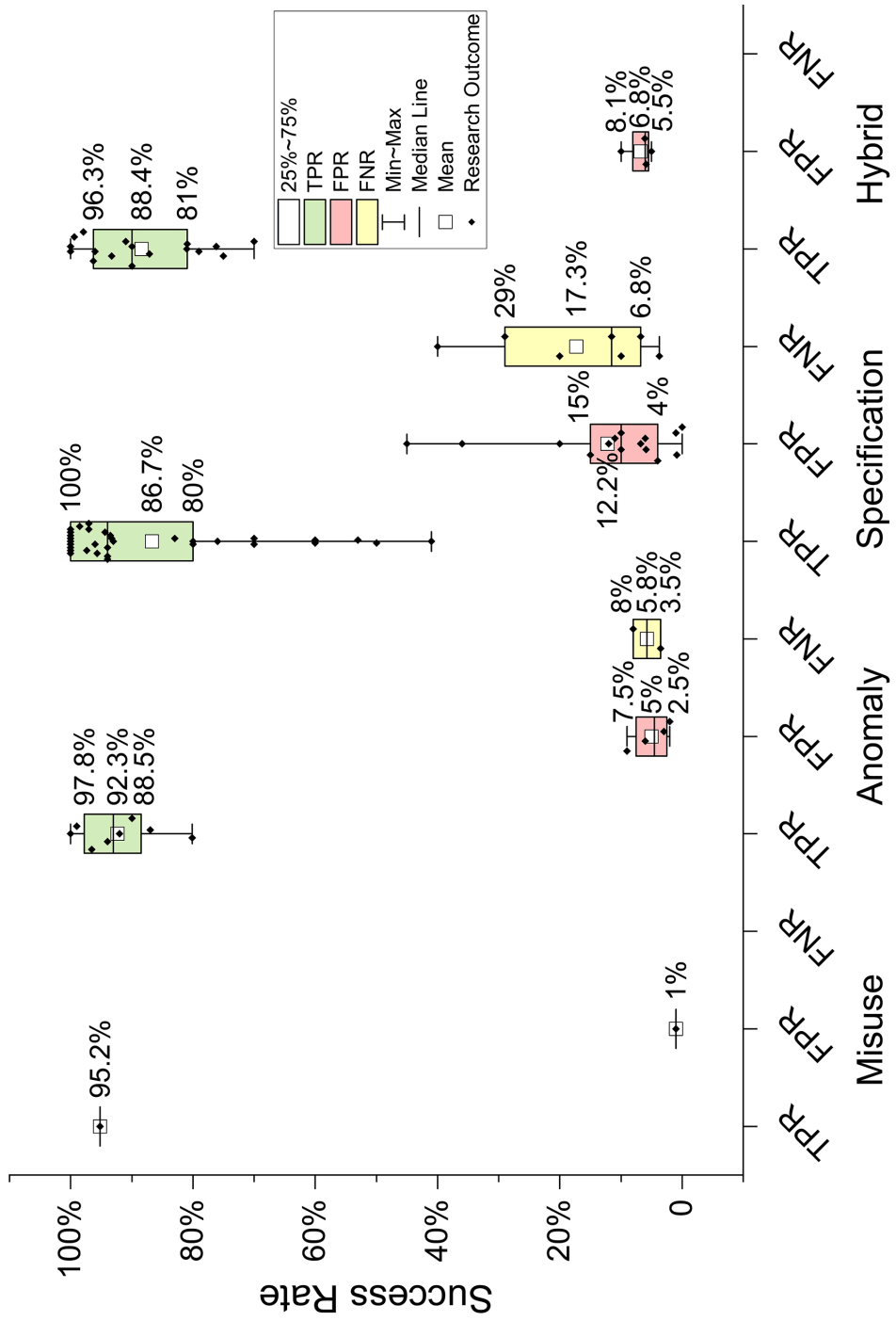


Figure 2.12: *Statistical results regarding detection strategies.*

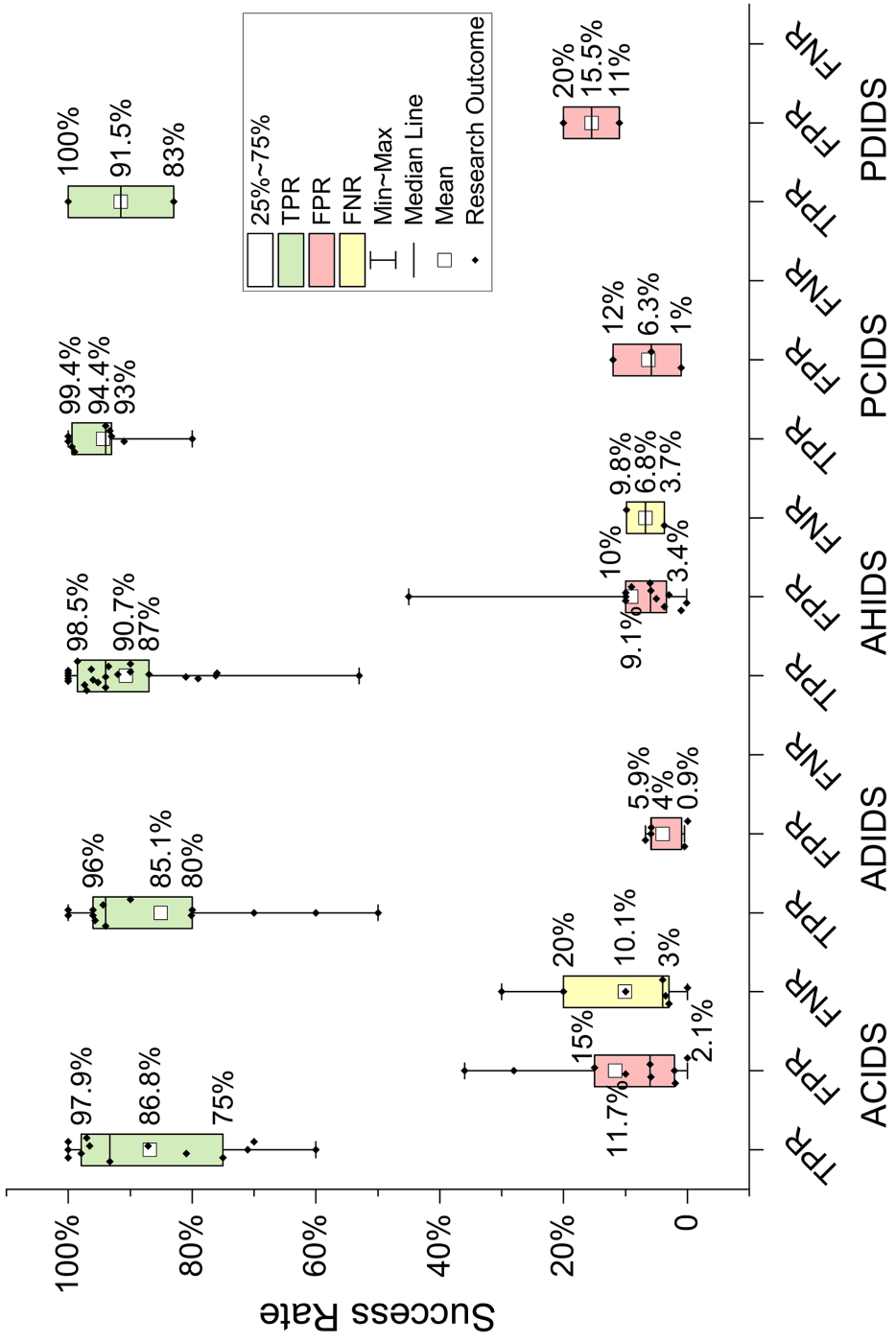


Figure 2.13: *Statistical results regarding monitoring strategies.*

Table 2.9: *Researchers objectives*

Paper	Obj 1	Obj 2	Obj 3	Obj 4	Obj 5	Obj 6	Obj 7	Obj 8	Obj 9	Obj 10
[92]	✓∗	-	-	-	-	-	-	-	-	-
[25]	-	-	-	-	-	✓∗	-	S	-	-
[108]	✓	✓	-	✓	-	-	✓	S	✓	-
[49]	✓	✓	-	✓	-	✓	✓∗	S	-	-
[84]	-	-	-	✓∗	-	-	-	S	✓	-
[53]	✓	-	-	✓	✓	-	✓	A	✓	✓
[79]	✓	✓	✓	-	-	-	✓	-	-	-
[76]	✓	-	✓	-	✓	-	✓	A	✓∗	-
[93]	✓	-	-	✓∗	-	-	✓	-	-	-
[69]	-	-	-	-	-	-	✓	-	✓∗	✓
[101]	✓	✓	-	-	-	-	-	-	✓∗	-
[102]	✓	-	-	-	-	-	-	-	✓∗	✓
[54]	-	-	-	-	-	-	✓∗	-	✓	-
[70]	-	✓	✓	-	-	✓∗	-	S	✓∗	✓
[89]	-	✓	✓	-	✓	✓∗	-	S	✓∗	-
[97]	-	✓	✓	-	-	-	✓	S	✓∗	✓
[90]	✓	-	-	✓∗	-	✓∗	✓	A	-	-
[54]	✓	-	-	✓∗	-	✓∗	✓	S, A	-	-
[55]	-	✓	✓	-	-	-	✓∗	S	✓∗	✓
[22]	✓∗	✓	-	✓∗	-	✓∗	✓	-	✓	-
[8]	✓	-	-	-	-	-	-	-	-	-
[85]	-	-	-	-	-	✓	-	-	✓∗	-
[56]	✓	✓	-	✓∗	-	✓	-	-	-	-
[57]	-	-	-	-	-	✓	-	-	-	-
[94]	-	-	-	-	-	-	-	-	-	-
[99]	-	✓	-	-	-	-	-	-	-	-
[80]	✓∗	-	-	-	-	-	✓∗	S	-	-
[58]	✓∗	-	-	-	-	✓∗	✓∗	S	✓	-
[109]	-	-	-	-	-	-	-	-	-	-
[103]	-	-	-	-	✓	-	✓	S	✓	-
[110]	✓	-	-	✓	-	-	✓	S	-	-
[100]	-	✓	✓	-	-	-	-	S	✓	✓
[52]	-	✓	-	-	-	-	✓∗	S	✓	✓
[59]	✓	-	-	✓	-	✓	-	-	-	-
[60]	✓	✓	-	✓∗	-	✓∗	✓∗	-	✓	-
[61]	✓	✓	✓	✓∗	-	-	-	S	✓	-
[77]	-	-	-	✓	-	-	✓	S	-	-
[87]	✓	-	-	✓∗	-	✓	-	-	-	-
[81]	✓∗	-	-	-	-	-	✓	-	✓	-
[88]	✓	-	-	-	-	✓∗	-	A	-	-
[111]	✓	✓	-	-	-	-	✓∗	A	✓	-
[112]	✓∗		-	-	-	-	✓	-	-	-
[105]	✓	-	-	-	-	-	-	-	✓	-
[113]	✓	-	✓	✓∗	-	-	-	-	-	-
[62]	✓	-	-	✓	-	✓	-	-	-	-
[71]	✓	-	-	-	-	-	-	S	✓	✓
[63]	✓	-	-	✓∗	-	✓∗	-	-	-	-
[115]	-	-	-	-	-	-	-	A	-	-
[48]	✓	✓	✓	-	✓	-	✓∗	S	✓∗	✓
[96]	✓	-	-	✓∗	-	-	-	S	-	-
[116]	✓	✓	✓	-	-	-	-	A	✓∗	✓
[72]	-	-	-	-	-	-	-	S	-	✓
[73]	✓	-	-	-	-	-	-	A	-	-
[91]	✓	-	-	✓∗	-	✓∗	-	S	✓	-
[65]	✓	✓	-	✓∗	✓	-	-	S	-	-
[66]	-	✓	-	-	-	-	✓	S	✓∗	-
[82]	-	-	-	-	✓	-	✓	A	-	-
[117]	✓	✓	-	-	-	-	-	H	✓∗	-
[67]	-	✓	-	-	-	✓	-	S	✓∗	-
[68]	✓	-	-	✓	-	-	-	S	-	-
[78]	✓	✓	-	-	-	-	-	S	✓∗	✓
[118]	✓	✓	-	-	-	-	-	S	✓∗	-
[98]	✓	-	-	✓∗	-	-	✓	A	✓	-
[95]	-	-	-	-	-	✓	-	A	-	-
[86]	✓	✓	✓	✓	-	-	✓∗	S	✓	✓∗
[74]	-	-	✓	-	-	✓∗	-	S	-	-
[75]	✓	✓	-	✓	-	-	✓	S	✓	✓
[51]	✓	-	-	-	✓	-	✓	S	✓	✓
[83]	✓	✓	-	✓∗	-	✓∗	✓	S	-	✓
[104]	✓	-	-	-	✓	-	✓	S	✓	-

Obj1: ✓∗ below Quartile 1(25th percentile) **Obj4**: ✓∗ denotes either FPR or FNR not provided. **Obj6**: ✓∗ indicates the research detect more than one but less than 3 RPL attacks. **Obj7**: ✓∗ determines only one attacker node placed among a large number of normal nodes. **Obj8**: There is not any research to provide a result for this objective. We can consider the possibility of addressing this objective by mentioned detection strategy, Anomaly-based (A), or Specification-based (S). **Obj9**: ✓∗ states that the proposed method contains a prevention mechanism; however, the study did not provide the amount of FPR, which is essential for such mitigation methods. **Obj 10**: The provided PDR is below quartile 1(25th percentile)

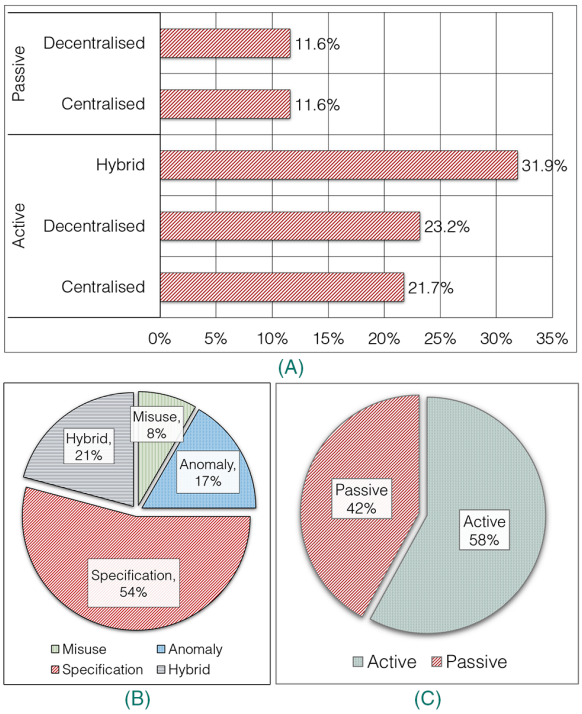


Figure 2.14: *The proportion of each monitoring and detection strategy.*

RPL networks. However, some well-known simulators such as Netsim, Opnet, NS-3, and Matlab did not appear in their study; we include them here while adding more detail to the information on existing simulators. The average numbers of normal and malicious nodes in the testbeds are 49 and 4, respectively. Researchers model 8.2% of nodes as malicious on average. The minimum, average, and maximum experiment runtimes were 30, 2196, and 50000 seconds, respectively.

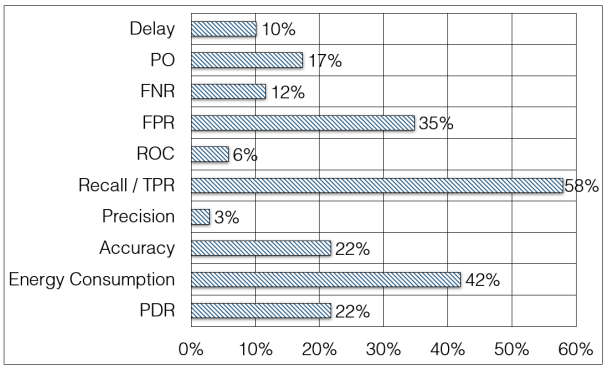


Figure 2.15: *The evaluation metrics and their usage.*

2.7.8 Used evaluation methods (Q 8)

We investigated to what extent each evaluation method was used to measure the performance of the proposed detection technique in this domain. Using a simulator for evaluation has the lion's share with 86%, while empirical evaluation makes up only 7%. Providing more evaluation results can help the readers to understand the strength and weaknesses of the investigated IDS. However, researchers mostly did not provide sufficient evidence to justify the achievement of their aimed contribution by comparing the performance of the proposed method over normal and attack scenarios. Researchers provide the least evidence for FNR while it is crucial for evaluating the capability of IDS. Fig. 2.15 reveals to what extent each evaluation metric was considered by researchers.

2.8 Existing Research Gaps

Reviewing the proposed IDS approaches for RPL enables us to identify gaps and research opportunities. We believe further study on the aforementioned and less-investigated research questions can enhance RPL security and make it more resilient. Below we summarise the remaining gaps and provide some suggestions to address them. Further study opportunities are provided to answer Q9.

2.8.1 Comprehensive in detecting attacks

Existing studies in IDS are often limited to the detection of one or a very small number of RPL attacks. Our review did not find any proposed IDS securing LLN against all known types of RPL attacks. There seems to be significant opportunity for detectors that can operate effectively over the range of possible attacks (both known and unknown).

2.8.2 Exploiting machine learning for defence

Table 2.11 shows the ML-based IDSs proposed to secure networks against RPL-attacks. Some of the proposed approaches [56] are scenario-based and may not perform well in LLNs different to the training target. There is a significant opportunity to further explore the rich defensive possibilities offered by ML. Crafting resource efficient intrusion detectors seems an obvious and important target for the application of ML, e.g. use ML to synthesise RPL attack detectors that consume little power.

2.8.3 ML-based intrusions

ML has proven to be very powerful and effective in detecting intrusions but what if intruders decide to use ML to establish ML-based RPL intrusions? After discovering vulnerabilities in the targeted LLN, intruders can adopt an ML-based malicious system (e.g., using a reinforcement learning algorithm) to enter a game with the RPL network and discover the most effective intrusion for damaging nodes' CIA (confidentiality, integrity, and availability). Since intruders may use advanced ML algorithms to achieve their goals, detecting such intrusions might be challenging. Thus, ML may be used to synthesise effective and highly stealthy strategies for attack. This study has not found any IDS research capable of securing LLN against such sophisticated intrusions.

Table 2.10: *The most popular network simulators*

Simulator	Strength	Shortcoming	Scripting
Cooja	- A comprehensive level of detail in simulating RPL networks. - Adjustable OF (ETX is the default OF). - Cooja is called an emulator rather than a simulator; the compiled firmware in the Cooja can upload and emulate over real devices (e.g., Tmote Sky, Z1). - Provide power consumption, CPU, RAM, ROM usage. - Open-source. - MRHOF and OF0 have been implemented; adding a new OF is possible - There is no limitation in the number of border routers in this simulator. - Several sensors/mote types with heterogeneous computational resources are available in the simulator.	- Does not include the repair mechanism of DODAG. - Limitation in the maximum number of nodes. - Low accuracy, especially in large-scale scenarios. - It cannot simulate communication between the border router and other network technologies. – lack of official documentation.	C
Netsim	- A comprehensive level of detail in simulating RPL networks. - Provide power consumption. – Emulator feature connects the simulation environment to real hardware and devices to generate more accurate results. - Can simulate a large-scale scenario with a significant number of nodes. - Can connect border router to other network technologies. Netsim supports other network technologies, namely 5G NR, LTE, Vanets, Manet, Satellite communication. – Support MRHOF and OF0. -Adjustable API code. – Comprehensive documentation. – Can interact with Node-red environment.	- DODAG repair mechanism is not supported. - Paid license. - Unlike Cooja, Netsim cannot provide CPU, RAM, ROM usages of nodes in the simulation environment. - Cannot place more than one border router in a scenario - The existing sensors are not heterogeneous in terms of computational resources; however, there is a possibility to develop heterogeneous sensors in terms of transmission rate by modifying the API codes.	C
Matlab	Easy to implement.	Generic support of RPL. Paid license.	Matlab
NS-3	- Open-source. - Although it can simulate scenarios with a massive number of nodes, increasing the number of nodes cause a negative impact on the accuracy and precision of its simulation results	- The GUI is only supported for simulating traffic flow. - Generic support of 6LoWPAN and AODV. - Inadequate support in MAC and PHY layer development.	C++& Python
NS-2	Open-source	- Only support GUI in simulating traffic flow, Not supporting RPL, 6LoWPAN, but WSN. - The number of nodes cannot be more than ∼100 nodes. - Highly errored and unstable simulation environment. - Lack of IDE debugger.	C++ & OTcl
TOSSIM	- Support an extensive scale network with thousands of nodes. - Support energy consumption with power TOSSIM add-on. - Open-source.	- Cannot simulate power consumption. – Only support few hardware types. - Although TinyRPL supports RPL, the TinyOS simulator does not include TinyRPL and cannot support the RPL simulation.	C++ & Python
OMNeT++	- Open-source. - Support power monitoring.	- Generic level of detail. - Simple Tree, Multipath Rings network exist in OMNET++; however, RPL can simulate in OMNET++	C++ & NED
OPNET	- Support power monitoring. – Support different network technologies (e.g., Zigbee, WSN, Satellite). - It can simulate a pervasive network scenario in an immense geographical scale, cities, countries. - Known for very accurate simulation results. - Educational license available	- It does not support RPL, 6LoWPAN.	Proto - C
WSim/WSNet	- Acceptable level of detail in simulating RPL networks. - Open-source. - Provide power consumption, CPU, RAM, ROM usage.	It does not have any GUI interface for configuration and simulation results.	C
J-Sim	- Support up to ∼500 nodes; however, this impact simulation times significantly. - Open-source.	- Hard to interact with. - Only support IEEE 802.11 and not compatible with RPL.	Tcl

Table 2.11: *ML-based IDS for RPL*

Ref	Algorithm	Attack	Evaluation
[88]	Neural Network, Genetic Algorithm	Man in the Middle (MitM)	Empirical (11 nodes)
[111]	Decision Tree, NB, LDA	SYN Flood, HTTP Flood, Exploit attack	Empirical (100 nodes)
[63]	Supervised Optimum-Path Forest (OPF) and Modified OPF (MOPF)	SF, SH	Simulation (12 nodes) and NSL-KDD
[62]	Voting (MLP, Random Forest)	Version, Rank, Sybil, Decrease Rank, Blackhole	Simulation (12 nodes)
[113]	Weighted voting, Bat, Random Forest	No RPL attack	KDD dataset
[81]	unsupervised K-means and supervised Decision Tree	WH	Simulation (∼200 nodes)
[54]	unsupervised Optimum-Path Forest Clustering (OPFC)	SH, SF, WH	Simulation (∼50 nodes)

2.8.4 Evaluating the detection of unknown attacks

The anomaly-based, specification-based, and hybrid IDSs are known for their capability in detecting unknown intrusions. Although 17%, 54%, and 21% of existing IDSs employ such detection strategies respectively, we did not find any performance evaluation of these IDS over unknown attacks. Here, an unknown intrusion is an attack that IDS is not trained for.

2.8.5 Adaptive IDS

The existing ML-based IDSs for 6LoWPAN are offline/batch trained. As discussed in Section 2.4.4, such ML-based IDSs require training over the training dataset to update/rebuild their detection engines, where the dataset includes new observations and old ones, where re-training a pre-constructed classifier using the entire training dataset is strenuous and expensive concerning computational exhaustion and time. Hence, an adaptive ML-based IDS is required for 6LoWPAN to facilitate this need. However, developing an adaptive/incremental IDS that capable of accurately classifying the 6LoWPAN evolving data stream is a challenging task. The IDS needs to update its detection classifier incrementally with the change (shift) in the network environment to be robust and maintain its intrusion detection accuracy.

2.8.6 Study dynamic scenarios

The proposed IDSs mostly consider a network scenario with a fixed number of nodes in a static environment. However, an LLN is a lossy and unstable dynamic network. Nodes may continuously move in and out of the LLN. Hence, the number of nodes increases and decreases over time. Therefore, it is essential to consider such a dynamic, unstable and scalable network while developing IDS for RPL because such elements have a direct effect in the detection of attacks such as DIS flooding, SH, SF etc.

2.8.7 RPL attack dataset

There is a pressing need for a comprehensive RPL network dataset that is freely available for researchers. This would be a major research enabler, allowing meaningful evaluation of any proposed RPL IDS techniques.

2.8.8 Real time notification

Accurate and timely detection of malicious activities critically depends on the monitoring technique adopted. The ability to detect breaches early is the most valuable aspect of any IDS. As stated in section 2.5, there are several proposed methods for deciding on where to place the monitoring nodes and the IDS agents for monitoring and detection purposes. However, the large number of geographically spaced connected devices makes it hard to inspect packets in real-time. This negatively impacts the alarm and response time. There is a need for more research to provide these means for IDS to detect RPL attacks accurately while providing real-time notifications.

2.8.9 Adopt a lightweight approach

The LLN nodes are constrained by nature and barely function properly for their assigned tasks; they are constrained in processing, memory and power and may not be able to hold tasks other than the ones assigned to them. Furthermore, 6LoWPAN suffers from a wide range of different disruptive attacks, as mentioned in section 2.3. Designing a complex detection algorithm that can detect major RPL attacks is more likely to exhaust LLN node computational and energy resources. Therefore, future IoT-RPL intrusion detection solutions must be robust yet lightweight.

2.9 Summary

The features and capabilities of IoT devices allow them to be utilized and incorporated everywhere: in health care sectors, smart cities, smart homes, and industrial environments. They are exposed to various types of routing attacks. The RPL protocol underpins the network operation of many modern LLNs. This review has explored attacks against this protocol and identified the state-of-the-art in the use of IDSs to detect attacks on networks that run this protocol. We have identified existing research gaps and possible future research directions. In the literature, several RPL attacks (e.g. Increase Rank, Worst Parent, and DIO Suppression) are overlooked or not addressed at all. Additionally, the lack of adaptivity in existing RPL-based IDS deprive detection models to adapt/adjusting to any shifts in the 6LoWPAN data environment.

Chapter 3

Adaptive Hybrid Heterogeneous IDS for 6LoWPAN

Developing a security infrastructure for 6LoWPAN is made difficult by resource constraints of nodes. Further challenges are posed by the streaming and evolving environment of LLNs. IDSs have been proposed as a means to detect RPL attacks but in practice their focus has been limited to specific attacks and they typically assume a stationary environment. Furthermore, IDSs for detecting RPL attacks must cope with the often significant resource constraints indicated above. This chapter introduces an adaptive hybrid IDS scheme to accurately detect and identify a wide range of RPL attacks in evolving data environments. We apply our proposed scheme to the networks under various levels of node mobility and maliciousness. To develop an adaptive hybrid Centralised IDS (CIDS), we experiment with several incremental machine learning (ML) approaches and various 'concept-drift detection' mechanisms (e.g. ADWIN, DDM, and EDDM).

3.1 Introduction

The 6LoWPAN has a streaming data environment. An IDS does not have access to the entire data stream at any point in time and cannot afford to store all incoming instances (data is unbounded). Existing IDSs proposed for 6LoWPAN work only in stationary environments where the number of nodes in each scenario does not change. However, 6LoWPAN has an evolving data environment, where node movement, inaccessibility, change in a node application, and unforeseen attacks, alter the data stream distribution. In 6LoWPAN, nodes cannot accommodate a large volume of data in their memories. Moreover, in non-stationary evolving environments, the data distribution evolves unpredictably and so the system needs to update its model incrementally or retrain it using recently observed batches of data. To address the aforesaid issues, "*concept drift*" detection approaches have been introduced in different network paradigms to enable adaptivity in IDS [121], where the "*concept*" can be defined as a joint distribution $P(X|Y)$, where X denotes a vector of attributes values (features) and Y is the target value (label) [131]. Concept drift is a shift in the data distribution $P(X)$, where $P_t(X, Y) \neq P_{t+1}(X, Y)$. The rate of concept drift is unknown to the system and can be abrupt, incremental, gradual or recurring [121]. Concept-drift Detection (CD) methods can enable an IDS to adapt to unforeseen intrusions and identify other shifts in the network

data stream [120]. Additionally, the CD approaches make efficient use of storage and memory resources and facilitate fast classification.

Developing an adaptive IDS capable of accurately classifying the 6LoWPAN evolving data stream is a challenging task. The classifier needs to update itself with each change (shift) in the environment to continue to detect novel attacks. Retraining a classifier using the entire training data is computationally expensive. This chapter proposes an adaptive form of hybrid ensemble IDS capable of enhancing system performance using streaming data mining techniques and drift detection. The proposed scheme is capable of identifying various internal (sourced inside 6LoWPAN, e.g., sinkhole, blackhole, and grayhole) and external (sourced over the Internet, e.g., wormhole and DIS flooding) routing threats targeting 6LoWPAN. Different ensembling techniques have been adopted and compared in this chapter. We employ a passive decentralised monitoring technique (where anomaly-based IDS agents passively monitor network communications and send abnormal/suspicious observations to the central IDS for further analysis) to collect and monitor LLN traffic from different locations and avoid additional computational overheads over legitimate nodes for intrusion detection purposes.

3.1.1 Related work

A broad range of routing vulnerabilities in 6LoWPAN and the lack of effective built-in security mechanisms in RPL [3] have encouraged researchers to develop IDSs for detecting RPL attacks and have adopted various monitoring and detection strategies [3]. These schemes [132; 79; 99; 65] use a specification-based IDS to detect Sinkhole (SH), Wormhole (WH) and DIS flooding (DA) attacks. 54% of existing IDSs employed a specification-based detection strategy for detecting routing attacks in 6LoWPAN [3]. Specification-based IDSs employ a set of static rules for identifying intrusions; they cannot update their rules automatically. Only 21% of the reported works have considered a hybrid detection strategy [3] but none considers mobility of nodes. The shortcomings of the statistical and rule-based detection approaches [3] have encouraged researchers to apply machine learning (ML) algorithms to enhance the performance of IDS in 6LoWPAN. Among existing hybrid IDSs for 6LoWPAN, only a few [81; 62; 54] are ML-based. Moreover, they [62; 81; 56; 54] used offline ML approaches, where the intrusion detection model is constructed using a stationary batch of training data. Due to the evolving data stream of 6LoWPAN, the detection performance of the batch-trained ML-IDS degrades over the time [120]. Nevertheless, because of memory constraints in 6LoWPAN, legitimate nodes cannot store extensive records of malicious activities. Hence, less critical records should be replaced with vital ones over time. However, to the best of our knowledge, no existing IDS for 6LoWPAN does this.

Various proposed monitoring techniques observe inter-node communication in the 6LoWPAN [3] (e.g. centralised and decentralised active or passive monitoring approaches). They [132; 79; 65; 117; 22; 90; 62; 81; 54] employed an active monitoring technique to detect RPL attacks in 6LoWPAN. According to [3], ∼77% of existing IDSs used an active monitoring technique, where legitimate nodes were required to participate in intrusion detection tasks with centralised or decentralised intrusion detectors. Active monitoring can provide more information about node configuration (e.g. geographical location, energy consumption, and CPU, RAM, ROM usage) and result in more accurate detection of RPL attacks. However, it also causes additional computational overhead on the legitimate nodes. Consequently, some 6LoWPAN IDS papers employed passive centralised [56; 111] and passive decentralised

Table 3.1: *Related works*

Scheme	Method	Attacks Considered	Desirable Properties						
			DP1	*DP2*	*DP3*	*DP4*	*DP5*	*DP6*	*DP7*
[22]	Hybrid active decentralised IDS	SH and GH (using Cooja simulator)	×	×	×	×	×	×	×
[132]	Specification-based IDS, Highest Rank Common Ancestor	WH and Sybil (using Cooja simulator)	×	×	✓	×	×	×	×
[65]	Specification-based active centralised IDS	SH (using Cooja simulator)	×	×	×	×	×	×	✓
[81]	Unsupervised Anomaly-based K-Means and Supervised Signature-based Decision Tree	WH	×	×	×	×	×	×	×
[62]	Ensemble Voting (MLP and RF)	SA, VN, SH, and BH	×	×	×	✓	×	×	×
[54]	Unsupervised Optimum-Path Forest Clustering	SH, WH, and SF	×	×	×	×	×	×	×
[56]	Hybrid ML-IDS using passive monitoring technique	SH, WH, and DA (using Cooja simulator)	×	×	×	×	×	×	×
[117]	Active decentralised hybrid IDS (requires geographical information of nodes)	SH (using Cooja simulator)	×	×	×	×	×	×	×
[90]	Active decentralised anomaly-based IDS	DA and NA	×	×	×	×	✓	×	×
[94]	Passive decentralised signature-based IDS	DA (using Cooja simulator)	×	×	✓	×	×	×	×
[49]	Active decentralised specification-based	WP, DA, SH, and DF	×	×	×	✓	×	×	×
[133]	Online adaptive RF + concept drift	KDDCup99 (application layer attacks)	✓	✓	✓	D/N	×	×	D/N
[134]	Online RF (Hoeffding Trees)	KDDCup99 (application layer attacks)	✓	✓	✓	D/N	×	×	D/N
[113]	Ensemble Weighted Voting, RF	KDDCup99 (application layer attacks)	✓	×	✓	D/N	×	×	D/N
[135]	Concept drift (HDDM) based ensemble incremental learning approach in IDS	KDDCup99 (application layer attacks)	✓	✓	✓	D/N	×	×	D/N
[136]	Online Sequential-Extreme Learning Machine (OS-ELM)	NSL-KDD 2009 (application layer attacks)	✓	D/N	D/N	D/N	×	×	D/N
Our Scheme	One-Class SVM, incremental OzaBaggingADWIN using KNN, and HalfSpace-Trees	SH, BH, GH, DA, DS, IR, WH, and WP (Netsim v13)	✓	✓	✓	✓	✓	×	✓

*D/N: Different Network-technology. * In the "Attack" column, the later entries refer to available datasets that contain a variety of attacks, (but these exclude RPL attacks); ✓: Satisfy; ×: Not addressed; **SH:** Sinkhole, **BH:** Blackhole; **WH:** Wormhole; **DS:** DIO Suppression; **WP:** Worst Parent; **GH:** Grayhole; **DA:** DIS Flooding; **IR:** Increase Rank;

[94; 99; 84] approaches. Passive monitoring does not cause any additional computation overhead for legitimate nodes [84]. Nevertheless, it can provide IDS only with control packets that are multicasted or unicasted by monitoring nodes' neighbours.

According to [3], existing IDS mainly focus on detecting sinkhole (21%), grayhole (14%), blackhole (10%) and DIS flooding (10%) attacks while other RPL attacks are overlooked.

There is no research in the literature to develop and evaluate the performance of IDS against external routing attacks (external DA and WH). Furthermore, only 13% of RPL IDS research has considered mobility [3]. Table 3.1 describes the related works in the literature and the contributions that this chapter makes.

3.1.2 Motivation and contribution

The Routing Protocol for LLNs (RPL) is vulnerable to various routing threats (e.g. sinkhole, blackhole, and wormhole). Further more, the 6LoWPAN data environment evolves on an unpredictable basis. Different IDSs have been proposed in the literature to detect existing RPL attacks in 6LoWPAN (as discussed in Section 3.1.1). However, none of the existing IDS satisfies all the desirable properties (as mentioned in Section 1.2.1).

Although batch-trained ML algorithms can enhance the performance of an IDS in detecting RPL attacks in a stationary data environment, they are offline and require a large training dataset to develop their detection engine. In 6LoWPAN, an IDS observes a considerable (unbounded) volume of data as a continuous flow; hence, it cannot explicitly store all observations to identify anomalous activities. To maintain detection performance, it is expected that the IDS modify its detection model on a regular basis and incrementally adapt to unforeseen data distributions. We propose and evaluate an adaptive heterogeneous ensemble hybrid IDS framework to detect various types of RPL attacks in 6LoWPAN. The hybrid detection strategy helps the proposed framework to balance the computational cost of the anomaly-based intrusion detection and the storage cost of the signature-based intrusion detection on legitimate nodes. Besides, various incremental ML algorithms and ensemble techniques are evaluated to determine the most suitable combinations for the proposed system. The major contributions of this chapter are to provide:

- A new adaptive hybrid IDS to detect internal and external RPL attacks.
- An efficient concept-drift-based ML-IDS, maintaining effectiveness in the face of environmental change.
- A powerful IDS identifying a wide range of RPL attacks, including less researched ones (e.g. Increase Rank, DIO suppression, and Worst Parent attacks)
- A comparative study of the proposed scheme with closely related existing schemes.
- A comprehensive dataset for ML-based IDSs containing different RPL attacks, namely Sinkhole, Blackhole, Selective Forwarding, Increase Rank, DIS Flooding, Worst Parent, DIO Suppression, Replay, and Wormhole. The generated dataset is publicly available to facilitate further research.

3.1.3 Organisation

The rest of the chapter is organised as follows. In Section 3.2, we present our proposed scheme. In Section 3.3, we describe our implementation and evaluation details. Section 3.4 concludes the chapter.

3.2 Proposed Scheme

Our proposed scheme employs a passive decentralised monitoring approach [84] using a cluster-based placement [124] strategy to analyse the data stream in 6LoWPAN. Anomaly-based detectors are spread over the 6LoWPAN to analyse their neighbours' control packets and report abnormalities to the Centralised IDS (CIDS) on the 6LoWPAN Border Router

(6BR). The CIDS is an adaptive heterogeneous hybrid IDS that protects 6LoWPAN against internal and external intrusions. Fig. 3.1 illustrates the system architecture. The proposed scheme has three components: an anomaly-based network IDS (ANIDS), incremental ensembles of signature-based IDS, and incremental ensembles of anomaly-based IDS. These are described in subsequent sections. Algorithm 9 shows the proposed scheme.

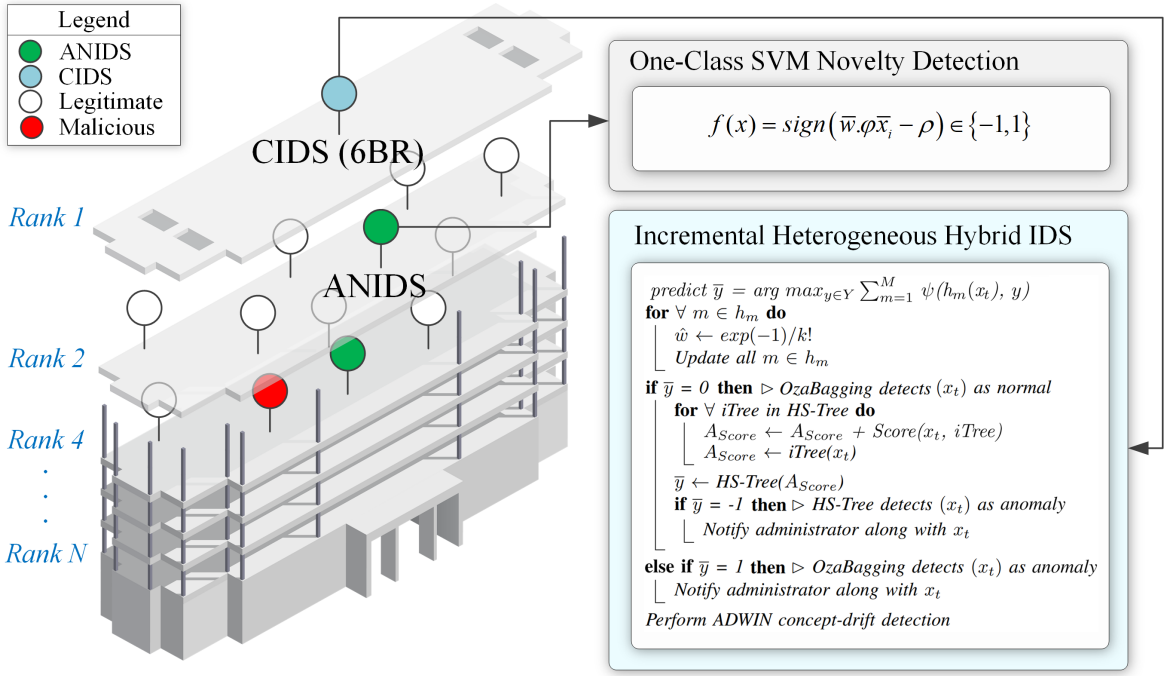


Figure 3.1: *System model.*

3.2.1 Anomaly-based network IDS (ANIDS)

Since the CIDS on the 6BR cannot observe network communications of distant nodes (since the 6BR has limited radio range and RPL may operate in storing mode [3]), the proposed scheme distributes Anomaly-based Network IDS (ANIDS) agents to passively monitor multicasted and unicasted control packets of their neighbouring nodes without requiring significant storage space. As shown in Experiment 1 (Section 3.3.2), a One-Class SVM (OCSVM) can provide excellent performance in detecting intrusions with negligible false-alarms and excellent recall value. The OCSVM is a novelty detection algorithm that develops a profile (model) of safe activities and classifies instances as an outlier if they deviate from that profile. The outcome of an OCSVM is bipolar, $y_t = -1$ for $x_t \in$ outliers and $y_t = +1$ for $x_t \in$ inliers. In OCSVM, the classifier assumes that the given training dataset X contains only normal (safe) instances, $X = \{\overline{x}_1, \overline{x}_2, ..., \overline{x}_N\}$ $\overline{x}_i \in Normal$, and considers the origin of a kernel-based transformed representation as an outlier. OCSVM aims to discover a separating boundary (hyperplane) $\overline{w}.\phi(\overline{x}_i)$ that maximises the distance between normal instances $(\overline{x})$ and the origin $(0,0)$, $\overline{w}.\phi(\overline{x}_i) - \rho = 0$ (define the hyperplane) where $\overline{w}$ and $\phi(.)$ denote weight and SVM kernel respectively; ρ denotes the maximal margin (threshold), Eq. 3.1, with N instances

Algorithm 9: Adaptive Hybrid Heterogeneous IDS

Initialisation
A stream of pair (x, y), as $(x_0, y_0), (x_1, y_1) \ldots (x_T, y_T)$, arriving one-by-one over time.
X is an evolving data stream ($X \rightarrow \infty$), where x_t is a set of features observed at time t (now).
y is the real class label and $\overline{y}$ is the classifier prediction, where $Y = \{-1, 1\}$
C_A: $C_{OCSVM} \cup C_{HST}$ // Anomaly Classifiers.
C_{OCSVM}: One-class SVM Classifiers $\subseteq C_A$.
iTree: a HalfSpace-Tree.
ω: Window Size.
A_{Score}: Anomaly Score.
C_{HST}: HalfSpace-Trees ensemble classifier $\in C_A$.
M is the number of models in the ensemble.
h_m is an adaptive OzaBagging ensemble model induced by learners $m \in \{m_1 \ldots m_n\}$.
Count $\leftarrow$ 0.
r: mass profile of a node in the reference window. //mass is used as a measure to rank anomalies.
l: mass of a node in the latest window.
k: Generate poisson $(\lambda = 1)$
ψ: is the generalised Kronecker function: $\psi(a, b)$ is 1 if $a == b$, and 0 otherwise.

for *all (x) in X* **do**
  $\delta \leftarrow$ using Eq. 3.3 c classifies (x_t), where $c \in C_{OCSVM}$
  if *δ == -1 (c has classified (x_t) as malicious)* **then**
    predict $\overline{y} = arg\ max_{y \in Y} \sum_{m=1}^{M} \psi(h_m(x_t), y)$
    for *all $m \in h_m$* **do**
      $\hat{w} \leftarrow exp(-1)/k!$
      Update m with (x_t, y_t) and weight $\hat{w}$
    if *$\overline{y}$ == -1 (h_m detect (x_t) as normal)* **then**
      $A_{Score} \leftarrow 0$
      for *all iTree in C_{HST}* **do**
        $A_{Score} \leftarrow A_{Score}$ + Score(x_t, iTree) // accumulate scores
        UpdateMass(x_t, iTree.root, false) // update mass l in iTree
      Report A_{Score} as the anomaly score for x_t
      Count++
      if *Count == ω* **then**
        Update model : Node.r $\leftarrow$ Node.l for every node with non-zero mass r or l
        Reset Node.l $\leftarrow$ 0 for every node with non-zero mass l
        Count $\leftarrow$ 0
    if *ADWIN detects change in error of one of the models (h_m)* **then**
      Replace the model with highest error with a new model
  Output: *Notify administrator if x_t is anomalous*

$\overline{x}_{i\in<1,N>}$. According to [137], the OCSVM can be solved efficiently using the quadratic Eq. 3.2. The ν (Nu) is upper bounded by the fraction of outliers and lower bounded by the fraction of support vectors. The ν intends to fine-tune the trade-off between over-fitting and generalisation. The conjoint usage of ν and the slack variable ξ ($\xi \geq 0$) enables the system to mitigate a dataset that contains a small fraction of outliers. In other words, ν is the probability of finding an outlier in X, where $outliers \subseteq X$. The γ (gamma) determines how much influence a single training example has. The larger γ is, the closer other examples must be to be affected. Since it is expected that ANIDS agents will generate some false-positive alarms (wrongly classify safe instances as intrusions), the instances that are classified as anomalies will be further analysed by CIDS.

$$\overline{w}.\phi(\overline{x}_i) \geq \rho - \xi_i \; \forall \overline{x}_i \in X \; and \; \xi_i \geq 0, \forall \; i \in \{1, ..., N\} \tag{3.1}$$

$$Min_{\overline{w},\overline{\xi},\rho} = \left[\frac{1}{2}\|\overline{w}\|^2 + \left(\frac{1}{\nu\gamma}\sum_{i=1}^{n}\xi_i\right) - \rho\right] \tag{3.2}$$

$$y_i = sign(\overline{w}.\phi(\overline{x}_i) - \rho) \tag{3.3}$$

where the y_i in Eq. 3.3 is an inliner (+1) if $\overline{w}.\phi(\overline{x}_i) - \rho \geq 0$ and an outlier (-1) otherwise.

3.2.2 Central IDS

The CIDS contains an incremental heterogeneous hybrid IDS and is responsible for analysing internal and external data streams. Experiments 2 and 3 (in Section 3.3.2) show that the incremental ensemble of OZABagging with KNNADWIN learners and HalfSpace-Trese (HS-Trees) [138] create a hybrid IDS that provides excellent performance in detecting intrusions. Concept-drift Detection (CD) enables the framework to enhance its intrusion detection performance over time by adapting to unforeseen intrusions and data distributions. The outcomes of experiments 4, and 5 show that the adaptive sliding window (ADWIN) CD algorithm [121] enhances the performance of the proposed scheme while using limited processing and memory at any point in time.

Oza Bagging Ensemble Classifier

Incremental ensemble classifiers provide better detection performance at the cost of more computation and memory usage [128]. An ensemble classifier $f(C_1(x_t), C_2(x_t)...C_n(x_t))$ is a set of classifiers (C_i) that make predictions over a given instance of feature set (x_t). The Ozabagging classifier builds an ensemble of classifiers such that $\forall \; c_i \in C$, c_i is trained over different bootstrap instances. Since it is challenging to draw samples with replacement in an online streaming environment, the Oza bagging classifier weights the observed instances using a Poisson[1] in bootstrap replica [141]. The OZABAGADWIN [142; 143] is the OzaBagging with ADWIN (adaptive windowing) concept-drift detection. The OZABAGADWIN implements several ADWIN drift detectors to monitor classifier error rates. On the detection of

[1] As N (number of samples) $\rightarrow \infty$ the distribution of K (number of copies of each n) tends to a Poisson(1) distribution: $K \sim \frac{\exp(-1)}{k!}$ [139; 140]

concept drift, OZABAGWIN replaces the worst classifier $c_i \in C$ with a new classifier, described as a "replace the loser" strategy [141]. The classification of the majority of individual classifiers that make up the ensemble is taken as the classification of the instance. Where the number of classifiers is odd, there is always a majority for one class. Where the ensemble has an even number of classifiers, then a tie is possible. In such a case, the instance is judged to be malicious [142; 143].

Incremental ensemble of anomaly-based IDSs

Although adopting adaptivity (concept-drift detection) enables a signature-based IDS to learn unforeseen intrusions (discussed in Section 3.2.2), a signature-based IDS (OzaBagADWIN) is prone to some degree of false-negative alarms for unknown intrusions. hence, to enable the proposed framework to identify unknown intrusions, the HalfSpace-Trees (HS-Trees) algorithm [138] analyses observations that are classified as normal so far. In HS-Trees, each tree contains nodes that capture the number of data items (known as mass) within a subspace of streaming data. In this context, the mass is used to profile the degree of anomaly. The OzaBaggingADWIN and HS-Tree form an incremental hybrid IDS on the 6BR.

Adaptivity

Adaptive learning updates the predictor model to respond to the concept drift through the predictor operations. The 6LoWPAN traffic routing evolves as nodes move or become unavailable (e.g. their energy resource may deplete), which results in reconstruction of the DODAG routing graph. Data forms a stream into the IDS with a distribution that varies over time. To reduce memory use, concept-drift-based IDS trains over a small number of training data at any point in time and does not load the entire dataset into memory [120]. The fundamental function of any concept drift detection approach is the mechanism to detect the drift occurrence timestamp. Accurate identification of the time that drifts happen plays a vital role in enhancing the system's adaptivity performance. Since the model never has full access to the entire data in a continuous environment, our proposed scheme employs the adaptive sliding window (ADWIN) concept [144] to perform concept drift detection. A window w is a snapshot of data; it gives more importance to the recently observed data and periodically discards the older data. ADWIN slides a window w on the prediction results as they become available, in order to detect drifts. The method examines two sub-windows of sufficient length, i.e., w_0 of size n_0 and w_1 of size n_1 where $w_0 \bullet w_1 = w$. The symbol $\bullet$ represents the concatenation of two windows. A significant difference between the means of two sub-windows indicates a concept drift, i.e., when $|\hat{\mu}_{w_0} - \hat{\mu}_{w_1}| \geq \varepsilon$ where $\varepsilon = \sqrt{\frac{1}{2m} \ln \frac{4}{\delta'}}$, m represents the harmonic mean of n_0 and n_1, and $\delta^{'} = \delta/n$. Here δ is the confidence level while n is the size of window w. Once a drift is detected, elements are removed from the tail of the window until no significant difference is observed. Algorithm 9 shows the proposed scheme.

Table 3.2: *Simulation parameters*

Parameters	**Values**
Number of nodes	16, 32, 64, 128
Number of Malicious nodes	∼10%, ∼20%, ∼30%
Number of Workstations	4, 8
Transmission Range	50m
Number of ML detectors	∼10%
Number of Mobile nodes	∼20%
Scenario Dimension (Terrain)	(250 × 250) to (850 × 850) s.meters
Traffic Rate	250 kbps
Simulation time	∼ 21,600 seconds
Application Protocols	COAP, CBR
RPL mode	Storing mode
Mobility Modes	Random Walk, Group Walk
Path Loss Model	Log Distance, Exponent(n): 2
Distance between nodes	25 ∼ 45 m
Objective Function (OF)	OF0, LQ
Receiver Sensitivity	-85 dBm

3.3 Implementation and evaluation

In this chapter, we use the Netsim simulator to evaluate the performance of the proposed scheme against different RPL attacks. In this context, we consider different network configurations (e.g. number of malicious and legitimate nodes, and objective function), as described in Table 3.2. The simulated 6LoWPAN scenarios include 16 to 128 LLN nodes (excluding 6BR and external computers), where 10% to 30% of the nodes are assigned as malicious (we round a decimal number[1]). In all scenarios, we consider 20% of the nodes, including half of the malicious nodes, walk around the terrain with a velocity of 5 m/s. Nodes distribute over terrain covering $250m^2 \sim 800m^2$ and are 25∼45 meters apart, *with 50 metres transmission range*. Each scenario is simulated for ∼480 minutes for performance benchmarking. This chapter uses the interleaved test-then-train approach to evaluate the proposed scheme [143]. It is assumed that the packets in the streaming data D sequentially appear in the target network, where x_t is an unlabeled instance vector observed at time t, containing different attributes about the node configurations and the DODAG. The actual label y_t of instance x_t will be available to the system at different points in time. In the continuous data environment like 6LoWPAN, the ground truth y_t may not be available immediately before observing x_{t+1}, and it may be available at some point in future [120]. Additionally, the observations of the data stream in the 6LoWPAN are independent. That means there is no relation between (x_{t+1}, y_{t+1}) and (x_t, y_t).

[1]if the last digit (the number of tenths) is less than 5, a rounding down is carried out. Otherwise, if it is 5 or above, rounding up is carried out

3.3.1 Data-set and feature construction

The simulations generate a dataset D, representing the malicious and normal (safe) network communications. Each observation x in D denotes a set of n features $x = \{f_1, f_2...f_n\}$, where f_i contains specific information about the sender and receiver. The header of each RPL control packet (e.g. DIO, DIS, DAO) contains different information about the sender of the packet [1; 2] that can facilitate the identification of anomalous network activities.

Engineering a set of informative features is essential to develop an IDS to accurately classify all types of RPL attacks in the streaming data environment. Therefore, in this chapter we perform feature engineering to facilitate the classification of data streams for IDS. The extracted features can enable the anomaly-based classifiers to correctly identify the anomalies through training over normal instances and make signature-based classifiers to accurately classify each type of RPL attack. The raw instances of 6LoWPAN simulations contain a set of features that are not applicable for conducting intrusion detection tasks. For instance, features that represent node identities (e.g. IP address, MAC address, and node id) can inhibit scheme generalisation. Since this chapter employs a passive decentralised monitoring approach [84], any feature that requires the internal configuration of legitimate nodes (e.g. power consumption, geographical location, CPU/RAM/ROM usages) are excluded. We simulated several pairs of networks $(\mathcal{A}, \mathcal{B})$ where $\mathcal{A}$ contains only the normal nodes and B contains both the normal and malicious nodes. Observing the statistical difference of control and application packets in $\mathcal{A}$ and $\mathcal{B}$ enables us to identify the adverse impact that each RPL attack has in the networks in $\mathcal{B}$. A simulated 6LoWPAN includes legitimate (safe) network communications (control and application packets) and malicious traffic. In each RPL attack scenario, malicious nodes cause adverse impacts inside the network by either generating malicious network traffic (e.g. DIS flooding, DIO suppression, and sinkhole attacks) or modifying legitimate network communication of their neighbouring nodes (blackhole and grayhole attacks). Witnessed system features which differ considerably between the two types of network can be used as indicators of malice, i.e., they are information rich features for such purposes.

We extract three types of features: basic, time-based, and connection-based features. Basic features contain general node information derived from ICMPv6 control packet headers (node rank, source and destination addresses, flags etc). Time-based features provide information about the number of times that the current node sends or receives a specific type of application or control packet. Connection-based features carry salient information about the sender's routing configuration (RSSI, link quality etc) and the number of collided control and application packets perceived by an IDS detector. The connection- and history-based features play vital roles in detecting the routing attacks in 6LoWPAN. Table 3.3 depicts the set of features engineered.

3.3.2 Performance evaluation and discussion

As discussed in Section 3.2.1, the novelty or anomaly detectors of the proposed scheme are responsible for identifying anomalies by observing the control packets of their neighbours; if the current observation is identified as anomalous, it will be further analysed by the heterogeneous hybrid ensemble IDS on the 6BR. Below, different outlier detection, incremental ensembling, and concept drift detection algorithms are evaluated and their performance have been compared against different criteria. We seek the best combination to gain the optimal

Table 3.3: *Engineered features*

Feature	Description
pkt_type	Type of packet (DIO, DAO, DIS, App etc)
pkt_status	Packet status (Collided, Successful)
dio_count	No. of DIO advertised by sender
avg_hopcount	Average No. of hopcount (global view)
dis_count	No. of DIS unicasted/multicasted by sender
dao_count	No. of DAO unicasted by sender
daoack_count	No. of DAO-Ack unicasted by sender
neighbour_count	No. of neighbouring node
child_count	No. of children
avg_intpkt_time	Average delay between packets
rnk_alt_count	No. rank alteration
cmp_snd_prt_lq	Compare LQ of sender with its parent
snd_ctrl_count	No. control packet transferred by sender
cmp_lq	Sender LQ & rnk < Receiver LQ & rnk
rcv_dao_count	No. of DAO received by current node
rcv_dio_count	No. of DIO received by current node
rcv_dis_count	No. of DIS received by current node
rcv_daoack_count	No. of DAO-Ack received by receiver
trans_app_count	No. of application okts trans by sender
pkt_e2e_delay	Packet end-to-end delay
pkt_loss	Application packet loss ratio
cpkt_loss	Control packet loss ratio
src_rank	Sender rank in DODAG
adv_vn	Advertised version number
rx_sens	Average receiver sensitivity
tx_power	Average transmission power
rssi	Received signal strength indicator of sender
same_parent	Sender has same parent as a detector node
rcv_cpkt_count	No. of control packets received by receiver
prt_bst_lq	Current parent provide best link quality

F1, accuracy, recall, precision [3] and kappa [128; 121] with least False Negative Rate (FNR) and False Positive Rate (FPR) [3]. In this context, we conduct *six* experiments utilising the underlying features of the Netsim emulator to execute the proposed framework over several Raspberry Pi 4 (model B, 4GB RAM) micro-controllers, to measure the execution time, and measure the power consumption using a UM25C digital multimeter.

Experiment 1. The anomaly-based detector (also known as novelty detector) plays a crucial role to identify outliers in the proposed scheme. Here we measure the performance of OCSVM in detecting RPL attacks. We have evaluated OCSVMs with different parameter values for Nu $\nu \in (0, 1]$ and Gamma $\gamma \in (0, 1]$ for finding the optimal configuration; Fig. 3.2 shows that the OCSVM with $\nu \in (0.01, 0.25)$ and $\gamma \in (0.6, 1]$ can maximise recall. However, since the aim of the ANIDS is to identify all the intrusions and maximise TPR, here we assign the OCSVM with $\nu = 0.2$ and $\gamma = 0.9$ to achieve 99.74% TPR with 89.39% recall (weighted average). Our experiments suggest that an OCSVM outperforms other existing anomaly detection algorithms, majority-voting ensemble of Local Outlier Factor and Isolation Forest, as shown in Fig. 3.3.

Table 3.4: *Performance bench-marking*

N	M	Accuracy								FNR							
		SH	***BH***	***GH***	***DA***	***IR***	***WH***	***DS***	***WP***	***SH***	***BH***	***GH***	***DA***	***IR***	***WH***	***DS***	***WP***
16	10%	91.5	91.8	96.2	99.8	95.8	98.3	97.4	98.6	14.1	13.8	3.4	0	7.3	2.4	2.6	2.7
	20%	98.7	95.4	98.4	100	97.9	96.5	98.7	97.5	1.8	5.4	3.0	0	4.0	4.4	2.3	2.9
	30%	97.6	97.0	96.6	100	94.1	99.6	98.2	99.5	3.4	5.4	5.8	0	11.3	0.1	2.9	0.2
32	10%	93.3	96.3	98.5	99.8	97.8	99.7	98.5	99.6	10.0	5.4	2.0	0.3	3.8	0.2	2.2	0.5
	20%	98.7	98.2	98.2	100	97.8	94.8	98.4	95.2	2.4	3.1	2.0	0	3.6	9.5	2.0	8.7
	30%	98.6	98.3	98.7	100	97.0	90.1	98.7	91.9	2.3	3.2	2.4	0	5.3	16.0	2.4	13.2
64	10%	92.5	93.1	90.6	99.9	94.9	91.6	89.5	92.7	13.8	13.2	16.8	0.1	9.1	12.0	18.9	10.6
	20%	93.0	93.4	96.2	100	94.9	91.0	97.0	96.3	11.0	11.4	6.7	0	8.4	10.7	4.9	6.7
	30%	93.7	93.8	96.2	100	96.4	94.5	98.7	96.6	11.5	9.4	7.1	0	5.0	10.1	2.4	5.9
128	10%	97.2	93.0	91.2	99.8	95.5	93.5	94.0	92.3	5.4	13.4	16.0	0.4	8.1	9.2	8.2	11.3
	20%	93.6	93.9	94.1	100	95.9	94.4	96.0	93.1	11.7	11.0	10.0	0	6.1	10.5	6.7	13.3
	30%	94.3	94.9	96.9	100	96.9	95.2	96.7	95.4	10.0	8.4	5.8	0	4.7	8.5	5.8	7.8

SH: Sinkhole, **BH:** Blackhole; **GH:** Grayhole; **DA:** DIS Flooding; **IR:** Increase Rank; **WH:** Wormhole; **DS:** DIO Suppression; **WP:** Worst Parent; **N:** No. Normal nodes; **M:** No. Malicious nodes;

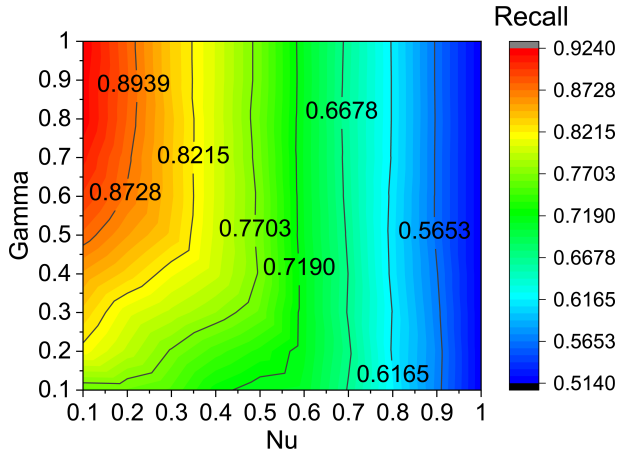
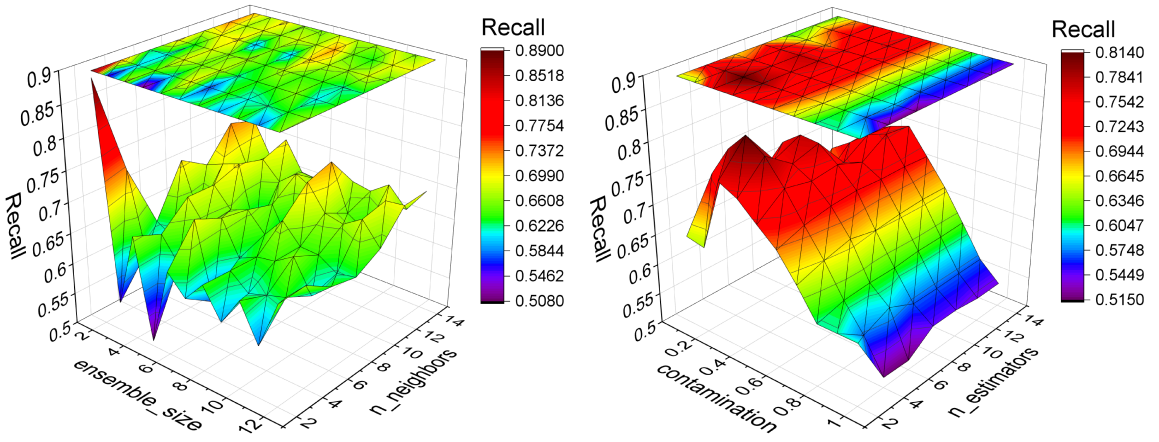


Figure 3.2: *OCSVM recall*

Experiment 2. Experiment 1 showed that although the OCSVM algorithm can accurately identify outliers it also incurs 20.25% FPR (i.e. it mis-classifies inliers to an unacceptable degree). To address this issue, we conduct our second experiment to measure the performance of different incremental ensemble algorithms to rectify ANIDS misclassifications. Here, we have compared the performance of OzaBagging [142], LearnPPNSE [145], Online Boosting [139], Online AdaC2 [139], Accuracy Weighted Ensemble [146], and Online SMOTE Bagging [139] algorithms in detecting RPL attacks. The outcome of our experiment (as shown in Fig. 3.5, Fig. 3.6 and Fig. 3.4) shows that the combination of OzaBagging using KNNADWIN provides the best possible means to identify known intrusions. OzaBagging using KNNADWIN with n_estimators as 4 and n_neighbours as 6 receives 91.5% F1 and 7.8% FPR and with n_estimators as 8 and n_neighbours as 6 receives 92.2% F1 and 7.3% FPR.

Experiment 3. Above, we showed how an incremental ensemble approach can identify known intrusions efficiently. Our proposed hybrid IDS targets both known and unknown



(a) *Local Outlier Factor Recall.* **(b)** *Isolation-Forest Recall.*

Figure 3.3: *Performance of different outlier detection algorithms.*

intrusions. Accordingly, we now investigate an incremental ensemble of anomaly-based classifiers which can rectify false-negative classifications of the signature-based IDS. False-negatives are very costly and indicate the IDS failing in its primary task. In this experiment, we show how the inclusion of an incremental HalfSpace-Trees (HS-Trees) classifier can enhance the overall performance of the system. Fig. 3.5 and Fig. 3.6 show that the HS-Trees algorithm forms a better hybrid IDS when it combines with the OzaBagADWIN compared to other incremental algorithms by around 6 to 10%. Fig. 3.5 and Fig. 3.6 give the current and moving mean (also referred to as moving average) F1, recall, kappa, and accuracy of the incremental ML algorithms.

Experiment 4. The concept drift detection method is the key mechanism for making the proposed scheme adaptive. Here, we investigate to what extent concept-drift detection can provide system adaptivity. We evaluate different drift detection algorithms to select one that can provide adaptivity in the system and enhance the framework performance over time. We consider the following (seven) concept-drift detection algorithms: Adaptive Windowing methods (ADWIN), Drift Detection Method (DDM), Early Drift Detection Method (EDDM), Kolmogorov-Smirnov Windowing (KSWIN), PageHinkley, Drift Detection Method based on Hoeffding's bounds (HDDM) with moving weighted average-test (HDDM-W) or moving average-test (HDDM-A) concept drift detection methods [135; 121]. Results are presented in Fig. 3.9 and Fig. 3.10. From Fig. 3.10 we can see that ADWIN gives the best accuracy of the concept-drift detection methods and does so in the shortest time interval (as shown in Fig. 3.9).

Experiment 5. In this experiment, we measure the time complexity of each component in the proposed framework. We consider 64 LLN nodes in 6LoWPAN, with 20% assumed malicious. Fig. 3.7 illustrates the outcome of our analyses over 1500 network packets, where 50% of instances are assumed normal and the remaining 50% include each RPL attack type equally.

ANIDS and CIDS are different components of our proposed scheme; we measure the time complexity for each component separately. Fig. 3.7 shows the time complexity that the OCSVM with $\nu = 0.2$ and $\gamma = 0.8$ causes the least time complexity in the system. On

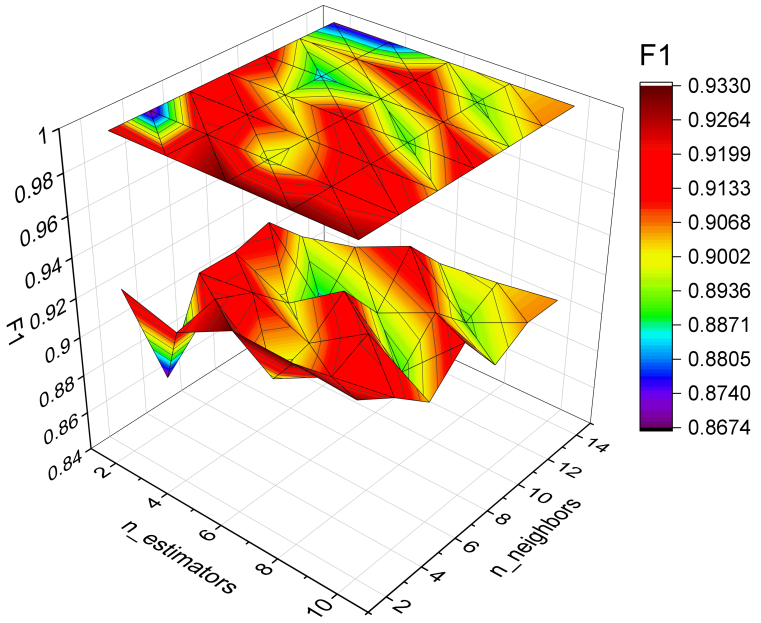


Figure 3.4: *OzaBagging ADWIN (KNN) F1.*

Table 3.5: *Time complexity.*

Comp	Training (sec)	Testing (sec)
ANIDS	$O(n)$: 0.36 + -2.4E-08*n	$O(log(n))$: 0.22 + -0.0021*log(n)
CIDS	-	$O((log\ n)^k)$: -2.3 * $\log(n)^{0.94}$

the other hand, the adaptive heterogeneous hybrid IDS, developed in our Experiments 2 and 3, using 4 learners and 8 neighbours (KNN) causes $O(log(n))$ time complexity in the system. Table 3.5 shows that ANIDS has linear and logarithmic time complexity in training and testing, while CIDS has polynomial time complexity in the proposed scheme. In order to measure the power consumption of each component, in this experiment here we use the Netsim Emulator feature to connect the physical microcontrollers with the simulation environment. By connecting digital ammeters to the microcontrollers, we measure the energy consumption of both ANIDS and CIDS. In this regard, we run our experiment for 10 minutes by disabling all the unnecessary background tasks and applications. At the end of the experiment, we find that the energy consumption of an ANIDS and the CIDS in a LLN with 64 nodes was 3.505 J/s and 3.754 J/s, respectively, whilst a legitimate node without any AIDS or CIDS consumed 3.17 J/s.

Experiment 6. Our sixth experiment comprises two sub-experiments, where we first evaluate the performance of the proposed scheme to detect each RPL attack in LLNs with different proportions of legitimate and adversarial nodes, as shown in Table 3.4. In the final part of Experiment-6, we consider the detection of unforeseen intrusions (shown in Table 3.6),

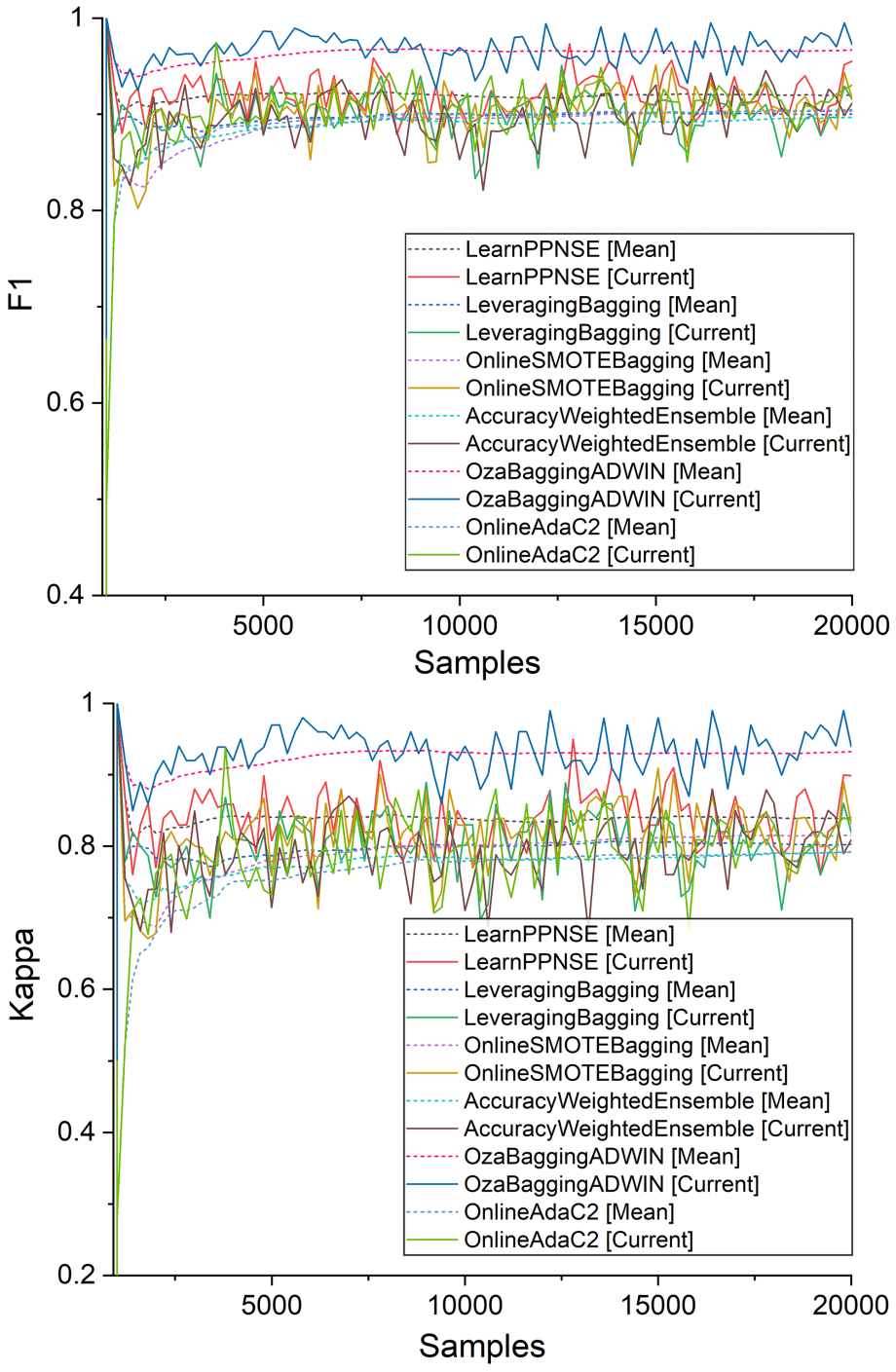


Figure 3.5: *Performance of the proposed scheme in detecting RPL attacks (F1 and Kappa) in streaming data environment*

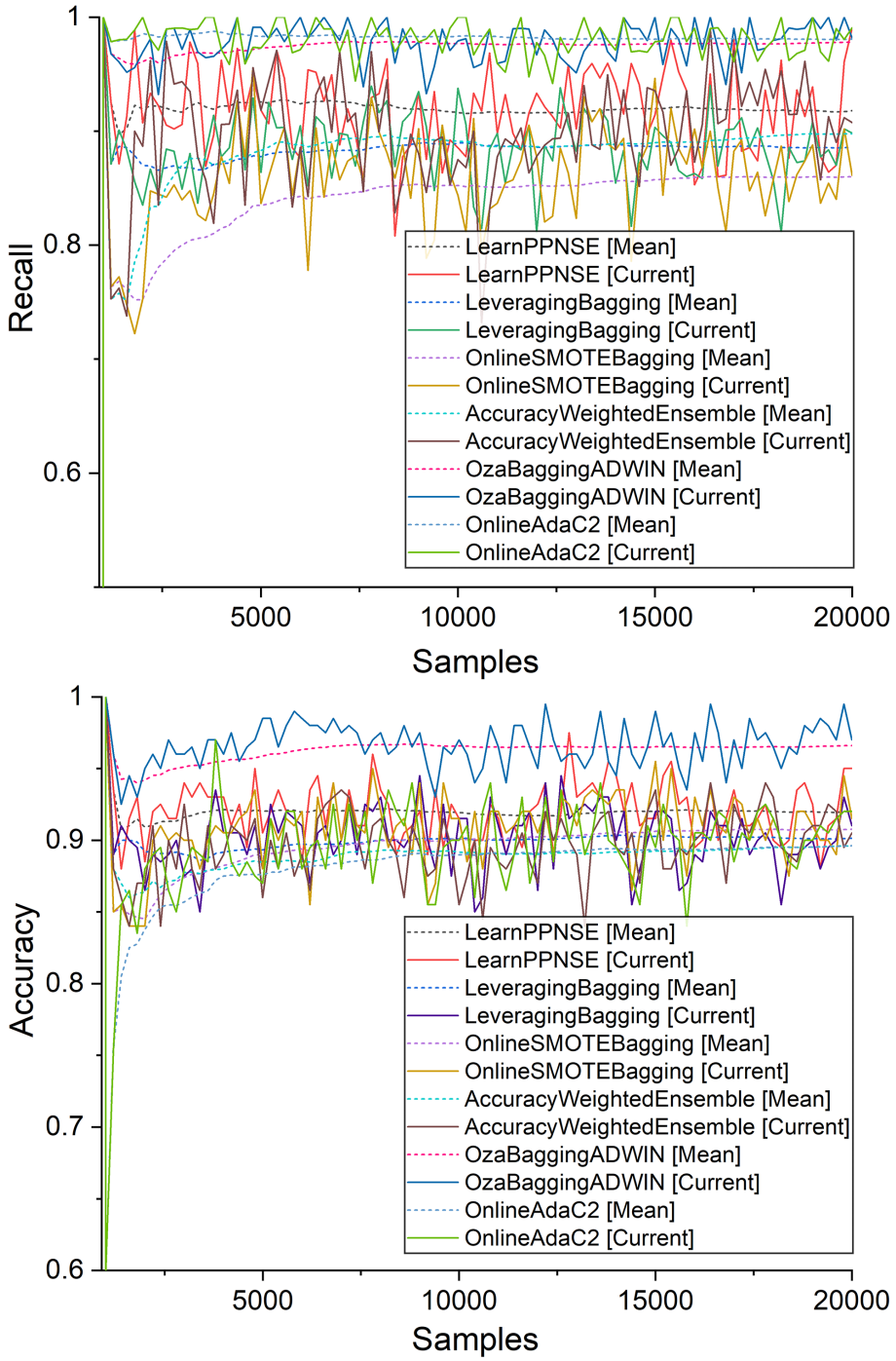


Figure 3.6: *Performance of the proposed scheme in detecting RPL attacks (Recall and Accuracy) in streaming data environment*

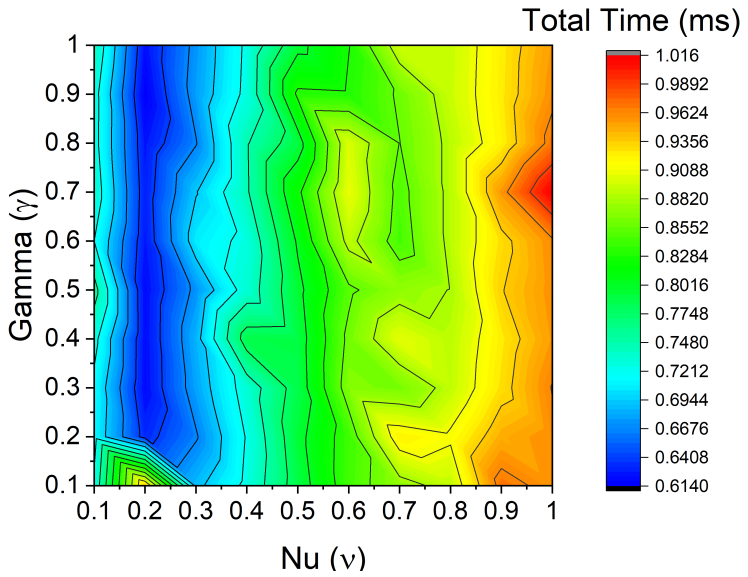


Figure 3.7: *One-Class SVM time complexity.*

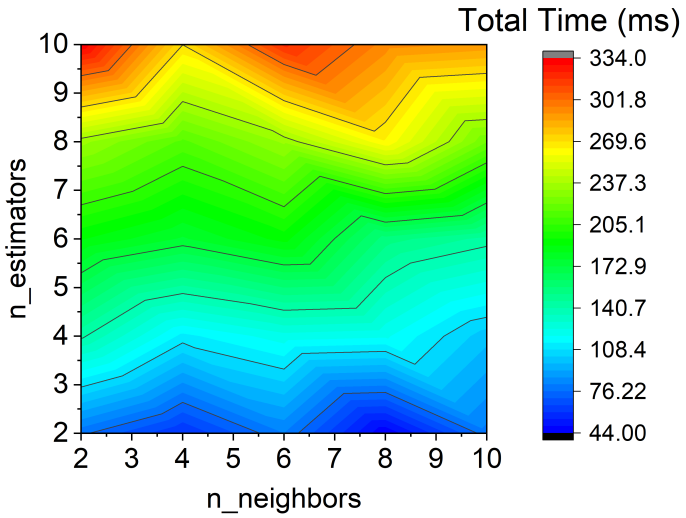


Figure 3.8: *CIDS time complexity.*

where each RPL attack was excluded from the pre-training data one-by-one and exclusively covered all the adversarial activities of the evaluation data stream.

Routing threats in 6LoWPAN and threats against RPL are highly significant. In this chapter, we have introduced an adaptive hybrid heterogeneous IDS scheme that is effective and efficient and can readily cope with changes to the environment and detect known and unseen routing intrusions in the 6LoWPAN. Table 3.7 gives an *indicative* comparison between our scheme and the results obtained by other authors. However, we stress our results are obtained in a much more challenging environment. We provide our results here as a benchmark for the research community.

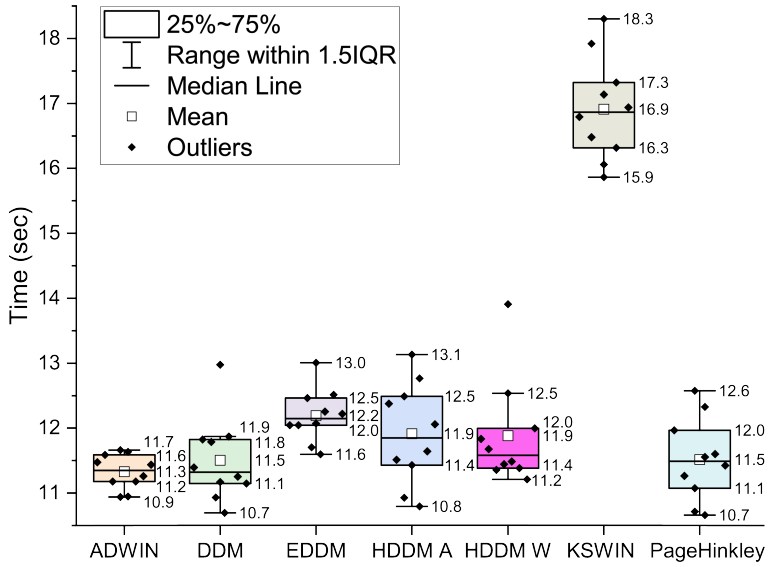


Figure 3.9: *Comparison of concept-drift detection methods.*

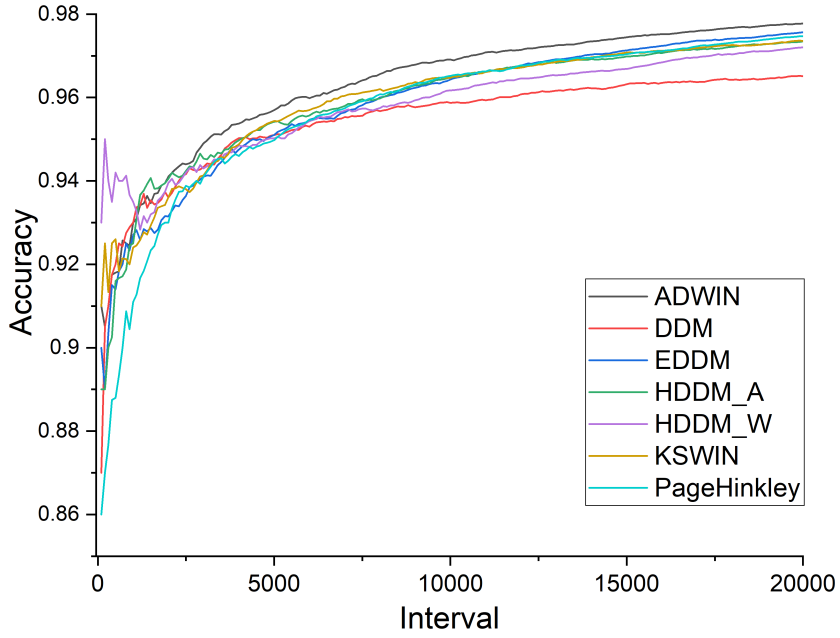


Figure 3.10: *Concept drifts comparison (accuracy).*

3.4 Summary

Due to the global connectivity, limited resources and RPL vulnerabilities, the 6LoWPAN is exposed to various routing threats internally (within the 6LoWPAN) and externally (through the Internet). The existing routing attacks (e.g. blackhole, grayhole, wormhole, DIS flooding) [3] cause the RPL to generate the suboptimalise routing topology, isolate legitimate nodes, and cause significant overhead over the target network and nodes to endanger confidential-

Table 3.6: *Unknown attack detection*

Unknown Attack	Performance Metrics				
	Accuracy	***Precision***	***F1***	***TPR***	***FPR***
SH	90.85%	91.16%	90.79%	86.52%	5.17%
BH	89.75%	90.30%	89.74%	83.62%	3.55%
GH	93.9%	94.07%	93.88 %	90.97%	3.31%
IR	91.75%	92.20%	91.71%	86.61%	3.25%
DA	98.30%	98.36%	98.29%	96.57%	0%
WH	98.35%	98.36%	98.34%	97.04%	0.30%
DS	93.95%	94.05%	93.94%	91.62%	3.76%
WP	95.10%	95.18%	95.09%	92.93%	2.71%

Table 3.7: *Performance bench-marking with offline IDS in 6LoWPAN*

Paper	No. Nodes	No. Malicious	Duration minutes	Mobility	RPL Attacks							
					SH	BH	GH	IR	DA	DS	WH	WP
[54]	5∼50	1∼5	20	No	100%	-	85.36% ∼92.68%	-	-	-	96% ∼97.53%	-
[90]	20∼40	1∼30%	30	No	-	-	-	-	100%	-	-	-
[62]	11	1	30	No	93.14%	93.14%	-	-	-	-	-	-
[94]	10	1	-	No	-	-	-	-	-	-	-	-
[99]	2∼10	1	480	No	-	-	-	-	-	-	-	-
[56]	8	1∼3	∼30	No	100%	-	-	-	100%	-	100%	-
[79]	8∼24	1∼2	30	No	-	-	-	-	-	-	94%	-
[22]	8∼64	1∼4	∼30	No	79%	-	81%	-	-	-	-	-
[117]	4∼8	2	-	No	90% ∼100%	-	-	-	-	-	-	-
[81]	10∼200	∼2	-	No	-	-	-	-	-	-	71% ∼75%	-
Proposed Scheme†	16∼128	10∼30%	360	Yes (20%)	91.5% ∼98.7%	91.8% ∼98.3%	90.6% ∼98.7%	94.1% ∼97.9%	99.8% ∼100%	94.0% ∼98.7%	90.1% ∼99.7%	91.9% ∼99.6%

*results indicate the accuracy of the proposed IDS in detecting each type of RPL attack;
†Details are shown in Table 3.4;

ity, integrity and availability (CIA) of the 6LoWPAN streaming data. In this chapter, we have introduced an adaptive hybrid heterogeneous IDS scheme that can identify RPL attacks accurately and adapt to changes/shifts in the network data stream promptly. The experiments have shown that our proposed scheme is effective, efficient, and agile compared to the related solutions. The proposed CIDS using the ADWIN concept-drift detection algorithm has the highest accuracy and shortest response time. Although the obtained results support *"hypothesis 1"*, the proposed scheme employs batch-trained OCSVM to identify anomalous activity inside the 6LoWPAN. Consequently, the failure of ANIDS (Anomaly-based NIDS) in identifying an attack may cause false-negative in the proposed scheme. Moreover, the anomaly-based IDSs are prone to false alarms (both false positive and false negative) on the occurrence of concept drifts (they may identify new/unforeseen data distribution as anomalous). In this chapter, we evaluate the performance of the proposed scheme in 6LoWPANs with the OF0 and LQ objective functions; hence, the performance of the proposed scheme on the 6LoWPAN using a different objective function (e.g. MRHOF, TAOF, and LBOF) is still unclear. The CIDS is the main decision-maker in the proposed scheme; all abnormal activities

detected by ANIDS should be transferred to the CIDS for accurate classification. Moreover, the CIDS is the only incremental ML-IDS in the proposed scheme. The lack of adaptivity in ANIDS detectors can reduce their performance on the occurrence of concept drifts. Since the proposed ANIDS (the first detection layer) provides 89.39% recall and 20.25% FPR, the false alarms may cause additional network overhead in the 6LoWPAN. Therefore, more accurate ML-IDS detectors are required to monitor the internal network communications of 6LoWPAN.

Chapter 4

Reinforcement-Learning-based IDS for 6LoWPAN

Protecting LLN nodes against RPL attacks is of critical importance. However, due to the computational limitation of the LLN nodes, sometimes it is difficult to adopt any highly promising leading-edge approaches (such as ML-based approaches). In the previous chapter, we discussed the requirements of an adaptive hybrid IDS that is placed on the border router to analyse alarms generated through the IDS agents located at nodes around the system. In this chapter, we present an RL-based IDS to enhance the intrusion detection performance in large-scaled 6LoWPANs, where the central IDS on the border router cannot perceive and analyse the entire network communications.

4.1 Related Works and Motivations

Below we classify the relevant research articles into three categories: IDS for RPL, ML-IDS for RPL, and RL for IDS. No extant research uses an RL-based IDS to detect RPL attacks. (Some studies use RL to enhance IDS performance against threats to different network technologies.) Researchers have investigated the detection of RPL attacks using signature-based, anomaly-based, and specification-based approaches, or a hybrid of those approaches. (For a survey of IoT-related IDS systems the reader is referred to [3]). Svelte [22] proposes a hybrid (signature-based and specification-based) IDS designed to monitor an LLN in a distributed manner, collecting traffic from nodes. As Svelte addressed only grayhole and blackhole attacks, the authors of [132] were encouraged to develop a specification-based IDS to detect Sybil and wormhole attacks. In [79] a different approach to detect wormhole attacks was taken, considering nodes to be equipped with GPS to transfer their location information to the centralised specification-based IDS. [79] and [62] use passive monitoring techniques to analyse LLN traffic and detect RPL attacks using a specification-based detection strategy. The limitations of specification-based detection strategies encouraged researchers to propose ML-IDS for mitigating RPL attacks. In [62] the use of various ML methods (Naïve Bayes, MLP, SVM, and Random Forests) was investigated to detect version number, sinkhole, blackhole, Sybil, and decrease rank attacks targeting RPL using the MRHOF and OF0 objective functions (specific performance metrics the RPL routing algorithm seeks to optimise) [147; 3]. They evaluated their proposed hybrid IDS over a small-scale LLN with a single malicious node.

Similarly, [56] investigated different ML methods (J48 Decision Tree, Logistic, MLP, Naïve Bayes, Random Forest, and SVM) and propose a hybrid ML-IDS with passive monitoring to detect sinkhole, wormhole, and DIS flooding. The unsupervised K-means and supervised Decision Tree (DT) algorithms are used by [81] to develop a centralised hybrid ML-IDS capable of detecting the wormhole attack. The work of [54] uses unsupervised Optimum-Path Forest Clustering (OPF) to develop specification-based anomaly-based decentralised ML-IDS to detect wormhole, sinkhole, and grayhole attacks.

Extant research has not proposed using RL to ensure security in the 6LoWPAN network. However, there are several studies [148; 149; 150; 151; 152; 153] where RL is used to enhance IDS performance in detecting application-based attacks. [148] employs Q-learning and a centralised hybrid IDS to perform the detection task over the data received through cluster heads in the WSN. The work of [149] employs Deep RL (DRL) for developing a centralised anomaly IDS. In their proposed model RL is used to enhance anomaly IDS detection performance. Similarly, [150] investigates different RL methods, namely DQN, Double DQN (DDQN), Actor-Critic, and Policy Gradient (PG), to improve the performance of a supervised anomaly-based IDS over the training phase. Enhancement of IDS performance using an adversarial RL training environment has been used by [151], [152]. In [152], researchers employ distributed DRL to boost IDS performance and prepare it against adversarial attack. The authors of [153] investigate the use of model-free Q-learning in intrusion detection using the NSL-KDD dataset. Table 4.1 describes the related works in the literature and the contributions that each article makes.

4.1.1 Our contribution

In this chapter, we introduce a new RL-based IDS (RL-IDS) that utilises heterogenous ML-based IDSs over the 6LoWPAN. A variety of internal (inside 6LoWPAN) and external (over the Internet) RPL attacks (Sinkhole, Blackhole, Grayhole, DIS flooding, Wormhole, DIO Suppression, Increase Rank, and Replay) are handled by our proposed approach. In this chapter we:

- propose an RL-IDS to enhance the strength of distributed ML-IDS in detecting internal and external RPL intrusions.
- engineer a set of features and correlate its elements with the effects each RPL attack has on an LLN.
- evaluate different supervised and unsupervised ML algorithms and develop a hybrid ML-IDS approach better suited to detection of known and previously unseen malicious activities and attacks.
- propose for the first time an IDS to detect Increase Rank (IR) , DIO Suppression (DS), and Replay attacks [15; 3].
- address for the first time attack scenarios with malicious mobile nodes.
- address for the first time both individual and combinations of RPL attacks.
- evaluate the performance of the proposed scheme in various scaled LLNs with respect to different numbers of malicious nodes.

Table 4.1: *Related works*

Scheme	Method	Attack	Desirable Imperative Features for IDS						
			DP1	*DP2*	*DP3*	*DP4*	*DP5*	*DP6*	*DP7*
[132]	Specification-based IDS, Highest Rank Common Ancestor	Wormhole and Sybil (Cooja)	×	×	✓	×	×	×	×
[99]	Specification-based IDS with a passive decentralised monitoring system Threshold-based	DODAG Inconsistency (Cooja)	×	×	✓	×	×	×	×
[79]	Specification-based IDS (requires geographical information of nodes)	Wormhole (Cooja)	×	×	×	×	×	×	×
[22]	Hybrid	Sinkhole and Grayhole (Cooja)	×	×	×	×	×	×	×
[62]	ML-IDS using voting technique (MLP, Random Forest)	Version, Rank, Sybil, Decrease Rank, Blackhole (Cooja)	×	×	×	✓	×	×	×
[56]	Hybrid ML-IDS using passive monitoring technique J48 Decision Tree, Logistic, MLP, Naïve Bayse, Random Forest, and SVM	Sinkhole, Wormhole, and DIS Flooding (Cooja)	×	×	×	×	×	×	×
[81]	Hybrid ML-IDS (Unsupervised K-means and supervised Decision Tree)	Wormhole attack (unknown C++ platform)	×	×	×	×	×	×	×
[54]	Anomaly ML-IDS Unsupervised Optimum-Path Forest Clustering (OPF)	Sinkhole, Grayhole, and wormhole (unknown C platform)	×	×	×	×	×	×	×
[150]	Anomaly-based IDS using RL in training phase. Experiment different RL algorithms (DQN, DDQN, Actor-Critic, and PG)	NSL-KDD and AWID	✓	×	D/N	D/N	×	×	D/N
[148]	Use RL Q-learning algorithm to develop centralised hybrid IDS in WSN	KDD Cup 1999	✓	×	D/N	D/N	×	×	D/N
[149]	Centralised anomaly-based IDS using Deep RL (DRL)	NSL-KDD and UNSW-NB15	✓	×	D/N	D/N	×	×	D/N
[153]	RL (Q-learning) based IDS	NSL-KDD	✓	×	D/N	D/N	×	×	D/N
[151]	Use DRL and Q-learning to enhance IDS performance through the adversarial training procedures	NSL-KDD and AWID	✓	×	D/N	D/N	×	×	D/N
[152]	Distributed DRL for IDS	NSL-KDD, UNSW-NB15 and AWID	✓	×	D/N	D/N	×	×	D/N
Our scheme	RL-based heterogeneous hybrid IDS	SH, BH, GH, IR, RA, DA, WH, DS	✓	×	✓	✓	✓	×	✓

* In the "Attack" column, the later entries refer to available datasets that contain a variety of attacks (but these exclude RPL attacks);
✓: Satisfy; ×: Not addressed; *D/N: Different Network-technology;

The rest of this chapter is organised as follows. In Section 4.2 we indicate how informative features are developed and selected. In Section 4.3, we describe the RL-based intrusion detection scheme, ML-based detectors, and the development of a flexible system using RL algorithms. In Section 4.3.3, the simulation setup is described, the experiments are carried out, and results reported. Finally, concluding remarks and analysis of results are given in Section 4.5.

4.1.2 Reinforcement learning

Reinforcement learning is an important area of machine learning that enables an agent to interact with its environment and learn through a trial and error process by receiving feedback from the actions it takes. Specifically, it helps an agent/decision-maker learn the system's dynamic through observations and interactions with the environment. The environment is everything outside the agent. The agent receives the observation (current state s_t) and the

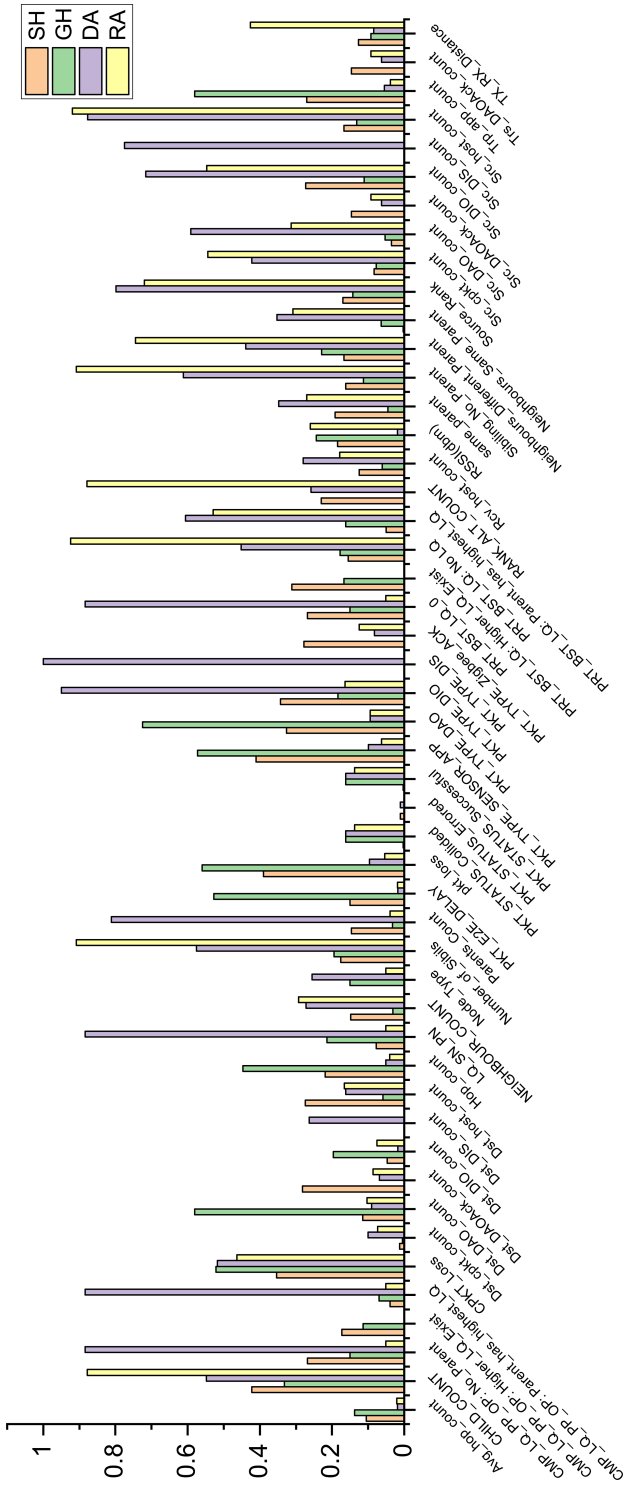


Figure 4.1: *Features' correlations (sinkhole, grayhole, DIS flooding and replay attacks).*

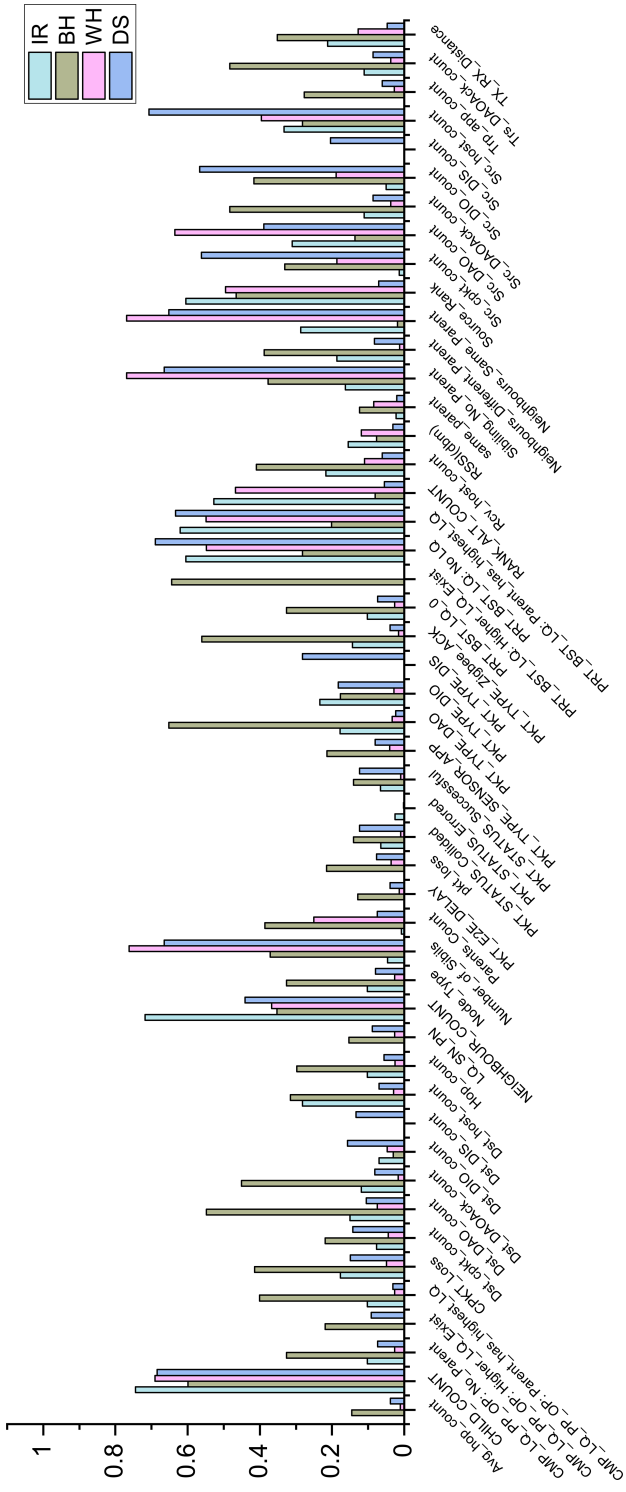


Figure 4.2: *Features' correlations (increase rank, blackhole, wormhole and DIO suppression).*

reward (r_t) from the environment at each iteration and follows its action value-function (Q) to take the action that increases the long-term reward. The action is the thing that agent can do in the environment given it is in the current state. The action value-function $q_\pi(s_t, a_t)$ informs the agent how taking the action a_t is good (in terms of expected return) at the given state s_t while following policy π. The reward (r_t) can be positive or negative (penalty) and indicates to the agent how well the agent has behaved.

In RL a transition function can be formulated as a Markov Decision Process (MDP), a mathematical framework for modelling sequential decision-making. An MDP characterises the agent interaction with its environment in a sequential decision-making process; the environment computes transition and rewards, and the agent generates the policy. The policy π is probability distribution that forms the behaviour of the agent. Formally, π is defined as $\pi(a|s) = P[A_t = a|S_t = s]$. This is Markovian (Markovian means memoryless) because the actions depend only on the current state, not how the system got into that state.

There are different approaches for computing policies and value-functions, namely look-up tables and approximation methods [150]. Since 6LoWPAN has a continuous environment, using a look-up table would be a highly resource-intensive task. Therefore this chapter uses the DQN and DDQN approximation methods.

4.2 Feature Engineering

The data elements that feed into our decision making algorithms are generally referred to as 'features'. Obtaining sets of high performing informative features is generally referred to as Feature Engineering (FE). We have identified a variety of potential features and determined how correlated they are with the effects of the various RPL attacks considered. In order to illustrate the importance of engineered features in classifying each RPL attack, here we employed the Pearson Correlation Coefficient's absolute value, as depicted in Fig. 4.1 and Fig. 4.2.

Enhancing algorithm accuracy and interpretability is the main aim of feature selection methods [154]. Feature selection may improve accuracy and efficiency. Feature selection reduces the memory footprint necessary for storing and executing the models and storing the raw data to a lesser degree. Similarly, it can reduce run-time, both during training and prediction. This study employs feature selection methods for constructing and selecting subsets of features to generate a good predictor.

In normally distributed and categorical data, the predominant advice is to use Chi-Square. Mutual information and Gini Impurity are also reasonable options to consider. The Analysis of Variance (ANOVA) works well for categorical features (independent variables) and a continuous target (dependant variable); Pearson's R2 works well for continuous features and a continuous target.

Since the RPL traffic dataset contains both continuous and categorical features and a categorical target, we use filter method feature selection Chi-square, Gini impurity to reduce the feature set's size and make it less costly in terms of time and computational resources. The Wrapper feature selection methods are computationally expensive [154]; therefore, this study avoids implementing such methods. Based on our experiments, chi-square is fast and can avoid over-fitting while it is computationally inexpensive compared to other feature selection methods.

The Chi-square (X^2) [155] is a statistical filter method that measures the deviation from the expected distribution considering the feature event is independent of the target value. X^2 measures how expected count (E) and observed count (O) deviate from each other Eq. 4.1. The intuition is that if the feature is independent of the target, it is uninformative for classifying observations. The O_{ij} is the observed count for the cell in row i, column j. The E_{ij} uses the Eq. 4.2.

$$X^2 = \sum_{i,j} \frac{(O_{ij} - E_{ij})^2}{E_{ij}} \tag{4.1}$$

$$E_{ij} = \frac{(ith\ row\ sum) \times (jth\ column\ sum)}{grand\ total} \tag{4.2}$$

4.3 Proposed Scheme

In this section, we present the proposed IDS methodology for 6LoWPAN networks. Since LLN nodes have limitations in terms of the computational resources, hence they cannot afford the computational requirements of extensive ML algorithms. This chapter seeks to address the above issue by proposing an RL-based intrusion detection scheme that uses several lightweight ML-based detectors for analysing 6LoWPAN traffics. Each ML detector trains over a subset of the training data that includes different proportions of the different attacks. Therefore each detector may have various strengths and weaknesses in detecting the various RPL attacks. The proposed method uses an RL algorithm to identify the appropriate detector for analysing current network terrific. Fig. 4.3 illustrates the proposed scheme design. Details of the Fig. has been discussed in next two sections.

4.3.1 ML-based intrusion detection

Machine learning (ML) is an intelligent method that optimises system performance using sample data. More precisely, ML classification algorithms build models of a problem by applying mathematical techniques on sample data sets. The sheer amount of data generated in an LLN can make ML bring intelligence to the system for various purpose, including security. ML algorithms are mainly supervised and supervised methods. (In supervised approaches data is labelled with its actual class. In unsupervised approaches it isn't.)

The number of features, training samples, and parameters of ML algorithms play vital roles in defining classifiers' complexity over training and prediction phases. A higher number of features and training data increase algorithm complexity significantly and may cause an adverse effect on model generalisation. Although increasing the ML algorithms' sensitivity (assigning higher depth in the decision tree, C value in SVM, smaller k in KNN etc.) may enhance model detection performance, it increases the model's complexity dramatically and leads to over-fitting [4]. Table 4.2 shows the complexity (O) of different ML classification algorithms [4].

This research employs both signature-based and anomaly-based IDS (hybrid IDS) [3] to detect known and unknown intrusions efficiently. The RPL attack detection ability of various supervised and unsupervised ML algorithms is investigated, Fig. 4.7. Some of these ML

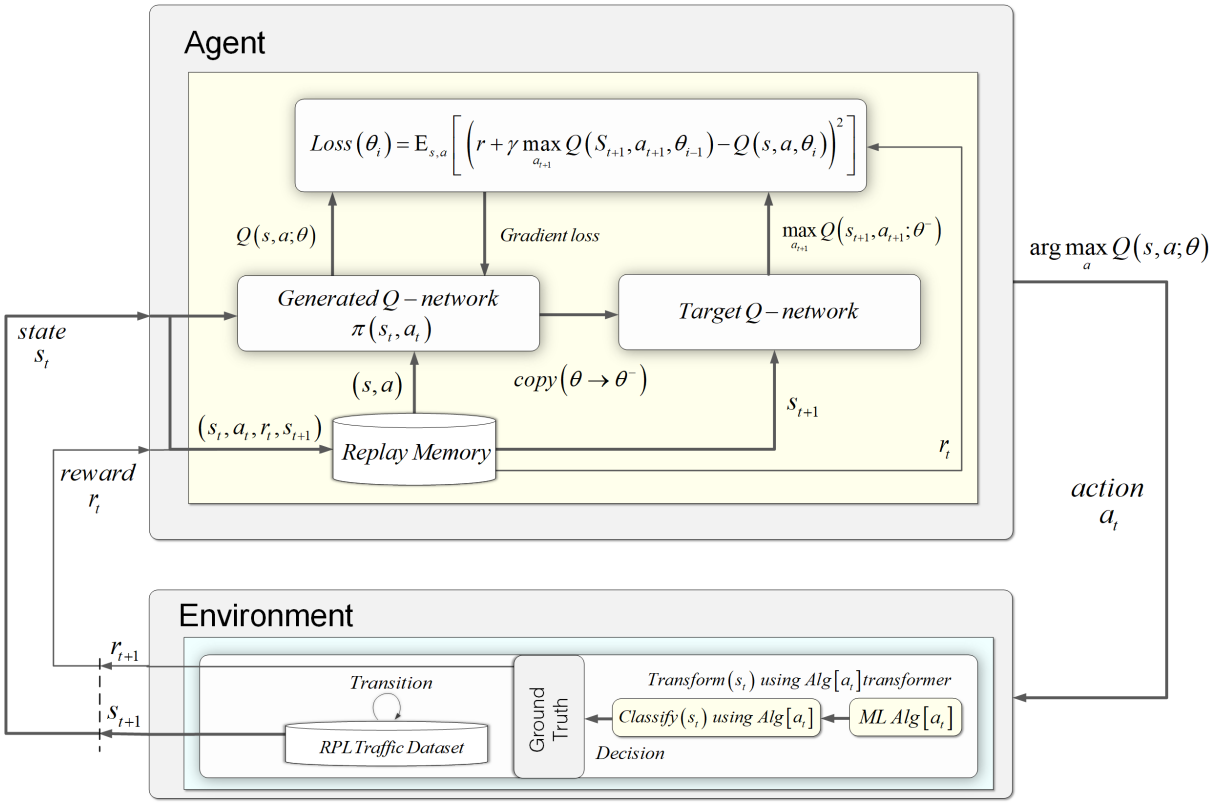


Figure 4.3: *RL-IDS.*

algorithms provide a slightly better performance, but this comes with the cost of greater computational complexity and resource usage that many LLN nodes cannot afford [4]. Since IoT has heterogeneous nodes with different computational resources, this research picks various ML algorithms over the LLN to analyse RPL's communications.

4.3.2 Reinforcement learning-based IDS

Supervised and unsupervised ML algorithms mainly focus on data analysis problems, while RL is preferred for comparison and decision-making problems [4; 150; 148]. Fast convergence, finding the action-value function $Q(s, a)$ and optimal policy (π^*) are the main challenges in implementing RL algorithms in a dynamic environment like LLN. The tabular RL methods, such as Temporal Difference, SARSA, and Monte Carlo, are exhaustive and inefficient methods for continuous environments that have large state space. The 6LoWPAN has a non-stationary (continuous) environment with an infinite number of states. Applying tabular methods reduces IDS efficiency and increases its computational needs since the agent will use a lookup table for taking action in each state. Therefore an RL approximation method is required to make the system generalise in the face of unforeseen states and reduce the system complexity. This chapter uses DQN and DDQN algorithms to find an optimum policy (π^*) that result in the maximum long-term reward (R). The policy (π) represents a probability distribution over actions given the current state (packet).

Table 4.2: *ML algorithms' complexity*

Algorithm	**Training**	**Prediction**
Decision Tree (DT)	$O(n^2p)$	$O(p)$
Support Vector Machine (SVM)	$O(p^2n + p^3)$	$O(n_{sv}p)$
k-Nearest Neighbours (KNN)	$O(np)$	$O(np)$
Gradient Boosting (GB)	$O(npn_{trees})$	$O(pn_{trees})$
Q-learning	-	$O(n^3)$
k-Means Clustering	Zero(negligible)	$O(n^2)$
Neural Network	-	$O((pn_l) + (n_{l1})(n_{l2}) + ..)$
Random Forest (RF)	$O(n^2pn_{trees})$	$O(pn_{trees})$
n = number of training samples; p = number of features; O = complexity; n_{trees} = number of trees; n_{sv} = the number of support vectors; n_l= the number of neurons at layer i in a neural network;		

The DQN and DDQN are model-free off-policy value-based RL algorithms. The model-free algorithm does not build a model of the environment to generate policy. A model-free algorithm estimates a value function or a policy from experience (the agent-environment interaction) without using neither the transition function nor the reward function. The model-free algorithms are suitable options for LLN since building the environment's dynamics is an expensive and unnecessary task. In off-policy learning, the agent can explore freely - its actions need not correspond to the current policy. In the DQN algorithm (Algorithm 10), the Deep Learning (DL) uses a Q-function $Q(s, a)$, also known as the action-value function, to approximate the value of taking a specific action (a_t) in the given state (s_t) to help RL in finding the optimum policy (π^*). Since there is no relation between sequence of states in 6LoWPAN (s_{t+1} is not the result of the action the agent has taken at s_t), the discount value (γ) is assigned as 0.001 in this chapter.

The Deep Q-Network (DQN) approximates the Q function. The DQN with probability ε selects a random a and with probability $1 - \varepsilon$ selects optimal Q-function (Q^*), (4.3). After executing the selected action a_t the agent observes next state s_{t+1} and reward r_t and stores (s, a, r, s_{t+1}) in the replay buffer D. Algorithm 10 shows how DQN functions.

Although there is a slight correlation between the incoming network traffic, the experiment replay strategy [156] is employed to guarantee the data are independent and identically distributed (IID) to avoid significant oscillations or divergence. The replay buffer D is a data structure including agent experiences $e_1, e_2, \ldots, e_n$ where $e_t = (s_t, a_t, r_t, s_{t+1})$.

$$\pi^*(a|s) \leftarrow arg\max_{a \in A} Q(s, a) \tag{4.3}$$

This chapter implements a lightweight Neural Network (NN) consisting of two hidden layers using the ReLU activation function, 100 nodes, and adam optimiser to approximate the Q-function. The agent's action space A comprises a set of actions $A = \{a_1, a_2, ..., a_n\}$, where the action a_t of the RL agent is the selection of a ML-IDS agent to classify s_t. If the selected action a_t (an ML-based IDS detector) makes a correct classification of the current state s_t (network observations), the reward is +1 and -1 otherwise. Since in this chapter, the states are not sequential (the packet that the agent receives at s_{t+1} is not the result of the action that the agent has taken at the previous time step s_t, the γ value assigned is near to zero (0.001).

To train the NN, the loss function needs to be determined. Since the goal of NN is to predict $Q(s, a)$, this chapter uses the squared difference between the actual action-value function and the prediction, (4.4) where θ represents the Q-function's parameter, i.e., the trainable weights of the network. The model aims to decrease the error and make current policy outcomes closer to the true Q-values. Therefore the model performs gradient (∇) descent over the loss function using (4.5) where $Q_{target} = (r + \gamma \max_{a'} Q(s', a'; \theta^-)$. The $(s, a, r, s') \sim U(\lambda)$ indicates that an experienced sample (s, a, r, s') is drawn uniformly at random from the replay buffer λ.

$$L(\theta) = \mathbb{E}_{\pi}[(r + \gamma \max_{a_{t+1}} Q(s_{t+1}, a_{t+1}; \theta) - Q(s, a; \theta))^2] \tag{4.4}$$

$$\nabla_{\theta_i} L_i(\theta_i) = \mathbb{E}_{(s,a,r,s' \sim U(\lambda))}[Q_{target} - Q(s, a; \theta_i))\nabla_{\theta_i} Q(s, a; \theta_i)] \tag{4.5}$$

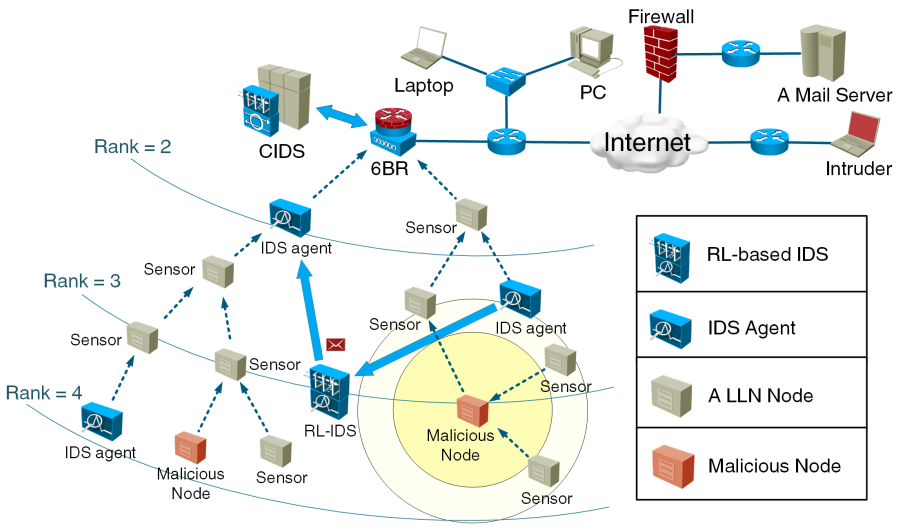


Figure 4.4: *System architecture.*

DDQN adds double learning to the DQN agent by using two Neural Networks (NNs). DDQN implementation and hyper-parameters are identical to DQN, and both use the off-policy Temporal Difference (TD) target [157]. However, DDQN employs two NNs, one for action prediction and another for action evaluation. Moreover, instead of MSE, DDQN uses

Algorithm 10: Deep Q-learning with experience replay

Initialisation
Initialise replay memory λ to capacity N
Initialise action-value function Q with random weights θ.
Initialise target action-value function $\hat{Q}$ with weights $\theta^{-} = \theta$.
T: final time step of an episode. τ: assigned to ten.
ϕ: refers to the preprocessing (scaling using normalisation) of the given state.
for *episode=1, M* **do**
    Initialise the environment, get s_1, and prepossess $\phi_1 = \phi(s_1)$
    for *t=1, T* **do**
        ***With** probability $1 - \varepsilon$*
            $a_t \leftarrow argmax_a Q(\phi(s_t), a; \theta)$
        Otherwise
            $a_t \leftarrow$ *select a random action*
        Execute action a_t in emulator and observe r_t and s_{t+1}
        Preprocess $\phi_{t+1} = \phi(s_{t+1})$
        Store transition $(\phi, a_t, r_t, \phi_{t+1})$ in λ
        Sample random mini-batch of transitions $(\phi_j, a_j, r_j, \phi_{j+1})$ from λ
        if *episode terminates s_{j+1}* **then**
            $y_j = r_j$
        else
            $y_j = r_j + \gamma max_{a'} \hat{Q}(\phi_{j+1}, a'; \theta^{-})$
        Perform a gradient descent step on $(y_j - Q(\phi_j, a_j; \theta))^2$ with respect to the network parameters θ
        Every τ steps reset $\hat{Q} = Q$

Huber loss for loss calculation. Huber loss tunes between MSE and Mean Absolute Error (MAE) using the parameter δ as threshold value [158].

We experiment with different epsilon (ε) values in this research; a higher ε value leads to exploration and taking less selected actions (detectors). This can help the model identify undiscovered ML classifiers that are precise in analysing particular types of network traffic and RPL attacks. Exploiting enhances the system performance by selecting actions (detectors) that have proven to be good at detecting particular types of attacks. Balancing exploration and exploitation by tuning the ε ($0 < \varepsilon < 1$) value is vital in designing an efficient system. The agent with probability epsilon (ε) explores and with probability ($1 - \varepsilon$) exploits. The best strategy is to initialise epsilon as a high value for more exploration and decay it over time to select greedy actions and accumulate more rewards. This study experiments with different exploration-exploitation, ε association strategies (softmax, linearly decaying ε value, etc.) and found that the exponentially decaying ε-greedy strategy [159] provides superior performance.

The computational complexity of Deep Q-Network (DQN) depends on various factors: the number of hidden layers, the number of neutrons per layer, etc. In DQN and Double DQN (DDQN), the environment has continuous state space, and computational complexity differs based on the algorithm strategy. In DQN using the experience replay method, the batch size defines the complexity [4].

Algorithm 11: RL-IDS Algorithm in action

Initialisation
S_{pkt}: *Collected packet*
C_{pkt} *{DIO, DAO, DIS, DAO-Ack, Application Packet}*
CIDS: Central IDS
RL_{alg}: *RL algorithm*
RL_{agent}: *RL agent*
IDS_{list}: *List of available IDSs*

An IDS *agent collects* S_{pkt} *from a LLN node*
if $S_{pkt} \in C_{pkt}$ **then**
    $D_1 \leftarrow IDS.analyse(S_{pkt})$
    if $D_1 == Abnormal$ **then**
        Transfer $S_{pkt} \rightarrow RL\_Agent$
        Regarding RL function-approximation algorithm (DQN or DDQN) compute
        $a \leftarrow argmaxQ^*(s_t, a)$ *(select IDS agent) given current state* $s_t(S_{pkt})$
        Take action 'a', (Transfer $S_{pkt} \rightarrow IDS_{list}[a_t]$*)*
        $D_2 \leftarrow IDS_{list}[a_t].analyse(S_{pkt})$
        Send $D_2 \rightarrow RL\_agent$
        if $D_2 == Abnormal$ **then**
            Transfer the alarmed packet (S_{pkt}) *to CIDS and notify Administrator*
            if *CIDS.analyse*(S_{pkt}) *== intrusion* **then**
                Send Reward (+1) $\rightarrow RL_{agent}$
                Notify Administrator
            else
                Send Penalty (-1) $\rightarrow RL_{agent}$
            RL_{agent} *receives feedback from CIDS and updates Q-function*

4.3.3 Data-set Preparation

Exploring Datasets In this chapter, the dataset is generated through simulations of several RPL scenarios with different number of malicious nodes. In each scenario, static and mobile nodes are randomly distributed over an LLN. The Tetcos Netsim simulator is used to simulate different RPL attack scenarios and generate raw datasets. The imbalanced dataset will be rectified during the pre-processing phase. The redundant, less informative records are removed from the dataset to make normal and malicious traffic equally distributed in the training dataset. Some ML algorithms (SVM, Logistic regression, etc.) are very sensitive about the scale of data [4] ; therefore, feature normalisation (Min-Max Scalar) and standardisation (Standard Scalar) techniques are adopted to scale features. This prevents IDS from being over-fitted to a particular type of traffic. The training dataset contains 48 features and 80,000 instances. The normal traffic constitutes 50% of the dataset, while each attack type equally has 5% of the dataset.

Data Preprocessing The data pre-processing reduces dataset complexity for ML algorithms; therefore, the ML algorithm can be trained over the pre-processed data faster and more efficiently than the raw data [160]. In this chapter, the data-processing constitutes data reduction, feature engineering, normalisation, and data sampling [160].

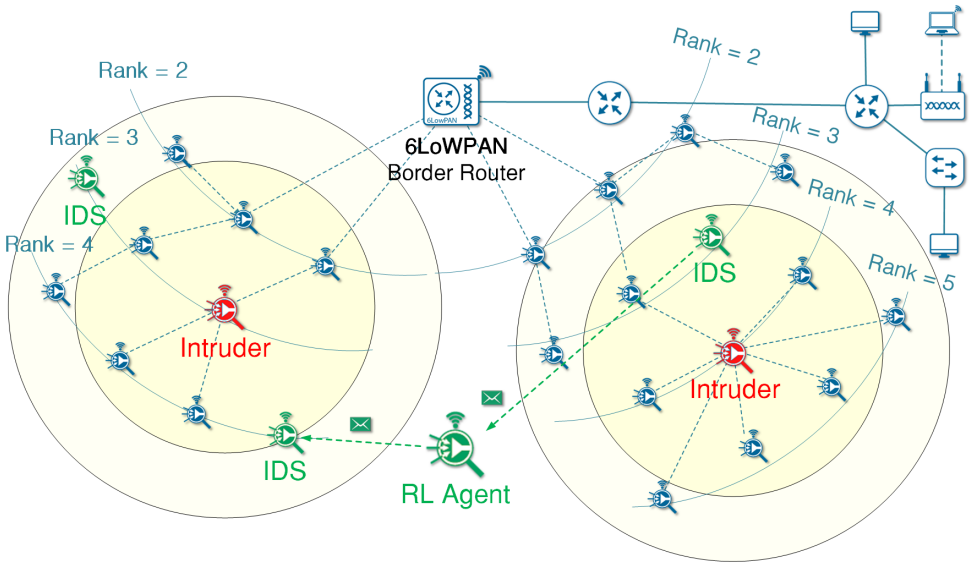


Figure 4.5: *Simulation environment.*

Data Generation This chapter uses Tetcos Netsim Simulator to simulate normal, and anomalous RPL traffics, Fig. 4.5. The Netsim is a well known paid license software known for accurate simulation of different network technologies, including 6LoWPAN. This chapter simulates several network scenarios (using the scenario generator feature of the simulator) for each type of RPL attack with different number of static and mobile nodes, from 8 to 128 nodes. Concerning the network's scales and the number of normal nodes, 10% to 30% of nodes associate as malicious nodes in scenarios. In Wormhole and DIS flooding attacks, half of the malicious nodes associated as external intruders. In all scenarios up to 10% of nodes are considered as IDS detectors in simulations. To generate a sufficient amount of malicious and normal traffic, based on the type of RPL attack each scenario is simulated for ∼21,600 seconds.

Feature Construction Feature construction, also referred to as feature engineering, emphasises that engineering salient features from the observed traffic leads to enhancement in classification. Every observed network packet contains different information about node configurations and identity. Training using the identity information of nodes leads to over-specialisation (over-fitting). Therefore such features should be excluded from training datasets. Constructing features based on nodes' geographical location [79], computational resource usage (CPU, RAM, ROM usages) [62], and power consumption [62; 56] can exhaust LLN nodes' resources [3]. Moreover, this significantly increases network overhead [84] on the LLN because nodes need to transfer such logged information to the IDS.

The header of RPL control packets (DIO, DAO, DIS, and DAO-Ack packets) contains information about node configurations, version number, advertised rank [147; 3]. Extracting information from these unicasted/multicasted control packets can help in constructing several features, described in Table 4.3. The engineered features play a vital role in improving the proposed IDS performance in detecting each RPL attack.

Table 4.3: *Engineered features*

Feature	Description
pkt_type	Type of packet (DIO, DAO, DIS, App etc)
pkt_status	Packet status
dio_count	No. of DIO advertised by sender
avg_hopcount	Average No. of hopcount (global perspective)
dis_count	No. of DIS unicasted/multicasted by sender
dao_count	No. of DAO unicasted by sender
daoack_count	No. of DAO-Ack unicasted by sender
neighbour_count	No. of neighbouring node
child_count	No. of children
avg_intpkt_time	Average delay between packets
rank_alteration_count	No. rank alteration
cmp_sender_parent_lq	Compare link quality of sender with its parent
snd_ctrl_count	No. control packet transferred by sender
cmp_lq	compare if sender has lower link quality than current node but advertise better rank
rcv_dao_count	No. of DAO received by current node
rcv_dio_count	No. of DIO received by current node
rcv_dis_count	No. of DIS received by current node
rcv_daoack_count	No. of DAO-Ack received by current node
trans_app_count	No. of application packet transferred by sender
pkt_e2e_delay	packet end-to-end delay
pkt_loss	Application packet loss ratio
cpkt_loss	Control packet loss ratio
src_rank	Sender rank in DODAG
adv_vn	advertised version number
rx_sens	Average receiver sensitivity
tx_power	Average transmission power
rssi	Received signal strength indicator of sender
same_parrent	sender has same parent as detector node
rcv_cpkt_count	No. of control packets received by sender node
prt_bst_lq	Current parent provide best link quality

4.4 Experimental Methodology

The proposed scheme employs both signature-based and anomaly-based ML algorithms to enhance the performance of IDS in detecting known and unknown intrusions. The proposed hybrid RL-IDS uses a passive decentralised monitoring technique [84] using a cluster-based placement [124] strategy to analyse 6LoWPAN traffics. The intended flow of the proposed scheme is shown in Fig. 4.4, the algorithm itself is described in Algorithm 11. We now evaluate the performance of the proposed method over 6LoWPANs with respect to different configurations and numbers of malicious nodes.

To evaluate the performance of the proposed scheme in detecting RPL attacks, four experiments (denoted as Exp1-Exp4) are conducted over different network configurations. In Exp1 we evaluate the performance of the proposed scheme using different homogeneous algorithms. Exp 2 evaluates the performance of the proposed RL-IDS using various heterogeneous ML detectors for detecting RPL attacks. Different scaled LLNs have been simulated with 10% $\sim$ 30% of malicious nodes. Exp 3 aims to evaluate the performance of RL-IDS using

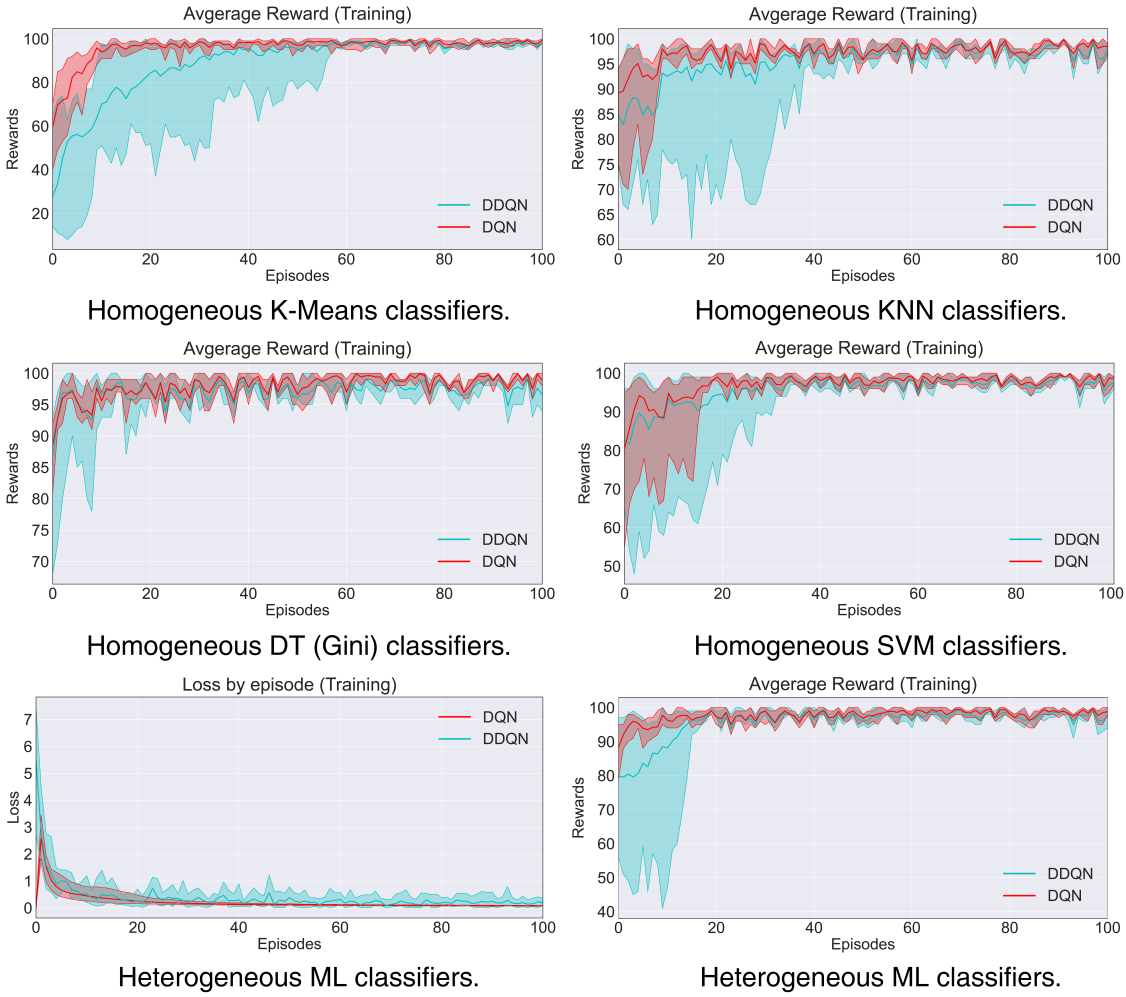


Figure 4.6: *Evaluation results of heterogeneous and homogeneous ML detectors.*

heterogenous detectors (hybrid detection strategy) in detecting unknown intrusions. Finally, in Exp 4, the performance of the proposed RL-IDS is evaluated against different types of RPL attacks using heterogeneous detectors, while 20% of nodes, including half of the malicious nodes, were mobile and in movement. All results are obtained from ten executions of each experiment.

This study evaluates the performance of RL-based IDS in terms of True Positive Rate (TPR), False Negative Rate (FNR), True Negative Rate (TNR), False Positive Rate (FPR), Accuracy (Acc), Precision (Pre), and F1 measure. The performance results are presented in Section 4.4.1. Here we use similar evaluation metrics as described in [3].

4.4.1 Experimental setup

RL-IDS with homogenous detectors: In the first experiment, we aim to evaluate homogenous ML algorithms' performance in detecting RPL attacks to discover the best combination of ML-detectors for hybrid heterogeneous RL-IDS. The parameters of each ML

Table 4.4: *Simulation parameters*

Parameters	Values
Number of nodes	16, 32, 64, 128
Number of Malicious nodes	$\sim 10\%, \sim 20\%, \sim 30\%$
Number of Workstations	4, 8
Transmission Range	50m
Number of ML detectors	$\sim 10\%$
Scenario Dimension (Terrain)	(250×250) to (850×850) s.meters
Traffic Rate	250 kbps
Simulation time	$\sim$ 21,600 seconds
Application Protocols	COAP, CBR
RPL mode	Storing and Non-storing
Mobility Modes	Random Walk, Group Walk
Path Loss Model	Log Distance, Exponent(n): 2
Distance between LLN Neighbors	$25 \sim 45m$
Objective Function (OF)	OF0, LQ
Receiver Sensitivity	-85 dBm

algorithm are configured to produce lightweight detectors with low complexity in the system. Each detector is assigned a batch of data to train over. The batch assigned to a detector is subject to chi-square feature selection to determine importance, and SelectKBest is then applied, with k=4, to determine the subset of features to be adopted by that classifier to train over. Since each training batch includes a different proportion of each RPL attacks and normal traffic, the chi-square nominates a different set of features for each ML detector. This chapter evaluates RL-based (DQN [158] and DDQN [157]) homogenous DT, KNN, K-means, SVM, and Logistic Regression (LR). The performances of different homogeneous ML algorithms using DQN and DDQN over ten runs are depicted in Fig. 4.6. In each run, we consider 10 IDS detectors present in the LLN. The performance of the proposed RL-IDS is the result of ten runs.

RL-IDS with heterogeneous detectors: Since each IDS detection strategy has unique strengths and weakness [3; 5], this chapter develops RL-based IDS with hybrid heterogenous ML detectors to incorporate the strengths of signature-based and anomaly-based IDSs. We develop combination of SVM, One-class SVM, DT, K-means, KNN, and LR to identify RPL attacks. The heterogeneous hybrid ML can provide optimum performance when we use an RL algorithm (DQN) for action-value selection, Fig. 4.6. To measure the performance of the proposed scheme against LLN's with different proportions of malicious nodes, we evaluate the performance of heterogeneous RL-based IDS against LLN's with different configurations, Table 4.4. Table 4.5 and Table 4.6 show the results of Exp 2.

Unknown Attack Detection: Table 4.7 indicates how our proposed IDS approach detects RPL attacks that were not present in the training dataset. We select each attack type in turn,

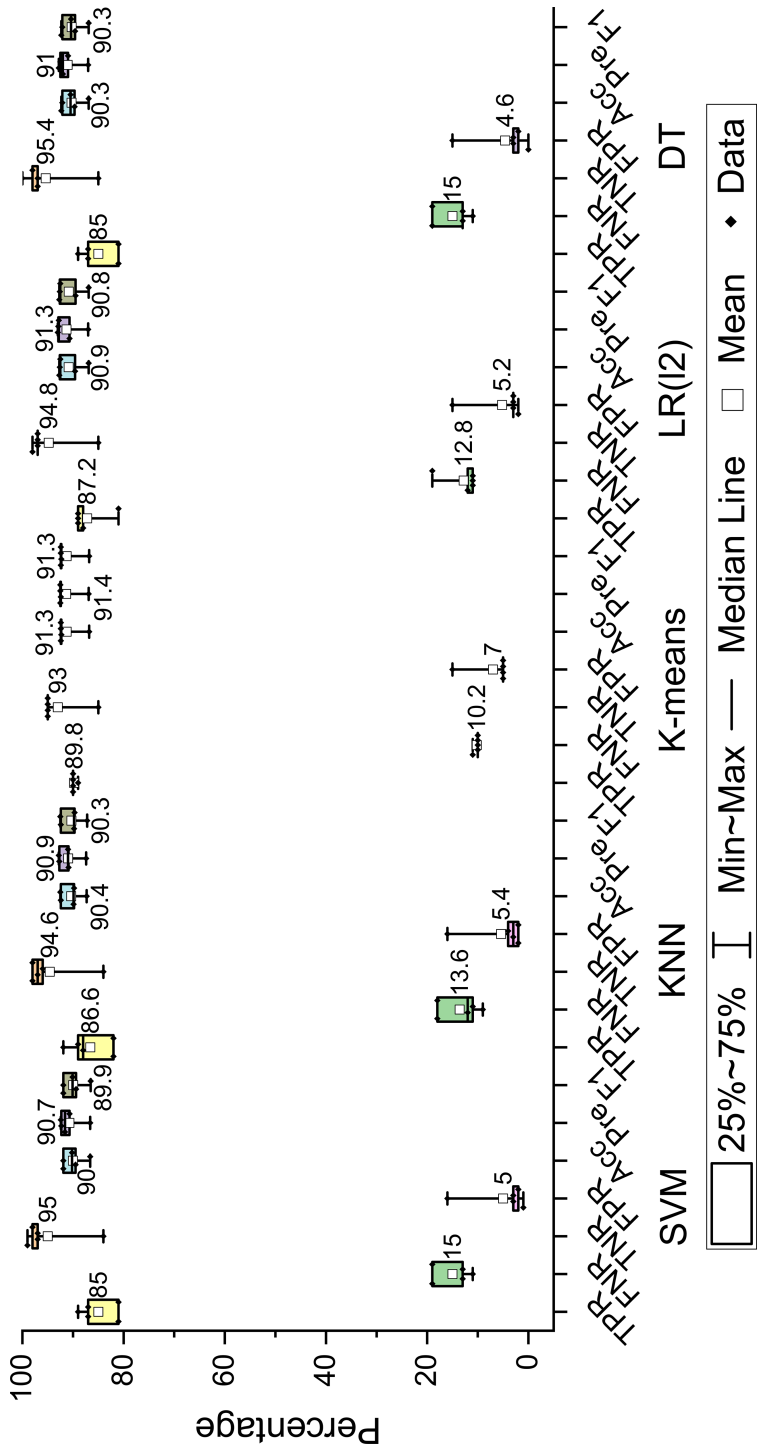


Figure 4.7: *Evaluation results of homogeneous ML detectors over all RPL attacks.*

Table 4.5: *Evaluation results, true positive rate and true negative rate*

N	M	TPR								TNR							
		SH	***BH***	***GH***	***DA***	***IR***	***WH***	***DS***	***R***	***SH***	***BH***	***GH***	***DA***	***IR***	***WH***	***DS***	***R***
16	10%	98.8	98.4	96.7	100	93.3	99.2	92.6	98.5	97.5	99.1	93.9	100	97.2	94.8	97.3	92.3
	20%	97.9	97.6	98.4	100	100	100	98.0	99.2	100	97.5	100	100	100	100	99.3	99.5
	30%	96.6	94.1	98.0	100	94.2	98.5	98.8	100	94.9	98.6	94.9	100	96.6	99.5	91.2	100
32	10%	91.4	98.8	93.0	100	98.5	100	98.0	94.1	97.7	97.9	98.6	100	99.6	100	97.3	99.2
	20%	98.8	98.8	98.0	100	99.2	100	100	100	99.3	99.3	99.3	100	99.5	100	100	100
	30%	97.8	98.4	99.4	100	97.9	91.8	100	98.6	98.3	99.6	95.2	100	96.7	96.4	100	98.3
64	10%	99.7	98.7	92.2	100	98.1	95.2	94.6	100	98.4	98.6	94.3	100	98.6	97.7	85.9	100
	20%	90.7	91.4	93.8	100	98.9	100	92.7	98.7	99.0	97.1	97.2	100	96.9	100	96.3	98.5
	30%	91.8	89.7	92.8	100	95.3	98.8	98.7	100	99.5	98.8	95.0	100	97.0	100	100	99.5
128	10%	98.8	95.5	95.9	100	98.4	100	93.7	94.8	99.4	98.6	100	100	99.5	97.6	99.2	100
	20%	100	91.5	98.1	100	96.7	98.4	92.2	98.5	100	96.7	97.8	100	86.4	92.3	99.3	97.9
	30%	99.0	95.6	98.7	100	98.5	100	94.8	99.0	99.3	98.0	100	100	98.0	98.6	100	99.5
SH: Sinkhole, **BH:** Blackhole; **GH:** Grayhole; **DA:** DIS Flooding; **IR:** Increase Rank; **WH:** Wormhole; **DS:** DIO Suppression; **R:** Replay; **N:** Total number of nodes; **M:** Number of Malicious nodes;																	

Table 4.6: *Evaluation results, Accuracy and F1*

N	M	Accuracy								F1							
		SH	***BH***	***GH***	***DA***	***IR***	***WH***	***DS***	***R***	***SH***	***BH***	***GH***	***DA***	***IR***	***WH***	***DS***	***R***
16	10%	98.2	98.8	95.4	100	95.2	97.2	95.0	95.6	98.1	98.8	95.4	100	95.2	97.2	94.9	95.5
	20%	99.0	97.6	99.2	100	100	100	98.6	99.4	98.9	97.6	99.2	100	100	100	98.6	99.4
	30%	95.8	96.4	96.5	100	95.4	99.0	95.2	100	95.7	96.4	96.5	100	95.4	99.0	95.1	100
32	10%	94.5	98.4	95.8	100	99.0	100	97.6	96.7	94.5	98.3	95.8	100	99.0	100	97.6	96.7
	20%	99.1	99.1	98.6	100	99.4	100	100	100	99.1	99.1	99.6	100	99.4	100	100	100
	30%	98.0	99.0	97.4	100	97.3	94.1	100	98.5	98.0	99.0	97.4	100	97.3	94.1	100	98.5
64	10%	99.1	98.6	93.2	100	98.3	96.5	90.1	100	99.1	98.6	93.2	100	98.3	96.5	90.1	100
	20%	95.2	94.2	95.5	100	98.0	100	94.6	98.6	95.2	94.2	95.5	100	97.9	100	94.6	98.6
	30%	96.0	94.2	93.9	100	96.2	99.4	99.3	99.8	95.9	94.2	93.9	100	96.2	99.4	99.3	99.8
128	10%	99.1	97.0	98.0	100	98.9	98.8	96.4	97.4	99.1	97.0	98.0	100	98.9	98.8	96.4	97.4
	20%	100	94.1	98.0	100	91.3	95.6	95.8	98.2	100	94.1	98.0	100	91.3	95.5	95.8	98.2
	30%	99.2	96.8	99.3	100	98.3	99.4	97.4	99.2	99.2	96.8	99.3	100	98.3	99.4	97.4	99.2
SH: Sinkhole, **BH:** Blackhole; **GH:** Grayhole; **DA:** DIS Flooding; **IR:** Increase Rank; **WH:** Wormhole; **DS:** DIO Suppression; **R:** Replay; **N:** Total number of nodes; **M:** Number of Malicious nodes;																	

train our system on the remaining 7 attack types, and then evaluate how well the trained system detects the omitted attack type (i.e. the evaluation set comprises only that attack type and normal). To the best of our knowledge, extant research does not address this issue [3].

LLN with mobile nodes: Only a few studies in the literature [3; 5] consider mobility among LLN nodes while mitigating some RPL attacks (SH, GH, DA, Sybil and Clone Id). To the best of our knowledge, there is no research that considers malicious mobile nodes on 6LoWPAN. In this chapter we take an initial step to shed light on the rationale underlying this prominent issue. In this regard, we measure the performance of the proposed RL-based IDS with heterogeneous detectors against different RPL attack scenarios (SH, BH, GH, DA, DS, IR, WH, and RA) with 20% of nodes, and half of the malicious nodes, being mobile. Fig. 4.8 shows the performance of the proposed scheme.

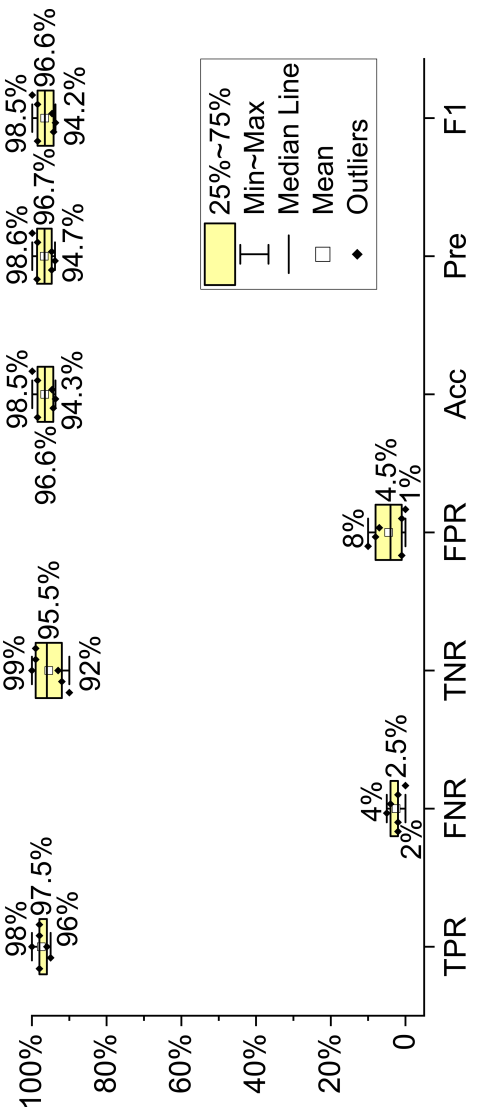


Figure 4.8: *Performance of hetrogenous RL-IDS in mobile scenarios.*

Table 4.7: *Unknown attack detection*

Unknown Attack	Performance Metrics					
	Acc	***Pre***	***TPR***	***FNR***	***TNR***	***FPR***
SH	87.3	87.7	92	8	82	18
BH	88.8	89	85	15	93	7
GH	95.8	95.9	98	2	94	6
IR	94.8	95.2	90	10	100	0
DA	100	100	100	0	100	0
WH	98.7	98.8	97	3	100	0
DS	94.9	95	92	8	97	3
RA	87.96	88.88	96	4	80	20

*__Acc__: Accuracy; __Pre__: Precision.

4.5 Analysing results

Both the DQN and DDQN converge to optimal policies in the proposed scheme; however, DQN converges faster than DDQN with lower bias and variance, as shown in Fig. 4.6. The proposed scheme provides an adaptive, robust intrusion detection solution (*DP1*) against RPL attacks. The adaptivity and robustness of the deep reinforcement learning not only helps the IDS to become flexible against various types of known intrusions but also makes them effective in detecting unknown intrusions, as shown in Table 4.7 (*DP5*). From the evaluation results (shown in Fig. 4.7, Table 4.5, and Table 4.6), we can argue that the proposed RL-IDS is effective against different RPL attacks for the networks with different configurations. Fig. 4.8 shows that heterogeneous RL-IDS is effective in detecting malicious nodes in mobile scenarios (*DP7*). Although all homogeneous detectors 4.4.1 converged to the optimal policy after 20 to 40 episodes, heterogeneous detectors using RL-based IDS converge faster with better performance in the detection of known and unknown intrusions. This is because heterogeneous detectors use a combination of signature-based and anomaly-based ML detectors to develop hybrid RL-IDS. Table 4.5 and Table 4.6 show that the proposed hybrid RL-IDS can provide an LLN with security against different internal (SH, BH, GH, IR, DA, WH, DS, and RA) and external (DA and WH) intrusions (*DP4*). Nevertheless, to ensure low overhead over LLNs (*DP3*) the proposed scheme uses the passive decentralised monitoring with its RL-based IDS.

4.6 Summary

In this chapter, we have presented a new RL-based IDS that employs hybrid heterogenous lightweight ML detectors to passively monitor 6LoWPAN traffic. Our approach has exhibited comprehensive feature engineering and has been shown to detect a much greater range of RPL attacks than extant research, including several previously unaddressed attacks. The work also addresses for the first time combinations of attacks. Also, as far as we are aware, evaluation against previously unseen RPL attacks has never been demonstrated in the literature. It

remains to be seen how the proposed scheme can perform in a larger scaled 6LoWPAN (with more than thousand LLN nodes). In such networks, we may require more than one RL-IDS (multi-agent RL-based IDSs) to identify RPL attacks. Furthermore, the ML-based IDS agents in the proposed scheme were batch-trained and incapable of adapting to new types of intrusion.

Chapter 5

Adversarial RL-based IDS for 6LoWPAN

In Chapters 3 and 4, we proposed batch-trained anomaly-based and signature-based "IDS agents", respectively, to detect intrusions in 6LoWPAN; these developed IDS agents could not adopt new data distribution or unknown intrusions. This chapter aims to develop a framework to generate adaptive and resource-efficient IDS agents using adversarial reinforcement learning, concept-drift detection and incremental machine learning algorithms. Moreover, in this chapter, we also evaluate the performance of our proposed scheme against black-box and grey-box ML-based RPL attacks. In Chapter 2 (Section 2.8.3), we discussed the vulnerability of 6LoPWAN against ML-based intrusions.

5.1 Introduction

In 6LoWPAN, the RPL forms a destination-oriented directed acyclic graph (DODAG) to connect the nodes to the 6LoWPAN Border Router (6BR). Each node in a DODAG has a rank that indicates its position relative to other nodes and with respect to the 6BR. In RPL, the Objective Function (OF) [32] defines how nodes should calculate their ranks through constructing a tree-based routing graph, known as a Directed Acyclic Graph (DAG). The node's rank expresses its distance from the 6BR; the closer nodes are to the 6BR (known as root). Although the RPL was initially developed as a routing protocol for stationary LLNs, a broader usage of LLN nodes motivated researchers to develop mobility for RPL [21]. Only 13% of research has considered mobility in detecting RPL attacks [3]. It is vital to develop an IDS to identify RPL attacks in the evolving environments. Because of evolving nature of the LLN environment (e.g. node mobility and application variation), the distribution of traffic is non-stationary. Furthermore, unpredictable and abnormal incidents cause the network environment to evolve and induce shifts in the statistical distributions of data known as concept drift. The concept drift occurrences can degrade the performance of IDSs in LLNs and prevent them from accomplishing their primary tasks.

In an imbalanced classification problem, the distribution of instances over the known classes is not equal. The class with abundant instances is called the majority class, whereas the class with a lower proportion is termed the minority class. In computer networks, the imbalance is a property of the problem domain, where the natural occurrence of one class

dominates other classes. This is because the process that draws observations in the minority class has less frequency. Imbalanced training data results in the development of biased models, most typically models which exhibit poorer performance on the minority class. Since the minority class is often more critical, misclassifications of this class are particularly problematic.

Existing supervised and unsupervised IDS for 6LoWPAN in the literature are batch-trained (offline-based) and are tuned to the environment of the specific, fixed (stationary) training data [3]. Hence, they are not robust in detecting/learning shifts in the network, such as the breakout of unforeseen intrusions or concept drifts. When implementing learning algorithms, one often faces the difficult problem of dealing with non-stationary environments whose dynamics evolve due to some unknown or not directly perceivable cause. In the evolving environment of 6LoWPAN, IDSs must analyse extensive, noisy, and imbalanced data. Hence, the proposed adversarial reinforcement-learning-based IDS (ARL-IDS) implies over-sampling and under-sampling on the imbalanced data environment of the LLN to develop a robust and generalised IDS. A non-stationary dynamic environment drives reinforcement learning (RL) methods to relearn the policy from scratch continuously. Therefore this chapter we employ Concept-drift Detection (CD) algorithms to generate accurate defenders upon different concept drifts in the 6LoWPAN. Learning under concept drift is associated with sequential decision making, where training and testing observations are conducted in temporal sequence (timestamps).

The streaming data in LLN require a dynamic security approach capable of updating the detection model on the fly. Moreover, since the IDS cannot accommodate the sheer amount of streaming data, it needs to train on succinct data explicitly. This chapter presents the *first* application of adversarial reinforcement learning, concept-drift detection, and incremental ML algorithm (KNNADWIN [161]) for development of robust and generalised IDS, and proposes a novel approach to incorporate the salient information of imbalanced attack profiles into the resource-constrained intrusion detectors. The developed ML-IDS can make reasonable trade-offs between variance and bias for detecting RPL attacks in evolving and imbalanced data environments of 6LoWPAN. Since the 6LoWPAN is subjected to various known and unknown routing intrusions and has an evolving data environment, adaptivity and concept-drift detection play vital roles in developing robust IDS.

5.1.1 Organisation

The rest of this chapter is organised as follows. Section 5.2 provides an overview of state-of-the-art related literature. Section 5.3 presents our proposed scheme. Section 5.4 provides the foundation for adversarial ML-based attacks. Section 5.5 describes our implementation and gives evaluation details and provides a concise analysis of the proposed scheme. Finally, Section 5.6 concludes the chapter.

5.2 Related Works and Motivations

More than a hundred articles have been published proposing IDS for 6LoWPAN in the literature[3]. 54% and 58% of the proposed IDSs are specification-based and utilise the active monitoring [3] (where legitimate nodes are required to participate in intrusion detection tasks

by collecting and aggregating data); while only 21% of the literature have considered a hybrid detection strategy[3]. There has been an ongoing interest to develop ML-based IDS (ML-IDS) for 6LoWPAN [62; 81; 56; 54] (*DP5*). Existing ML-IDSs perform offline training to develop an IDS model for the 6LoWPAN; hence, the lack of adaptivity degrades their performance against concept drifts [121; 133; 113; 135]. To the best of our knowledge, existing articles in the literature assume that the 6LoWPAN data distribution remains stationary over time and develop their IDS model with regards to a prespecified LLN configuration [3; 49; 94; 90] (*DP1, 2, and 6*).

Researchers in [132; 49] propose a specification-based IDS using active monitoring (active monitoring is explained in [3; 84]), where legitimate nodes have to report their observations to the IDS. Likewise, the proposed system in [90] utilises active monitoring in developing an anomaly-based IDS. The shortcomings of active monitoring (e.g. the computational and network overhead that they cause on LLNs) [3] encouraged the authors of [94] to utilise a passive monitoring approach (passive monitoring is explained in [3; 84]) in developing a signature-based IDS. To enhance the performance of IDS in 6LoWPAN, and to incorporate the strength of different detection strategies, [81; 62; 54] propose hybrid ML-IDS (*DP5*). Although the proposed ML-IDS's may identify RPL attacks in the 6LoWPAN with a stationary data environment, the detection performance of offline ML-IDS degrades dramatically under concept drifts. To the best of our knowledge, there is no concept-drift based incremental IDS for the 6LoWPAN in the literature (*DP6*). Although ML algorithms can enhance the performance of an IDS in identifying routing intrusions, existing ML-IDS in the literature address stationary network environments and are incapable of adjusting to shifts in network configurations. Nevertheless, researchers in [135; 113; 134; 133] utilise concept-drift detection and incremental ML algorithms (e.g. Hoeffding Tree, and RF) to develop IDS for different network technologies (using the KDD dataset).

On the other hand, the adoption of RL can make the IDS robust. [162; 153; 149; 150; 148] utilise RL approaches to develop IDS for different network technologies. With the evolving nature of the network environment, the network observations are imbalanced toward the majority class (legitimate activities). An imbalanced training dataset causes a negative impact on the performance of IDS over minority classes (malicious activities) (*DP1*). Therefore, [163; 152; 151] utilise an adversarial RL approach to train IDS over more important instances of minority and majority classes. Researchers in [62] employ the Synthetic Minority Oversampling Technique (SMOTE) algorithm to generate balanced offline training datasets and develop balanced ML-IDS for a stationary 6LoWPAN environment. However, to the best of our knowledge, there is no article in the literature that utilises RL or adversarial RL approaches for IDS in the 6LoWPAN. Regarding the scarcity of training data, existing articles are mostly restricted to a limited set of stationary network scenarios [3], which may induce an over-fitted model. Despite the wealth of literature available in the field, there is a lack of generalised and robust IDS for 6LoWPAN (*DP1*). The desirable IDS should accommodate succinct data and be generalised across different scaled 6LoWPAN and routing intrusions (*DP3*). Table 5.1 compares related works to our approach with respect to the method applied, attacks considered, and desirable properties (as discussed in Section 5.2.1).

Table 5.1: *Related works*

	Scheme	Method	Attacks Considered and Datasets	Desirable Properties						
				DP1	*DP2*	*DP3*	*DP4*	*DP5*	*DP6*	*DP7*
IDS*	[132]	Specification-based IDS	WH and Sybil	×	×	✓	×	✓*	×	×
	[90]	Active decentralised anomaly-based IDS	DA and NA	×	×	×	×	✓*	×	×
	[94]	Passive decentralised signature-based IDS	DA	×	×	✓	×	×	×	×
	[49]	Active decentralised specification-based	WP, DA, and SH	×	×	×	✓	✓*	×	×
ML-IDS*	[81]	Hybrid ML-IDS (k-means and decision tree)	WH	×	×	×	×	✓*	×	×
	[62]	Ensemble Voting (MLP and RF)	SA, VN, SH, and BH	×	×	×	✓	✓*	×	×
	[56]	Hybrid ML-IDS using passive monitoring	SH, WH, and DA	×	×	×	×	✓*	×	×
	[54]	Unsupervised Optimum-Path Forest Clustering	SH, WH, and SF	×	×	×	×	×	×	×
	[164]	Transfer-learning-based centralised IDS	WP, DA, SH, and VN	✓	×	×	✓	✓	×	×
	[165]	Reinforcement-Learning-Based IDS	SH, BH, GH, IR, DA, WH, DS	✓	×	×	✓	✓	×	✓
CD-IDS	[133]	Online adaptive RF + Concept-drift detection	KDDCup99	✓	✓	✓	D/N	×	×	D/N
	[134]	Online RF (Hoeffding Trees)	KDDCup99	✓	✓	✓	D/N	×	×	D/N
	[113]	Ensemble Weighted Voting, RF	KDDCup99	✓	×	✓	D/N	×	×	D/N
	[135]	CD-based ensemble incremental ML-IDS	KDDCup99	✓	✓	✓	D/N	×	×	D/N
RL-IDS	[148]	Q-learning, centralised hybrid IDS for WSN	KDDCup99	✓	×	×	D/N	✓*	×	D/N
	[150]	Anomaly-based IDS using RL	NSL-KDD and AWID	✓	×	×	D/N	✓*	×	D/N
	[149]	Centralised anomaly-based IDS using Deep RL	NSL-KDD and UNSW-NB15	✓	×	×	D/N	✓*	×	D/N
	[153]	Q-learning based IDS	NSL-KDD	✓	×	×	D/N	✓*	×	D/N
	[162]	Non-stationary multi-armed bandit RL	Smart Home Cyber-Physical	✓	✓	×	D/N	✓*	×	D/N
ARL-IDS	[151]	Adversarial DRL to enhance IDS performance	NSL-KDD and AWID	✓	✓	✓	D/N	✓*	×	D/N
	[152]	Distributed DRL for IDS	NSL-KDD, UNSW-NB15 and AWID	✓	×	×	D/N	✓*	×	D/N
	[163]	DQN and SMOTE using adversarial DRL	NSL-KDD	✓	✓	✓	D/N	✓*	×	D/N
	Proposed Scheme	Adversarial DRL, CD, and Incremental ML	SH, BH, GH, DA, DS, IR, WH, WP, and RL-based Adversary	✓	✓	✓	✓	✓	✓	✓

D/N: Different Network-technology. In the "Attack" column, the later entries refer to available datasets that contain a variety of attacks, (but these exclude RPL attacks); ✓: Satisfy; ×: Not addressed; **SH:** Sinkhole, **BH:** Blackhole; **GH:** Grayhole; **DA:** DIS Flooding; **IR:** Increase Rank; **WH:** Wormhole; **DS:** DIO Suppression; **WP:** Worst Parent; * They proposed IDS for 6LoWPAN; ✓*: It may satisfy but has not justified it.

5.2.1 Contribution

Various IDSs have been proposed in the literature to identify existing routing attacks (e.g. sinkhole, blackhole, and wormhole) against 6LoWPAN. However, none of the existing schemes

in the literature satisfies all the desirable properties that are mentioned in Section 1.2.1. The *major* contributions of this chapter can be summarised as follows:

- The proposed scheme is *novel* in developing adversarial RL, concept-drift detection, and incremental-ML approaches for generating efficient, robust, and generalised IDS.
- Our work is the *first* to use the exploratory data analysis (EDA) approach to characterise each RPL attack and engineer a set of features to aid their identification in 6LoWPAN.
- A *new* sophisticated adversarial ML-based attack called *RL-based combinational intrusion* has been introduced for 6LoWPAN, where the adversary uses RL approaches and desires to maximise the negative impacts upon the target network while avoiding its exposure. These intrusions are difficult to identify by existing IDSs in the literature.
- Our evaluations are carried out using a more extensive range of routing attacks than we have encountered in the literature, including less researched ones (e.g. Increase Rank, DIO suppression, and Worst Parent attacks).

5.3 Proposed Scheme

In this section, we propose our adversarial reinforcement learning scheme for developing efficient incremental IDS that is stable and robust against different intrusions in various scaled networks.

The proposed scheme consists of three phases: initialisation (Algorithm 12), drift detection, and concept drift adaptation. In the initialisation phase, instead of sampling all the intrusions in various scaled networks, the proposed adversarial environment jointly trains a second agent (called the adversarial agent) to disrupt the original agent (called the defender agent) by selecting the most sophisticated intrusions. The adversary obtains a reward only when the defender fails to classify the intrusions, where reward is a real number $R_t \in \mathbb{R}$. Accordingly, the adversary learns to sample the difficult instances (malicious activities that may cause the defender agent to fail) over time. On the other hand, the defender agent applies an incremental concept-drift-based classifier to develop a detector robust to any intrusion generated by the adversary. The development of the defender agent classifier in the proposed adversarial environment enhances its stability and robustness to different training/testing conditions. Next, the drift detection phase enables the proposed scheme to detect concept drifts (e.g. gradual, sudden, and recurring [121]) in the streaming data environment. If a fluctuation of the defender error rate is proven to be significant statistically, a drift warning will be raised. In the concept drift adaptation phase, the proposed scheme re-initialises the developed detectors through the drift adaption and adversarial environment. This makes the proposed scheme dynamic and robust against an adversary that changes its strategies over time, as depicted in Fig. 5.3. Without windowing and buffering, analysis of streaming data in real-time is impractical on resource-constrained devices [162]. Furthermore, because of volume of streaming data, upstream network and width saturation and throttling, it proves impractical to offload the processing of observations. Fig. 5.1 depicts the proposed system architecture. Our proposed scheme employs a passive decentralised monitoring approach using a cluster-based placement strategy to analyse the data stream in 6LoWPAN.

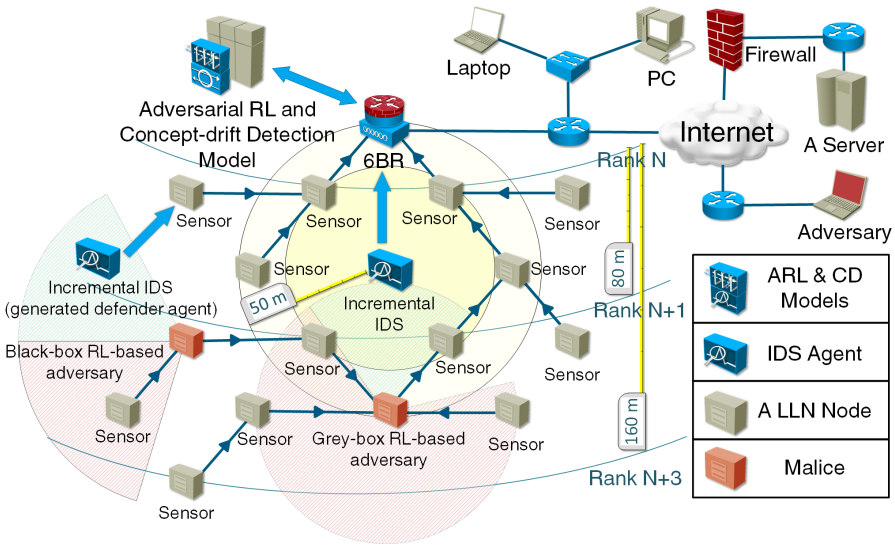


Figure 5.1: *System architecture.*

Initialisation: The major bottleneck in developing a high-capacity detector is its dependency on large amounts of balanced training data. Consequently, accumulating and accommodating such large amounts of data is time-intensive and could be infeasible for an LLN node. As a result, instead of sampling all types of intrusions in a variety of networks, the proposed adversarial environment jointly trains the adversarial agent to disrupt the defender agent by selecting the most sophisticated intrusions. In order to eliminate the imbalances and ensure the generalisation of the defender agent, each observation frequency is adjusted based on the adversarial-agent action-value function $Q_{adv}(s, a)$. To develop an efficient IDS, the proposed adversarial RL model performs appropriate over-sampling and under-sampling of the categories within the imbalanced data environment. The proposed adversarial environment generates network data streams (states) and rewards a defender agent that aims to identify the true nature of the current state. The adversarial environment is also leveraged to provide the most intriguing instances by adjusting the RL exploration-exploitation strategy. The utilised sampling approach circumvents the unbalanced learning issue for the minority class intrusions.

Now, in order to implement the proposed adversarial environment, we make some required changes in the RL environment. The modified RL environment has stochastic non-stationary state space. Each state $s_i \in S$ is a feature vector set of network observations $X = \{< x_1, x_2...x_n >\}$ (shown in Table 5.3). Considering $S = \{s_t, s_{t+1}, ..., s_\infty\}$ as the states corresponding to the network communication between nodes (containing features that are engineered in section 5.5.1), where s_t and s_{t+1} are not correlated. That means both the s_t, and s_{t+1} may be generated from two different data distributions i.e., $D_t \neq D_{t+1}$ where D denotes an underlying distribution. In the proposed adversarial environment, the responsibility of the defender agent is to classify the states generated by the adversarial agent. The adversarial environment generates positive or negative rewards based on the defender agent's correct or incorrect action a_{def} (classification) given the current state s_t.

The adversarial agent dynamically explores and exploits different available actions in its

Algorithm 12: Initialisation (Adversarial RL)

Initialisation
Initialise two replay memories λ_{adv} *and* λ_{def} *to capacity* N
Initialise action-value function $Q_{adv}(s, a_{adv})$ *and* $Q_{def}(s, a_{def})$ *with random weights* θ_{adv} *and* θ_{def}
Initialise target action-value functions $\hat{Q}_{adv}$ *with weights* $\theta^{-}_{adv} = \theta_{adv}$ *and* $\hat{Q}_{def}$ *with weights* $\theta^{-}_{def} = \theta_{def}$*;*
T: final time step of an episode; τ*: assigned to 10; M: assigned to 100. s: is a feature vector set* $X = \{< x_1, x_2 ... x_n >\}$ ϕ*: refers to the preprocessing (scaling using normalisation) of the given state.*

for *episode=1, M* **do**
  Preprocess $\phi_1 = \phi(s_1)$
  for *t=1, T* **do**
    With *probability* $1 - \varepsilon_{adv}$
      $a_{adv_t} \leftarrow arg\max_{a_{adv}} Q_{adv}(\phi(s_t), a_{adv}; \theta_{adv})$
    Otherwise
      $a_{adv_t} \leftarrow$ *select a random action from* A_{adv_t}
    With *probability* $1 - \varepsilon_{def}$
      $a_{def_t} \leftarrow arg\max_{a_{def}} Q_{def}(\phi(s_t), a_{def}; \theta_{def})$
    Otherwise
      $a_{def_t} \leftarrow$ *select a random action from* A_{def_t}
    Take a_{def_t} *and observe* r_{adv_t}, r_{def_t}
    $a_{adv_{t+1}} \leftarrow Q^{\pi}_{adv}(\phi(s_t), a_{adv_t})$
    Random sample s_{t+1}*, where* $s_{t+1} \in S(y = a_{adv_{t+1}})$
    Prepossess $\phi_{t+1} = \phi(s_{t+1})$
    Store transition $(\phi, a_{adv_t}, r_{adv_t}, \phi_{t+1})$ *in* λ_{adv}*, and* $(\phi, a_{def_t}, r_{def_t}, \phi_{t+1})$ *in* λ_{def}
      Sample random mini-batch of transitions $(\phi_j, a_{adv_j}, r_j, \phi_{j+1})$ *from* λ_{adv}*, and* $(\phi_j, a_{def_j}, r_j, \phi_{j+1})$ *from* λ_{def};
      if *episode terminates* s_{j+1} **then**
        $\psi_{adv_j} = r_{adv_j}$
        $\psi_{def_j} = r_{def_j}$
      else
        $\psi_{adv_j} = r_{adv_j} + \gamma \max_{a'_{adv}} \hat{Q}_{adv}(\phi_{j+1}, a'_{adv_j}; \theta^{-}_{adv})$
        $\psi_{def_j} = r_{def_j} + \gamma \max_{a'_{def}} \hat{Q}_{def}(\phi_{j+1}, a'_{def_j}; \theta^{-}_{def})$
      Perform gradient descent steps on $(\psi_{adv_j} - Q_{adv}(\phi_j, a_{adv_j}; \theta_{adv}))^2$ *with respect to the network parameters* θ_{adv}*, and* $(\psi_{def_j} - Q_{def}(\phi_j, a_{def_j}; \theta_{def}))^2$ *with respect to the network parameters* θ_{def}
    Every τ *steps reset* $\hat{Q} = Q$

action space $a_{adv} = \{a_0, ..., a_9\} \in A$ to diminish the rewards R of the defender agent by initiating the intrusions that the defender agent can not identify. When the defender agent incorrectly identifies the current state s_t, the adversary agent receives a positive reward. The

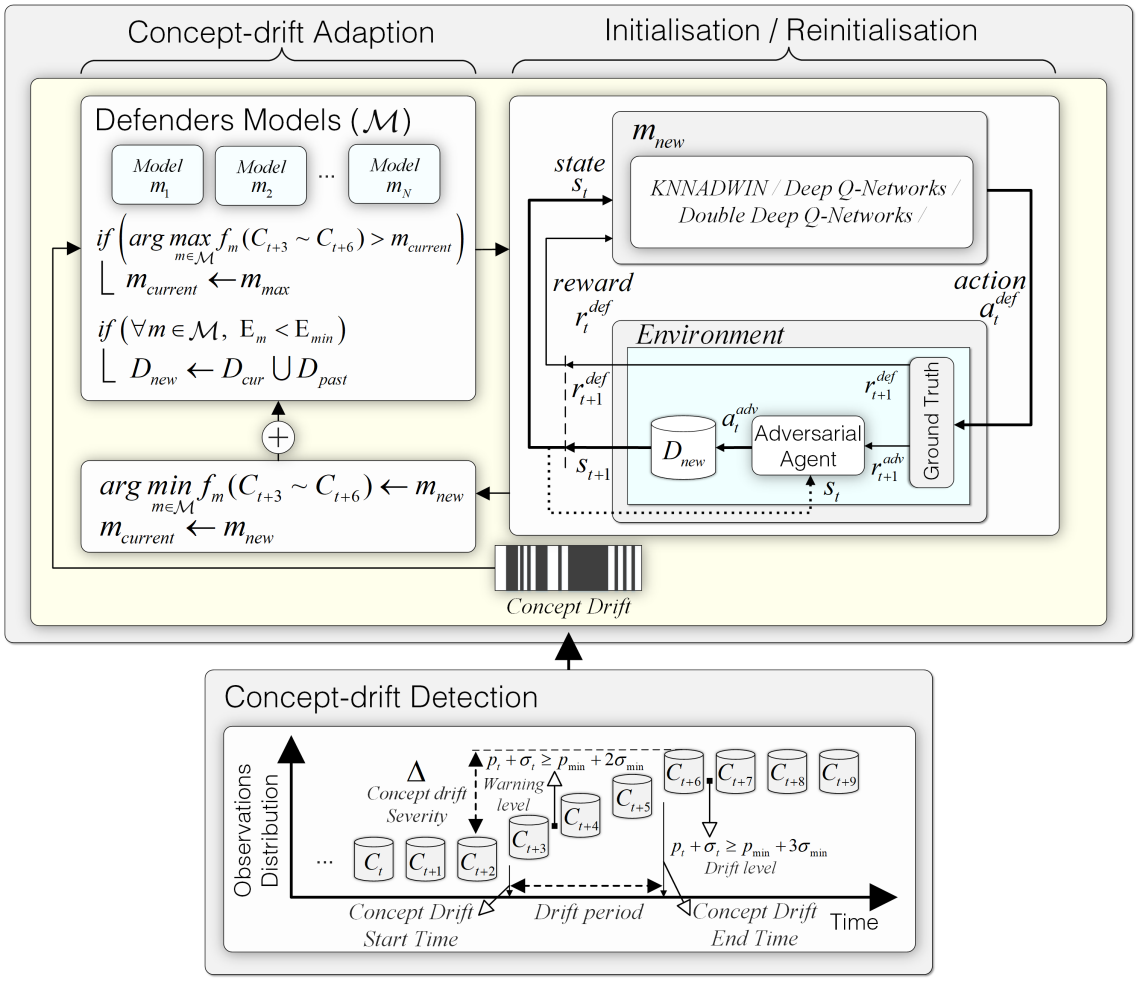


Figure 5.2: *Adversarial RL-based IDS.*

action executed by one agent affects the goals and objectives of the another and vice-versa. Consequently, the incorrect predictions and negative rewards force the defender agent to learn the most sophisticated observations and adapt its mistakes to accumulate more rewards in the long-run. By obtaining rewards, the adversarial agent tries to obtain a policy (π) that maximises negative impacts to the 6LoWPAN through initiating intrusions that the defender agent fails to identify. In general, both the adversarial and defender agents intend to maximise their total amount of discounted rewards following time t ($G_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1}$) [123] where a reward received k time steps in the future is worth only γ^{k-1} times what it would be worth if it were received immediately.

The primary component of Q-learning algorithms is a method for efficiently and properly estimating the Q-value. In Q-learning, the Q-value is updated by the equation 5.1.

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha[R_{t+1} + \gamma \max_A Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)] \tag{5.1}$$

Where (s_t), (s_{t+1}), (a_t), (a_{t+1}), (r_{t+1}) denote the current state, next state, current action, next action, and next reward-value, respectively. On the other hand, the parameters (α), (γ)

denote the learning rate and the discount factor, respectively. The $Q(s_t, a_t)$ is the motivation of the action a_t in state s_t, which means the agent will choose the action with the max Q. In equation 5.1, α and γ are the attenuation parameters, within domain $(0, 1]$. Since the Deep Q-Network (DQN) [166] employs the experiment replay approach for sampling, the drawn states by the adversary agent are not in sequential order [151; 163]; hence, in the proposed scheme, the γ is set close to zero. Consequently, the algorithm is not required to remember prior states. The α and γ are initialised with values 1.0 and 0.001, respectively.

The Q-leaning (off-policy temporal difference) is a tabular-based algorithm that represent the expected state-value function $V(s)$ or the $Q(s, a)$ in a lookup table; however, in the non-stationary environment, when the number of states increases, the table's size increases exponentially, making Q-leaning impractical. Hence, in this chapter we employ the DQN [166] and Double Deep Q-Networks (DDQN) [157], model-free, and off-policy algorithms that approximate the $Q(s, a)$ using NN. Since the DQN and DDQN are Off-policy algorithms, they are independent of the policy being followed to act in the environment. Deep Q-Learning replaces the regular Q-table with a neural network (NN). The agent that employs a NN to represent the Q-function is referred as a Q-network, which is denoted as $Q(s, a, \theta)$. The parameter θ denotes the weights of the NN, and the Q-network, trained by updating θ at each iteration to approximate the real Q-values. Deep Q-network that utilises Deep NN (DNN) is referred as DQN, and it is proven to be more favourable with better performance and more robust learning [166]. In general, DQN utilises two primary techniques for stabilising the Q-learning:

- **Target-Q-Network**, that is a stationary network to generate the target Q-values, used to calculate the loss for each action during the training procedure. Furthermore, the target network will be synchronised with the primary Q-network at every τ steps by copying directly. The fixed target-Q-network ($\hat{Q}$) uses the previous weights of NN i.e., θ^- for the Q-learning target ψ_i, shown in equation 5.2.

$$\psi_i = r + \gamma \max_{a_{t+1}} \hat{Q}(s_{t+1}, a_{t+1}, \theta^-) \tag{5.2}$$

- **Experience Replay**, that save its interactions experience tuple $e_t = (s_t, a_t, r_t, s_{t+1})$ at time t into a replay-memory $\lambda_t = \{e_1, ..., e_t\}$, and randomly draws a batch of instances from the experience pool to train the deep convolutional network's parameters rather than directly applying the online instances as in Q-learning. An experience replay-memory λ stores the k most recent experiences an agent has gathered. (In this chapter the k is assigned to 100.) If λ is full, the earliest experience is discarded to make space for the latest one (first in, first out buffer).

As discussed, our goal is to find the best parameter θ in the deep neural network such that $Q(s, a, \theta) \approx Q_\pi(s, a)$. In DQN, optimising θ can be done sequentially by minimising a sequence of loss functions $Loss_i(\theta_i)$ that is optimized at each iteration and represented in equation 5.3.

The DQN applied in the proposed scheme helps to gather experiences used in approximating the Q-function, as follows. The DQN uses the current state s_t, which corresponds to the features extracted from the network observations, as an input. The output of the DQN denotes the Q-function $Q(s, a)$ for $a \in \mathcal{A}$. The $Q(s_t, a_t)$ is the agent's motivation and

corresponds to how good it is if the agent take some action a_t in the given state s_t. The DQN model applies Equation 5.3 to update the Q-function. In DQN, the evaluation network updates quickly and is used to estimate the value of $\hat{Q}$; however, the $\hat{Q}$ updates slowly and is employed to approximate the Q value.

$$\mathcal{L}(\theta) = \mathbb{E}_{\pi}[(r + \gamma \max_{a_{t+1}} \hat{Q}(s_{t+1}, a_{t+1}; \theta^{-}) - Q(s, a; \theta))^2] \tag{5.3}$$

Considering the imbalanced evolving environment of 6LoWPAN, normal traffic is highly over-represented and is referred to as the majority class. However, the proposed adversarial RL is cautious to avoid developing an imbalanced IDS model by performing a dynamic and intelligent data re-sampling method. The defender agent predicts the attack type for the given state (s_t) by taking action a_t. On the other hand the adversary agent selects the target type (y_t), which will be used to draw the next state, where $y \in \{\text{RPL attacks} \cup \text{Normal activities}\}$. Now, due to the low frequency of appearance or the difficulty in predicting some states, the proposed adversarial environment more often presents some instances where the defender agent may fail to classify. In order to address this issue, the adversary agent needs to emphasise such deceptive observations to enable the formation of a generalised defender agent. All positive rewards R^+ for the defender agent will be considered as penalties R^- for the adversarial agent. Hence, this chapter employs adversarial DQN which is responsible for optimising the adversary agent inside the environment. The Q-function responsible for optimising the environment has a set of actions ($A \in [0-9]$) corresponding to each possible RPL attack in the data-set. On the other hand, the defender agent gradually develops an incremental model that can reduce negative impacts of the adversarial agent.

Balancing exploration and exploitation by tuning the ε ($0 < \varepsilon < 1$) value is vital. The agent explores with probability epsilon (ε) and exploits with probability ($1 - \varepsilon$). The best strategy is to initialise ε as a high value for more exploration and decay it over time to accumulate more rewards (have experimented in section 5.5.2). This study experiments with different exploration-exploitation, ε association strategies (softmax, linearly decaying ε value, etc.) and found that the exponentially decaying ε-greedy strategy [159] provides good performance.

We consider each episode as a training round throughout the entire data-set; The adversary agent chooses its action considering its policy $\pi_{adv}(a_t|s_t)$, being the lower bound of epsilon different for each case. The lower bound of the adversary ε will be set as 0.5. This chapter empirically finds that a higher ε-value for adversary agent can enhance the defender agent generalisation and robustness. Initially, the adversarial agent takes stochastic actions. However, the adversarial agent improves and takes actions (initiates intrusions) to maximise its reward, that is fluctuating over time. Fig. 5.9 depicts that the distribution of the initiated attacks has a stochastic trend, but it also has the tendency to exaggerate the importance of several intrusions.

Through the development of defender agent in the proposed adversarial environment, we consider the following sequences:

(1) The Q-function of the adversary agent is randomly initialised. The initial state s_i is arbitrary drawn from available states $s_t \in S$ to maintain the Q_{adv} and acquire the action value for the given state $Q_{adv}(s, a)$.

(2) The adversary agent takes action a_{adv} (selects an intrusion type) following its π_{adv} and the s_t.

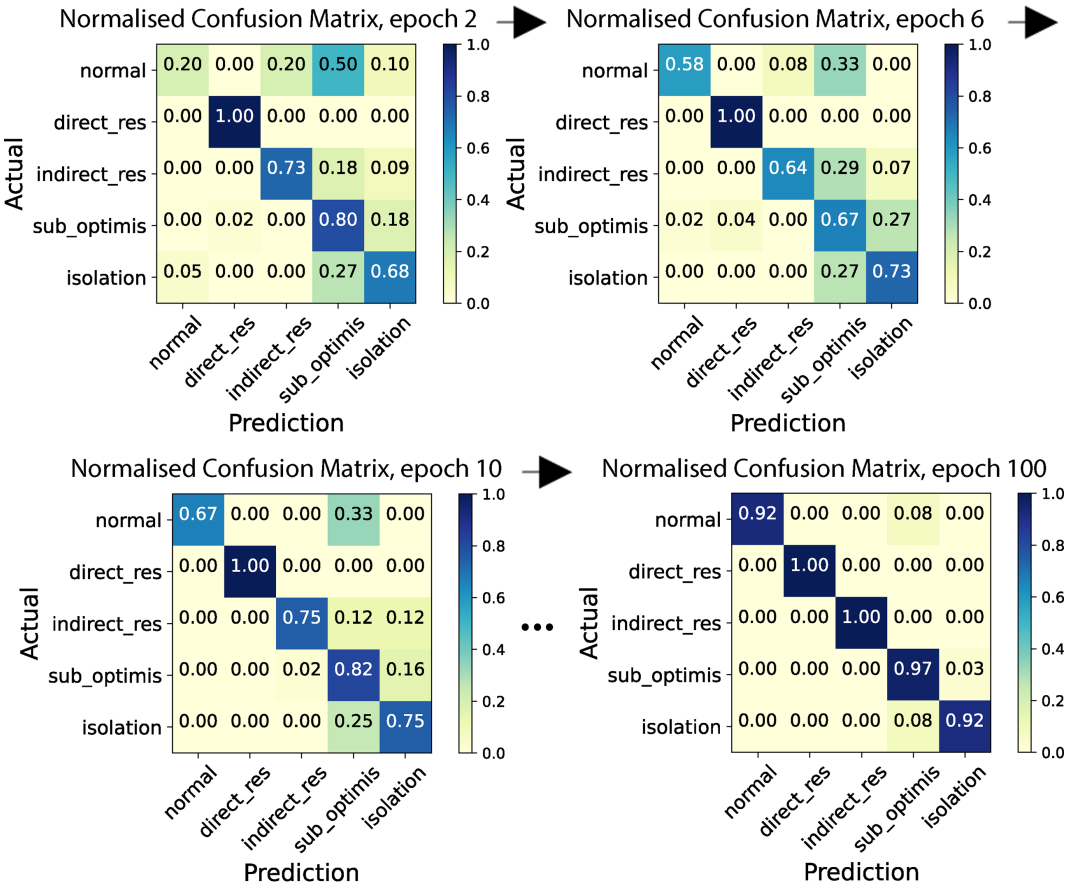


Figure 5.3: *The evolution of a defender agent against an adversarial agent in the initialisation phase (confusion matrix).*

(3) The environment draws an instance (X_t, y_t) from the labelled dataset, where $y_t = a_{adv_t}$, and returns s_t to both agents. X_t is then sent to the defender to classify.

(4) Given the state drawn by the adversary agent inside the environment, the defender agent attempts to classify s_t using its classifier and takes an action a_{def}.

(5) The defender computes a class prediction for the given s_t, where $s_t = X_t$, and takes action a_{def} by sending its prediction $\overline{y}_t$ to the environment; if $\overline{y}_t = y_t$, a positive reward R^+ will be given to the defender agent, otherwise adversary agent will receive the R^+.

(6) Adversary agent takes action $a_{adv_{t+1}}$ using policy derived from $Q^{\pi}_{adv}(s_t, a_t)$. The s_{t+1} is generated based on $a_{adv_{t+1}}$.

(7) With the reward values obtained and the next states inferred, the policy function of the adversary agent is updated according to the DQN update rule, and the defender agent updates its classifier using the actual label y_t of the observation x_t.

After the initialisation phase, the proposed scheme preserves the observations of minority and majority classes that contribute to the current defender model generation, denoted as D_{past}.

Drift-detection: In non-stationary stochastic environments, the defender agent faces an additional problem i.e., detecting concept drift, which is an arbitrary, partially perceptible

shift in the underlying distribution of the objects of interest. Formally, the concept drifts in the target environment can be defined as follows, given a time period $[t_0, T]$, a set of samples, denoted as $C = \{c_t, c_{t+1}...c_T\}$, where $c_i = (X_i, y_i)$ is one observation, X_i is the features and y_i is the target label. Concept drift at time t can be defined as the change of joint probability of X and y at time t, denoted as equation 5.4. Since the joint probability $P_t(X, y)$ can be decomposed $P_t(X, y) = P_t(X) \times P_t(y|X)$, concept drift can be classified as virtual concept-drift (where $P_t(X) \neq P_{t+1}(X)$ while $P_t(y|X) = P_{t+1}(y|X)$), actual concept-drift (where $P_t(y|X) \neq P_{t+1}(y|X)$ while $P_t(X) = P_{t+1}(X)$) or the combination of both.

$$\exists X : p_t(X, y) \neq p_{t+1}(X, y) \tag{5.4}$$

The fundamental function of the concept drift detection approaches is the mechanism to detect the drift occurrence's start time-point, drift period, and end time-point. Accurate and prompt identification of the drift occurrence plays a vital role in the learning system of the proposed scheme. The proposed scheme, employs the Drift Detection Method (DDM) [121] algorithm for detecting concept drift occurrence. In the streaming data environment, the ground-truth are available to the model sequentially with delay. When a new (X, y) is available, it is classified using the actual model, where X is the vector of features for different attributes and y is the ground truth. In this chapter, the proposed scheme identifies concept-drift using the reward prediction accuracy. Every timestamp, t, the ratio of errors is calculated as the probability of misclassifying (p_t), with standard deviation (σ), denoted as $\sigma_t = \sqrt{p_t(1-p_t)/t}$. In online learning, the statistical theory proves that when the data distribution is stationary, the p_t reduces [122]. On the other hand, when the distribution shifts, the p_t will increase, and the defender model becomes inappropriate. If the concept is unaffected, then the $1-\beta$ confidence interval for p_t with $n > 30$ observations is approximately $p_t \pm z_\beta/2\sigma_t$, where $z_\beta/2$ denotes the $(1-\beta/2)th$ percentile of the standard normal distribution. The β depends on the confidence level. DDM manages the values of p_{min} and σ_{min} and updates them with p_t and σ_t at time stamp t if $p_t + \sigma_t \leq p_{min} + \sigma_{min}$.

In order to compute an appropriate concept window size, the proposed scheme utilises a warning level. The concept window includes the earlier observations that are joint with the recently observed concept and a minimal representation of instances from the preceding concept. Suppose that there is an observation at time t with corresponding p_t and σ_t in the sequence of streaming observations that traverse a node. The utilised concept-drift detection method stores observations in short-term memory while $p_t + \sigma_t \geq p_{min} + 2\sigma_{min}$ is satisfied. It rebuilds the classifier from the stored examples and resets all variables if $p_t + \sigma_t \geq p_{min} + 3\sigma_{min}$. DDM raises a warning level to indicate the possible occurrence of drift. The utilised threshold for the warning level is the relaxed version of the drift level; the warning level and the drift level are set p-values of 2σ (95%), and 3σ (99.7%), respectively. The data gathered between the warning and drift levels are applied as the training set for updating a learning model. In the proposed scheme, the severity of concept drift indicates the difference between current concept and the earlier one, denoted as $\Delta = \delta(P_t(X, y), P_{t+1}(X, y))$, where δ measures the discrepancy between D_t and D_{t+1}, and t is the timestamp of the concept-drift occurrence. In the proposed scheme, the Δ measures the difference between $\hat{p}_{min}$ and $\hat{p}_i$, denoted as $\Delta \sim \hat{p}_i - \hat{p}_{min}$. A very large value of Δ shows a high severity in the concept drift.

Concept-drift Adaptation: The concept-drift adaption is a mechanism for generating, updating and selecting one among several defender models $(\mathcal{M})$, shown in Algorithm 13.

Algorithm 13: Concept-drift Adaption

Initialisation
new_model : Generate a new defender agent (Initialisation phase)
E : Quality signal; δ : threshold
$\mathcal{M}$: Set of all available defender models
m_{cur}: The current defender model
η: Maximum number of defender models (assigned to ten)
y: Ground truth

if $m_{cur} = \varnothing$ **then**
  $m_{cur} \leftarrow new\_model()$
  $\mathcal{M} \leftarrow \mathcal{M} \cup \{m_{cur}\}$
$a_t \leftarrow \pi_{m_{cur}}(s_t)$
Observe y_t and reward r_t
for $\forall m \in \mathcal{M}$ **do**
  $\Delta R_m \leftarrow Q(s_t, a_t) - r_t$
  $e_m \leftarrow 1 - 2\sigma(\frac{(\Delta R_m)^2}{(R_{max} - R_{min})^2})$
  $E_m \leftarrow E_m + \rho(e_m - E_m)$
$m_{cur} \leftarrow arg \max_{m \in \mathcal{M}} (E_m)$
if $E_{m_{cur}} < E_{min}$ **then**
  if *length($\mathcal{M}$)* $= \eta$ **then**
    discard $arg \min_{m \in \mathcal{M}} (E_m)$
  $m_{cur} \leftarrow new\_model()$
  $\mathcal{M} \leftarrow \mathcal{M} \cup \{m_{cur}\}$
$s_t \leftarrow s_{t+1}$

The defender models have incrementally developed their detection performance on the given network observations. Since the occurrence of concept drift causes a negative impact on the defender agent's performance, the defender agent should re-adapt to concept drifts to maintain its performance. The concept drifts intentionally (opponent strategy) or unintentionally (evolving environment) drive the defender agent to gradually hinder its former policy while adapting a new one (known as catastrophic forgetting [167]). Hence, the proposed scheme maintains previous copies of the defender agent to circumvent this issue.

To maintain the detection performance, the IDS should systematically adjust to unforeseen data distributions. This chapter implies that maintaining multiple representations of defender agents (and their respective policies) can enhance the system performance versus different concepts in the evolving data stream. The employment of multiple defender agents enables the proposed scheme to partition the knowledge of environments with different concepts into representative models. Each model ($m \in \mathcal{M}$) is associated with being responsible for analysis of a particular type of concept drift. In this regard, one model can be active at any given time t. On detection of concept drift, the active model will be replaced.

In the proposed scheme, the quality factors [168] are used to compute a total quality for the defender agent in terms of the instantaneous error in reward prediction accuracy, using $\rho \in (0, 1]$ to weight the impact of new measurements. Where ΔR_m denotes difference between expected reward $Q(s, a)$ and actual reward r.

$$e_m = 1 - 2\sigma(\frac{(\Delta R_m)^2}{(R_{max} - Rmin)^2} \tag{5.5}$$

$$E_m \leftarrow E_m + \rho(e_m - E_m) \tag{5.6}$$

An agent maintains this error signal, for all concept models $m \in \mathcal{M}$, including its active concept m_{cur} and all stored concepts, and uses E_m to explicitly identify concept-drifts, Eq. 5.6. Whenever some previous model m_i outperforms the current defender agent, m_{cur}, the system re-activates m_i. A minimum quality threshold, E_{min}, is utilised to specify the minimum performance performance execution. When $\forall m \in \mathcal{M} : E_m < E_{min}$, a new defender agent will be created (through an initialisation phase). To overcome the issue of class imbalances, the new defender model needs to collect observations of minority classes in the past concepts and propagate them into the latest concept ($D_{train} \leftarrow D_{past} \cup D_{new}$) through the initialisation phase. Observations seen in previous concepts should be preserved and accessed. The system begins with only one model m_{cur} and then gradually develops new ones when they needed. Note that, the model recognition and re-initialisation enable the proposed scheme to mitigate the catastrophic forgetting dilemma [167]. Furthermore, the adoption of adversarial RL enables the proposed scheme not to load the entire training data into memory and only utilise the defender model that is relevant to the current data distribution.

5.4 Adversarial ML-based Attack

In this type of attack, the adversary uses its policy π_{adv} and state s_t to select observations of different attacks from its attack profile ($profile_{adv} = \{X_1, X_2, ...X_n\}$, $where\ y_i|X_i\ is\ a_i \in A_{adv}$) to distort the exposure of the adversary node. The exponentially decaying ε-greedy strategy [159] enables our adversary model to explore and exploit different intrusions in its attack space (action-space A_{adv}). Each state $s_i \in S$ is a feature vector set of network features $X = \{< x_1, x_2...x_n >\}$ (shown in Table 5.3). In this scheme, incrementally, the adversary crafts adversarial policies to evade the defender agent.

The primary policy of the adversary node is to select malicious observations (take actions) that the defender IDS would misclassify as benign. After sufficient attempts, the adversary node possesses the cunning to select malicious observations with a lower degree of distortion to reduce the probability of intrusion exposure. In reality, the Network-based IDS (NIDS) performs in ghost mode and is not perceivable by malicious nodes [3]; however, in this chapter, we assume that the adversary can perceive the reaction of the IDS to its attack. This chapter concentrates on two primary attack strategies that the adversary could perform:

(1) **Black-box setting:** The adversary has zero knowledge about the features and parameters that the defender agent employs for detecting intrusions; however, it can observe the defender's reaction (a_{def}) to the intrusions that it initiates (a_{adv}) at the state (s_t), and receives reward (R^+) or penalty (R^-) with regards to that.

(2) **Grey-box setting:** In this type of attack, the adversary performs in a grey-box setting, where it has constrained knowledge or limited access to the defender configurations and can perceive the impacts of its actions by observing the next-state (s_{t+1}) and reward (r). The grey-box setting indicates that the adversary can observe the defender's actions

and knows all properties and features that the defender extracts and monitors to detect intrusions;

In both adversarial attacks, initially, the adversary has zero knowledge about the defender agent model and reinforces its policy gradually by exploration and exploitation. The adversary tries to find a malicious behaviour that will be misclassified to class $c_{adv} \notin c_{attack}$ while also trying to minimising its exposure probability in the long term. The adversary selects instances of multiple attacks from its dataset through each attack profile (depicted in Fig. 5.4), where each attack profile constitutes a hundred observations. Note that, there are some works (e.g., [169; 170]) where a perturbation value has been applied to the samples in order to increase the misclassification errors of the classifier. However, in 6LoWPAN, applying adversarial perturbation would cause the network traffic to lose its functional behaviour and make the attack impractical. For instance, if the adversary increases a rank value to avoid exposure, the altered control packet might fail to attract any neighbouring nodes. Hence, in our proposed scheme we do not allow the adversary to carry out any perturbation on the sampled network traffic.

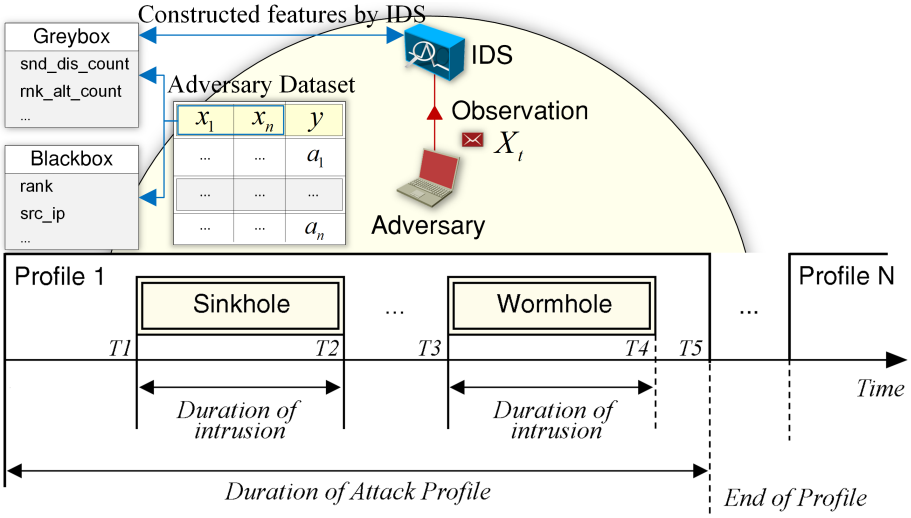


Figure 5.4: *Combinational malicious activities.*

5.5 Evaluation

In order to show the effectiveness of the proposed scheme we now conduct several experiments using the Netsim emulator to simulate various routing attacks against 6LoWPAN. During our evaluation process, we consider different network configurations (as shown in Table 5.2). In this regard, the emulated network configurations contain 16 to 128 LLN nodes (excluding 6BR and external PCs), where 30% of nodes are malicious. In each scenario, 30% of the nodes were in movement with a velocity of 5 m/s. All simulations are performed for ∼240 minutes. In our experiments, we consider sixteen network topologies, where half of the topologies are for the initialisation phase (section 27) and another half for evaluation. In this regard, eight networks are in hexagon-shaped (photonic) topologies. In another eight topologies, nodes are

randomly distributed on the terrain. In all the topologies, nodes distribute over the terrain with $250m^2 \sim 800m^2$ and 20∼45 metres apart.

The emulated network scenarios have a streaming data environment. The nodes are mobile with network data generation profiles that vary over time. In the evolving environment, the IDS needs to analyse streaming data with an imbalanced data distribution, where the majority of observations are legitimate, and a minority of them are abnormal. In this context, classifying an instance from the majority class (legitimate activity) as an instance from the minority class (malicious activity), is known as a false positive alarm. Although a false-positive error is undesirable, it is less critical than misclassifying an observation from a minority class as a member of the majority class (classifying malevolent observation as a legitimate activity), a so-called false negative. It should be noted that the exisitng solutions in the literature evaluate their proposed models using two types of evaluation metrics, namely the threshold metrics (e.g. accuracy and F-measure), and the ranking methods and metrics (e.g., receiver operating characteristics (ROC) analysis and area under the curve (AUC)). In our proposed scheme the developed models are evaluated against both types of evaluation metrics.

It is crucial to assure that we are collecting data that will help us to draw conclusions. Hence, after feature extraction, we perform data wrangling [171] to pre-process the data and getting it into a shape that can be utilised for data analysis (using inferential and descriptive statistics to identify potential data issues and perform the sanitisation, normalisation, discretisation, and data encoding [127]). Next, to facilitate the intrusion detection task, this chapter performs exploratory data analysis (EDA) over different routing intrusions to analyse them in more depth and extract the network irregularities they cause on the 6LoWPAN.

To perform EDA, here we simulate several pairs of networks $(\mathcal{A}, \mathcal{B})$ where $\mathcal{A}$ contains only the legitimate nodes and $\mathcal{B}$ contains both the legitimate and malicious nodes. Observing the statistical difference of control and application packets in $\mathcal{A}$ and $\mathcal{B}$ enables this study to identify the adverse impact that each RPL attack causes in $\mathcal{B}$. Through EDA, we utilise visualisations and summary statistics to obtain a better perception of the simulated data. EDA helps identify appropriate features for our ML-based intrusion identification classifiers. Fig. 5.5 illustrates the outcomes of EDA. Next, to illustrate the importance of the engineered features as shown in Fig. 5.6 here we use the Mean Decrease in Impurity (MDI) importance metric.

5.5.1 Data-set and feature construction

In 6LoWPAN, the majority of RPL attacks' observations are slightly imbalanced (fall in minority class) in nature. Our experiments show that sinkhole (SH), blackhole (BH), grayhole (GH), DIS flooding (DA), increase rank (IR), wormhole (WH), DIO suppression (DS), and worst parent (WP) imbalance ratios ranging from 4:100 up to 56:100. In [62], authors applied SMOTE to balance the offline training data for a stationary 6LoWPAN environment. However, in the streaming data environment, the incremental classifier must be dynamic and adapt to concept drifts where the new data distribution is severely imbalanced (contains no or negligible proportion of different minority classes). The problem with imbalanced data-sets is that classifiers are often biased towards the majority class (legitimate activities); hence there is a higher misclassification rate in the minority classes (representing positive cases). In this chapter, we develop an adversarial reinforcement learning environment (discussed in section

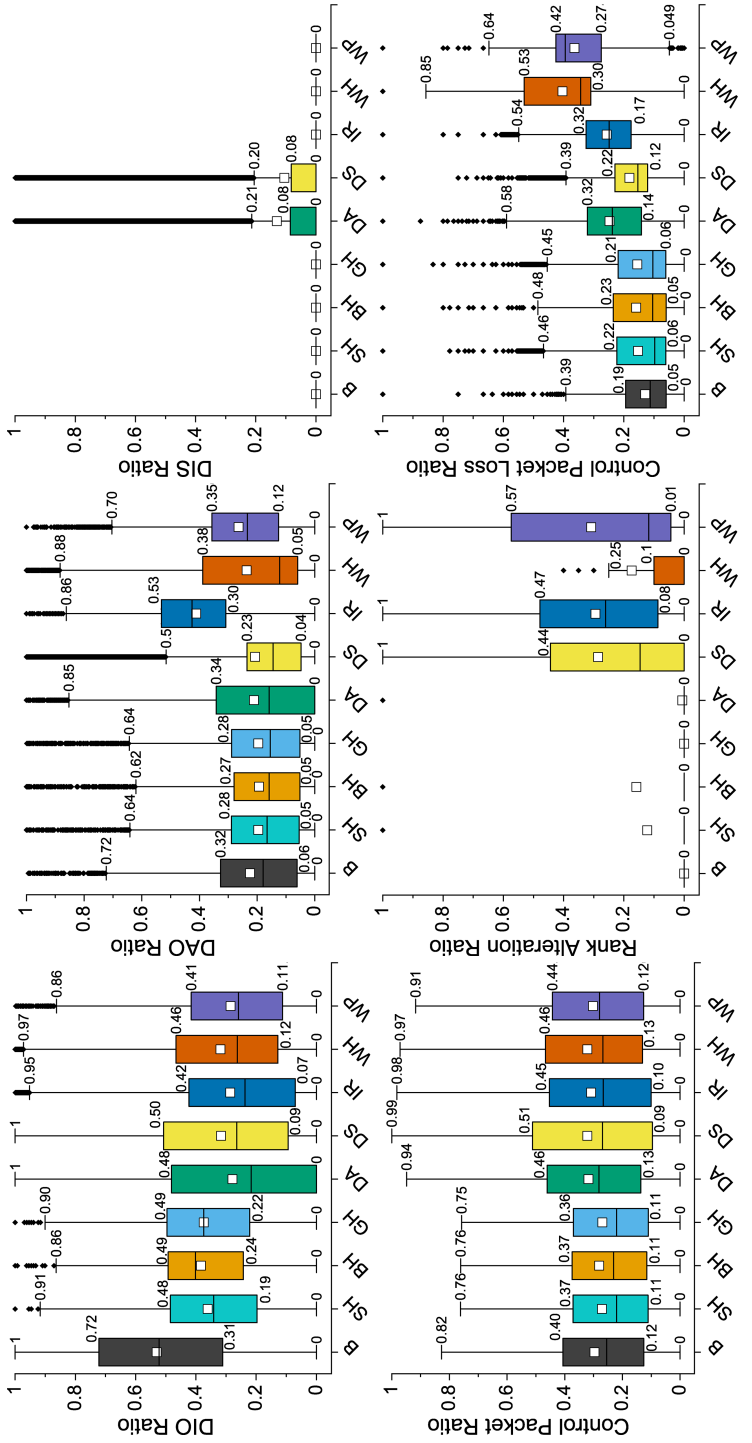


Figure 5.5: *Exploratory Data Analysis (EDA) outcomes. (B stands for Benign)*

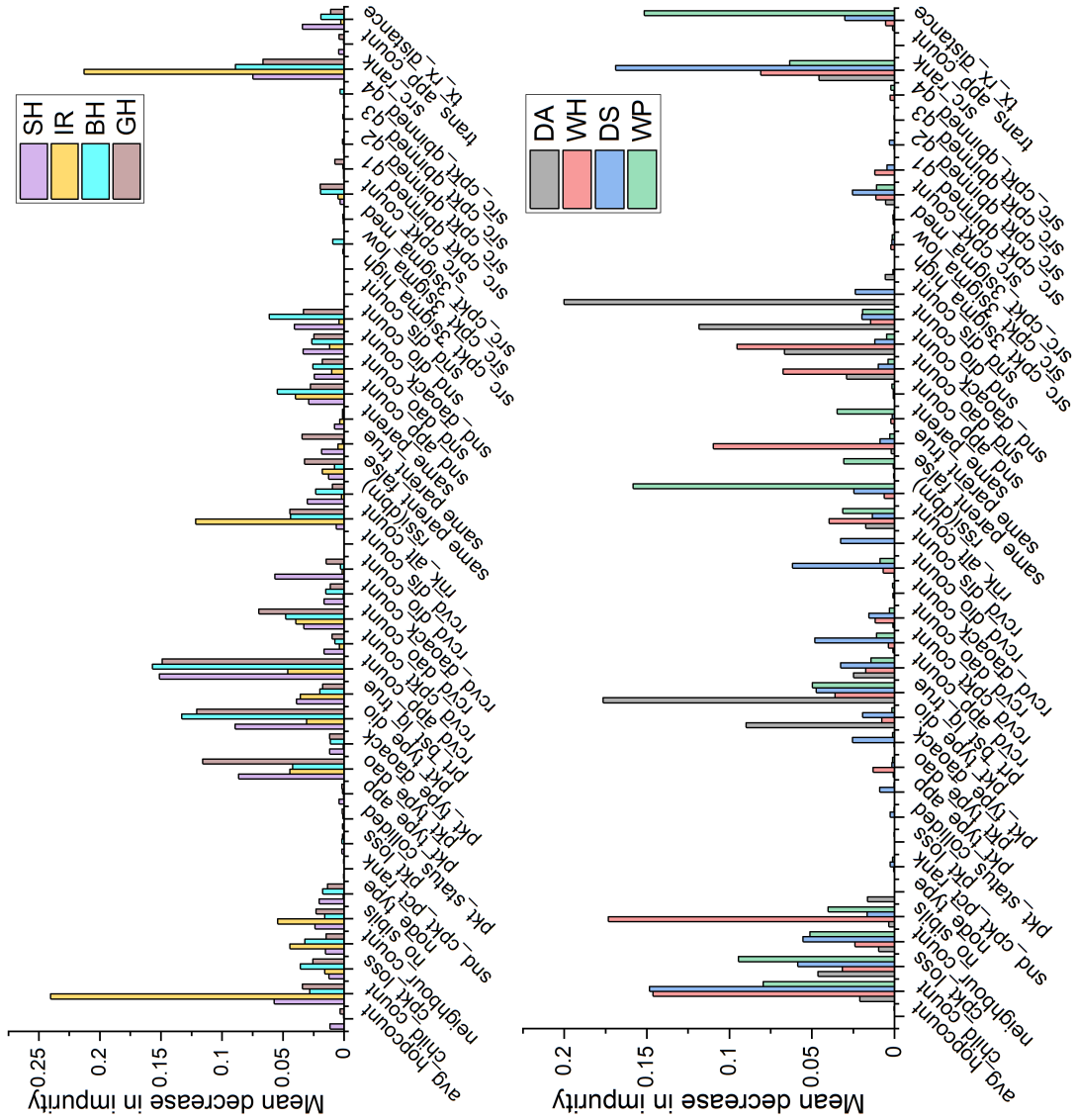


Figure 5.6: *Feature importance.*

Table 5.2: *Simulation parameters*

Parameters	**Values**
Number of nodes	16, 32, 64, 128
Number of Malicious nodes	∼30%
Number of Workstations	4, 8
Transmission Range	50m
Number of ML detectors	∼10%
Number of Mobile nodes	∼30%
Scenario Dimension (Terrain)	(250 × 250) to (850 × 850) s.metres
Traffic Rate	250 kbps
Simulation time	∼ 14,400 seconds
Application Protocols	COAP, CBR
RPL mode	Storing mode
Mobility Modes	Random Walk, Group Walk
Path Loss Model	Log Distance, Exponent(n): 2
Distance between nodes	20 ∼ 45 m
Objective Function (OF)	OF0, LQ
Receiver Sensitivity	-85 dBm

27) to address this issue. The proposed approach eliminates the fundamental imbalance issue that plagues defender agent initialisation. Analyses of the EDA outcomes demonstrate the impact of each routing attack on the extracted features from 6LoWPAN traffic (e.g. DIO, DAO, and DIS control packets). In this section, we perform feature engineering to construct features that can reveal the abnormalities caused by each RPL attack.

Engineering a set of salient features has a vital role on how accurately an IDS classifies all types of RPL attacks in the streaming data environment. The extracted features can enable the anomaly-based classifiers to correctly identify all the anomalies through training over normal instances and make signature-based classifiers classify each type of RPL attack accurately. The raw instances of 6LoWPAN simulations contain a set of features that are not applicable for conducting intrusion detection tasks. For instance, including features that represent nodes identities (e.g. IP address, MAC address, and node id) can prevent the proposed scheme from having general applicability. Since in this chapter we employ a passive decentralised monitoring approach [84], any features that require the internal configuration of legitimate nodes (e.g. power consumption, geographical location, CPU/RAM/ROM usages) are excluded.

We now extract three types of features, namely basic features, history-based features and connection-based features. Basic features contain the general information about the nodes that are derived from ICMPv6 control packet headers. The history-based features provide information about the number of times that the current node (the sender of current packet) sends or receives a specific type of application or control packet. Connection-based features carry salient information about the routing configuration of sender (RSSI, link quality etc) and the number of collided control and application packets perceived by an IDS detector. Based on our observations, the connection-based and history-based features play a vital role in detecting routing attacks in 6LoWPAN. Table 5.3 depicts the set of features engineered in this chapter. A row describes an observation about an entity. A feature describes some property of the observation. In this streaming data environment, observations are pre-processed

Table 5.3: *Engineered features*

	Feature	Description
Basic	pkt_type	Type of packet (DIO, DAO, DIS, App etc)
	pkt_status	Packet status (Collided, Successful)
	src_rank	Sender rank in DODAG
	rcv_rank	Receiver rank in DODAG
	adv_vn	Advertised version number
History-based	snd_dis_count	No. of DIS unicasted/multicasted by sender*
	snd_dio_count	No. of DIO advertised by sender*
	snd_dao_count	No. of DAO unicasted by sender*
	snd_daoack_count	No. of DAO-Ack unicasted by sender*
	rcvd_dis_count	No. of DIS rcvd by current node in past*
	rcvd_dio_count	No. of DIO rcvd by current node*
	rcvd_dao_count	No. of DAO rcvd by current node*
	rcvd_daoack_count	No. of DAO-Ack rcvd by receiver*
	rcvd_cpkt_count	No. of control packets rcvd by receiver*
	snd_cpkt_3sigma	cpkt sent by sender (σ, 2σ, and 3σ)*
	snd_cpkt_pct_rank	Pct-change-rank of cpkt sent by sender*
	snd_ctrl_count	No. control packet issued by sender*
	avg_intpkt_time	Average delay between pkts issued by snd
	rnk_alt_count	No. rank alteration by sender*
	prt_alt_count	No. times sender changed its parent*
	vn_alt_count	No. version number alteration by sender*
	trans_app_count	No. of application trans by sender*
	pkt_e2e_delay	Packet end-to-end delay
Connection-based	cpkt_loss	Control packet loss ratio
	pkt_loss	Application packet loss ratio
	avg_hopcount	Average No. of hopcount (global view)
	neighbour_count	No. of neighbouring node
	child_count	No. of children
	same_parent	Sender and the detector have same parent
	rx_sen	Average receiver sensitivity
	tx_pwr	Average transmission power
	rssi	Received signal strength indicator of sender
	prt_bst_lq	Current parent provide best link quality

*In the past 5 seconds.

(sanitised, normalised, discretised, and encoded) on the fly through each window interval.

5.5.2 Performance evaluation and discussion

In this section, performance analysis is undertaken to determine the effectiveness of the proposed scheme using different measures. Here, we intend to detect the direct-resources-

topology (dis-flooding), indirect-resources-topology (increase-rank), sub-optimisation-topology (sinkhole, wormhole, replay, dio-suppression, and worst-parent), and isolation-topology (black-hole, and grayhole) attacks against 6LoWPAN. The streaming class-imbalanced data causes the development of a model with accuracy paradox performance. As the accuracy metric does not differentiate the numbers of correctly classified instances in the minority and majority classes, it provides erroneous evaluation results in the streaming data environment. Therefore, we apply F-measures to balance the evaluation of precision and recall. Since in IDS the sensitivity (TPR) is more desirable than specificity (TNR), we now measure the performance of the proposed scheme in terms of both F2-measure (β=2, less weight on precision, and more weight on recall) and F1-measure (β=1, equal weighting on precision and recall). On the other hand, since the ranking metrics do not make any assumptions about class distributions, this chapter also measures the performance of the proposed scheme in terms of ROC and AUC [172]. In order to evaluate the proposed scheme, we have conducted seven experiments and analysed the outcomes. We evaluate the performance of the proposed scheme in three different 6LoWPAN environments: **(i)** stationary (to demonstrate whether the integration of ARL and concept-drift algorithms cause a negative impact to the performance of the defender agent). **(ii)** non-stationary (to demonstrating the performance of the proposed scheme against concept drifts). and **(iii)** non-stationary with black-box and grey-box adversaries (adversarial ML-based attacks discussed in Section 5.4). This chapter implements shallow multi-layer perceptrons (with 2 hidden layers, 100 nodes, Adam optimiser, Huber loss, and Relu activation function) for DQN and DDQN. Experimental results show that our proposed method offers better performance than existing IDSs for 6LoWPAN [3].

Table 5.4: *Performance bench-marking*

N	M	F2_measure ($\beta = 2$)								FPR							
		SH	***BH***	***GH***	***DA***	***IR***	***WH***	***DS***	***WP***	***SH***	***BH***	***GH***	***DA***	***IR***	***WH***	***DS***	***WP***
16	No	99.29	99.19	98.49	100	98.89	98.32	98.59	98.69	0.04	1.2	0.28	0	0.05	0	0.06	0.26
	Yes	99.2	98.99	99.69	100	98.79	99.62	99.29	97.58	0.08	0.14	0	0	0.02	0.07	0.02	0.48
32	No	98.89	99.69	99.49	100	99.49	99.79	99.8	91.29	0.07	0.04	0	0	0.04	0.04	0.02	0.48
	Yes	99.59	99.49	99.49	100	99.79	100	99.89	90.75	0.06	0	0.02	0	0.04	0	0	0.49
64	No	99.79	99.09	98.09	100	99.89	99.69	92.34	90.2	0	0	0.06	0	0	0.06	0.29	0.57
	Yes	99.09	99.69	96.98	100	98.59	99.89	94.68	90.95	0	0	0.06	0	0	0	0.31	0.47
128	No	99.59	99.39	95.77	100	98.79	99.8	93.24	93.28	0	0.02	0.12	0	0	0.02	0.57	0.22
	Yes	99.19	99.29	96.27	100	98.12	99.79	90.09	91.68	0	0.04	0.04	0	0	0.04	0.47	0.71

N: No. Nodes (here ∼30% of nodes are malicious); **M:** Mobility; **SH:** Sinkhole, **BH:** Blackhole; **GH:** Grayhole; **DA:** DIS Flooding; **IR:** Increase Rank; **WH:** Wormhole; **DS:** DIO Suppression; **WP:** Worst Parent;

Experiment 1. To evaluate the performance of the proposed scheme we now apply different discount values for the adversarial agent and the defender agent in the initialisation phase. Here, our aim is to find the best γ_{def} and γ_{adv} values for the proposed scheme. Fig. 5.7 shows our experimental results with different exploration and exploitation parameters (ε-value strategies) for the adversarial and defender agents (in the initialisation) to find configurations that facilitate IDS convergence to high performance over all types of attacks. Fig. 5.7 provides convincing evidence in favor of a defender agent with ε-value between 0.01 and 0.2. In order to find a suitable discount factor for our proposed scheme, we evaluate the performance of the proposed scheme using different (γ). As illustrated in Fig. 5.8, a lower

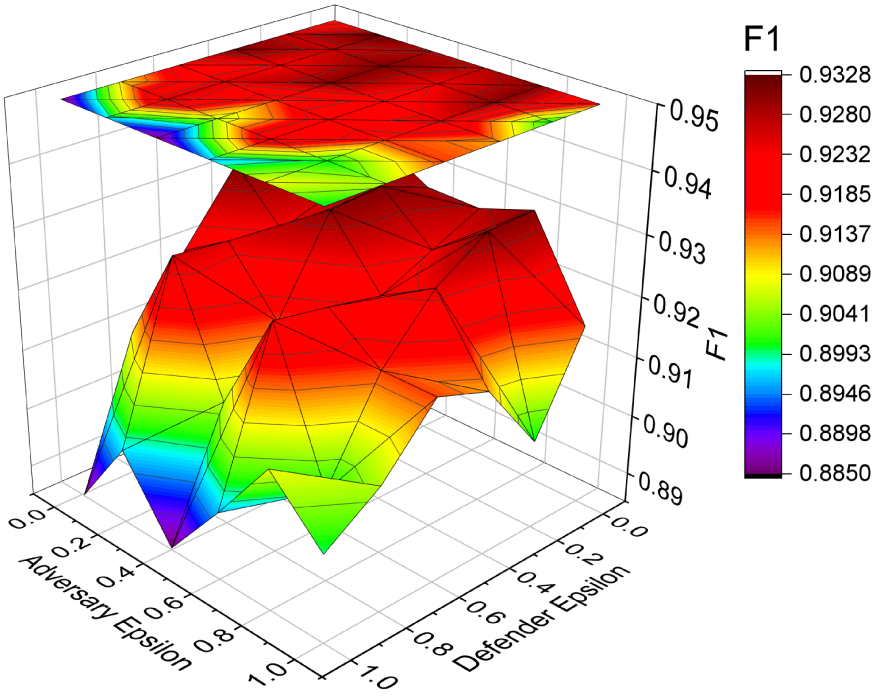


Figure 5.7: *Defender agent and adversarial agent minimum ε-values.*

discount value for defender agent can ensure a higher F2 ($\sim$95%) for the defender agent. Hence in this chapter we associate the discount factor close to zero (γ_{def}=0.001).

Experiment 2. In this experiment we identify the strengths and limitations associated with the defender agent under different RPL attacks. Fig. 5.9 illustrates the frequency of initiated intrusions by the adversarial agent during a hundred epochs. This illustration shows how the adversarial environment systematically adjusts the imbalanced data in order to enhance the classification results. It is evident that the emerging frequencies of sub-optimisation and isolation topology attacks reflect the meagre competence of the defender agent against some intrusions within the corresponding attack profiles. Our investigation reveals that the striking resemblance between sinkhole, blackhole, and grayhole attacks is the reason behind the defender agent's inability to differentiate them. Thus, the adversary desires to take these actions more often to accumulate more rewards. The performances of DQN, DDQN, and KNNADWIN defender agents are depicted in Fig. 5.10. The demonstrated results are outcomes of ten runs.

Experiment 3. The Concept-drift detection method is a critical indicator for measuring the variance of a data distribution over time and is also regarded as a crucial mechanism for making the proposed scheme adaptive and robust against the evolving data environment of 6LoWPAN. In this experiment, we empirically evaluate different concept-drift detection algorithms to select one that can ensure the system's adaptivity and enhance the proposed scheme performance over time. The candidate concept-drift detection algorithms are: Adaptive Windowing method (ADWIN), Drift Detection Method (DDM), Early Drift Detection Method (EDDM), Kolmogorov-Smirnov Windowing (KSWIN), Drift Detection Method based on Hoeffding's bounds (HDDM) with moving weighted average-test (HDDM-W) or moving

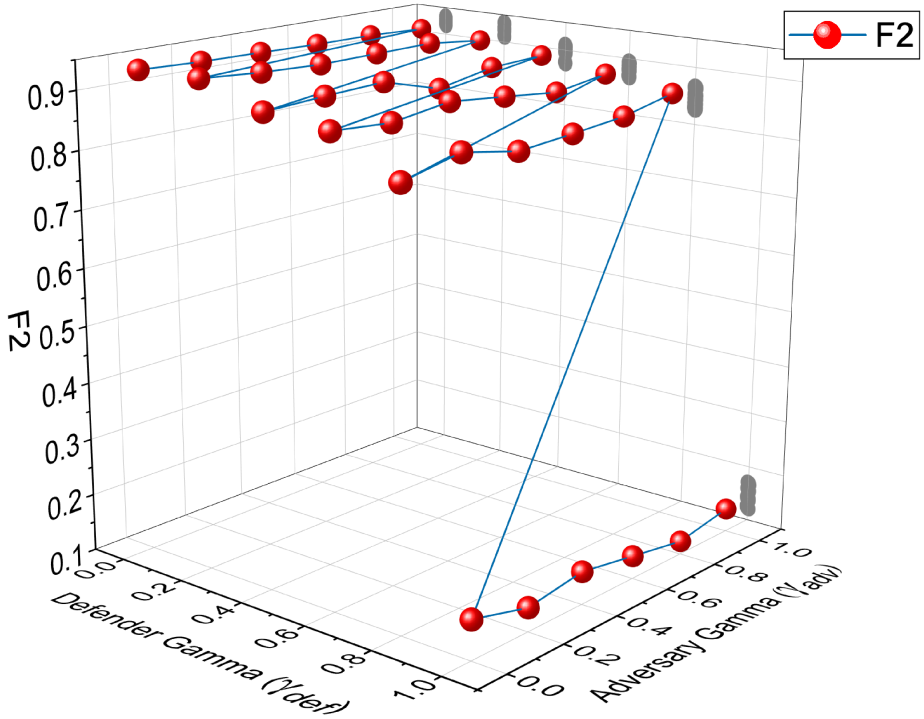


Figure 5.8: *Searching for γ_{def} and γ_{adv}*

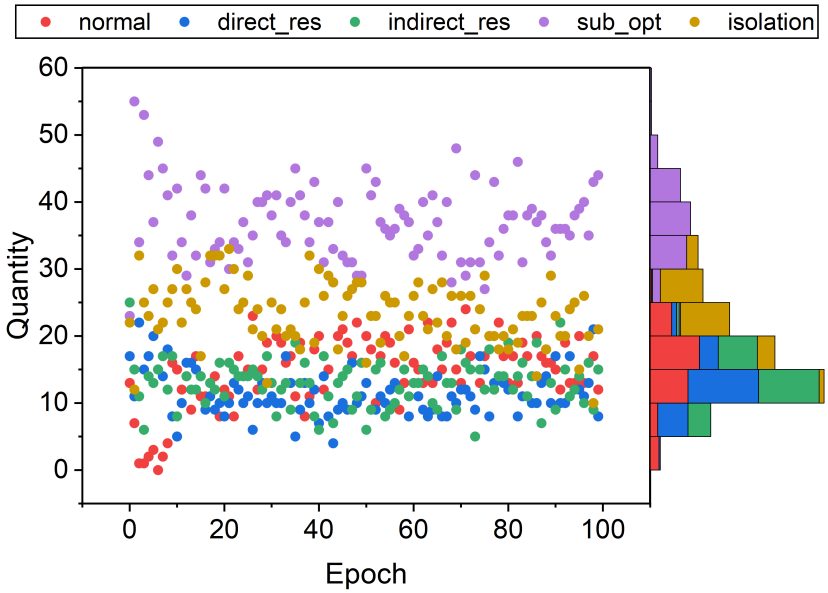


Figure 5.9: *Marginal histogram of the initiated intrusions by an adversary.*

average-test (HDDM-A) [135; 121]. Outcomes of our analyses are presented in Fig. 5.11. The general picture emerging from the analysis is that the performance of DDM and HDDM-A are higher than the other concept-drift detection methods

Experiment 4. In this experiment, we measure the time complexity and energy con-

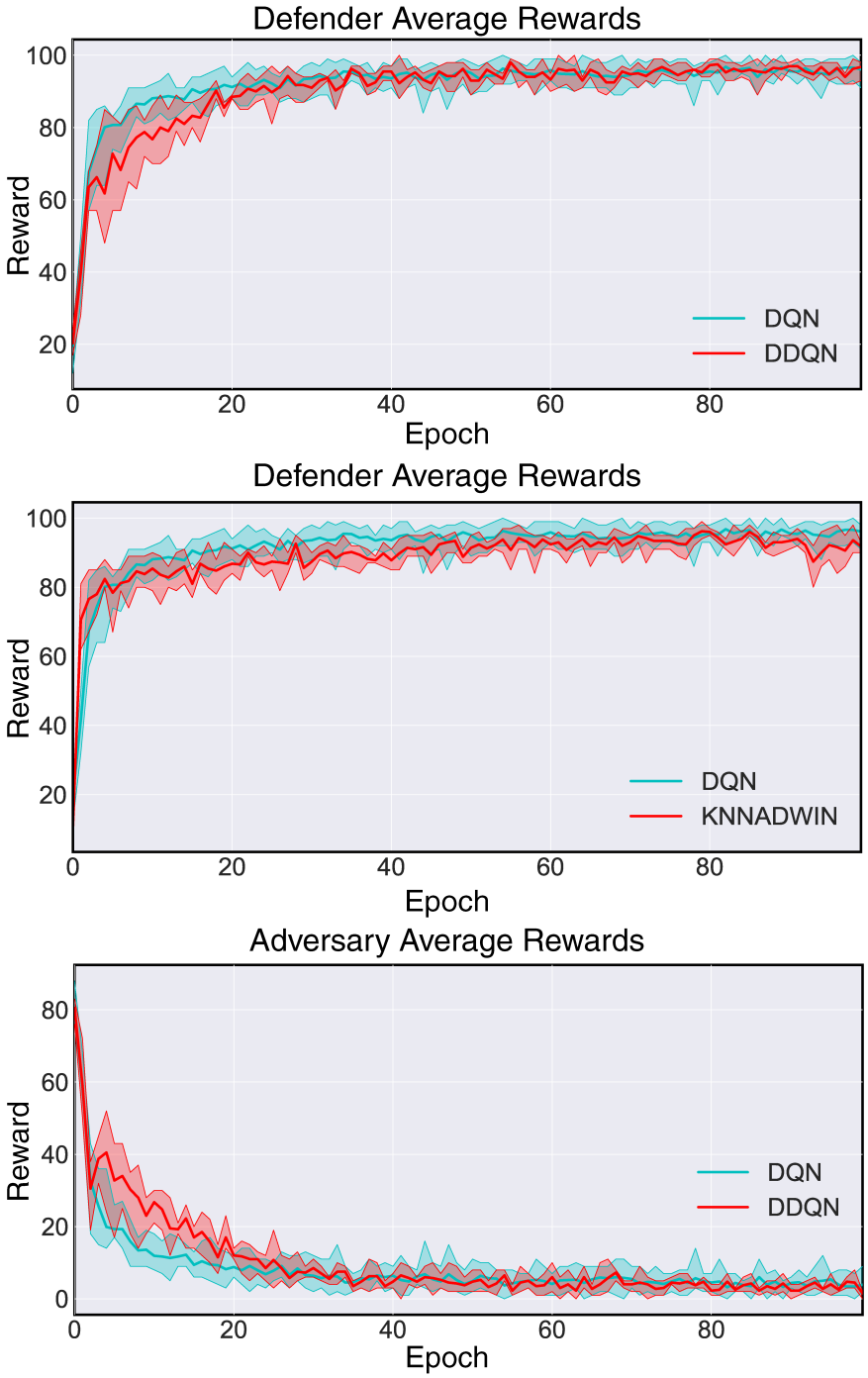


Figure 5.10: *Initialisation phase rewards (mean and confidence intervals for ten runs).*

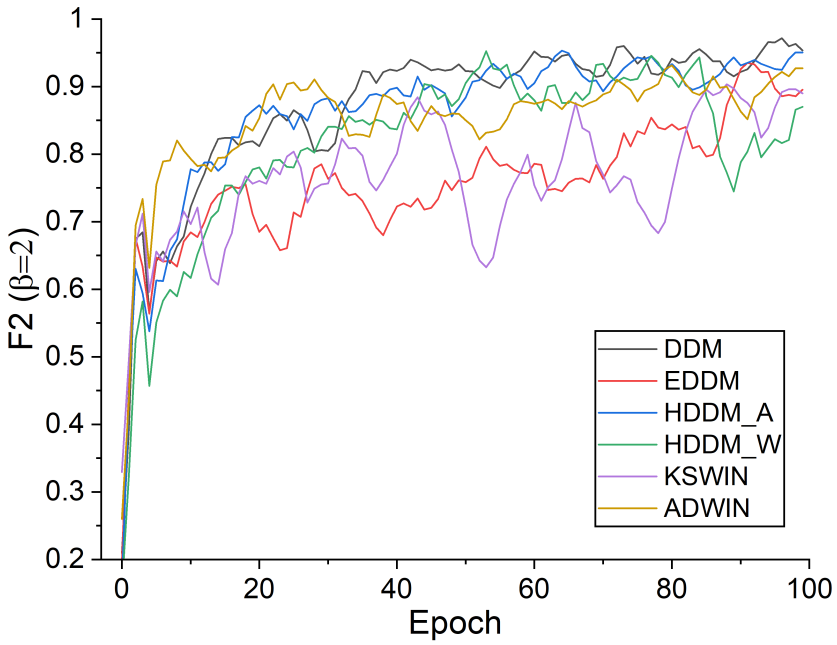


Figure 5.11: *Concept-drift detection methods against a grey-box adversary.*

sumption of the generated defender agent. In our experiment, we consider 64 LLN nodes in 6LoWPAN, where 20% of the nodes are assumed to be malicious. Next, we measure the prediction time complexity of the defender agent. In our experiments, a defender agent using KNNADWIN has $O(n) : 0.038 + 0.038 * n$ (linear) time complexity. On the other hand, DQN has $O(n)$ time complexity. As a rule of thumb, the $O(n)$ and $O(n\ log\ n)$ algorithms are considered linear time and desirable for online approaches [119]. In Fig. 5.12 we have illustrated the response times of KNNADWIN with different settings. On the other hand, a defender agent using DQN has a 10.7 milliseconds response time (average of thousand predictions).

In order to measure the power consumption of the defender agent, we use the Netsim Emulator feature to connect the physical microcontrollers (Raspberry Pi 4 4GB) with the simulation environment. By connecting a digital ammeter to the microcontrolers, we measure the energy consumption of the developed IDS. In this regard, we run our experiment for ten minutes while all unnecessary background tasks and applications are deactivated on the microcontrollers. On the basis of our observations, we find that the energy consumption of a KNNADWIN and the DQN in a LLN with 64 nodes were 1.8 J/s and 1.9 J/s, respectively, whilst a legitimate node without any IDS consumed 1.4 J/s. Therefore the KNNADWIN and DQN have 463.15 mJ/s and 485.54 mJ/s energy overheads on a LLN node, respectively. In Fig. 5.14, the largest generated KNNADWIN (with window-size 1000) occupies ∼440kb on a hard disk. The windows_size is the maximum size of the window storing the last viewed observations [161]. The response times of defender agents using different settings are included in Appendix C, (i.e., Fig. C.6.) of the supplementary material.

Experiment 5. In this experiment we evaluate our proposed scheme to show how it can ensure the security of 6LoWPAN upon various RPL attacks. Tables 5.4 highlights the benchmarking outcomes of the proposed scheme against different RPL attacks in the literature. In Table 5.4, ∼30% of nodes (average number of malicious nodes in the literature [3])

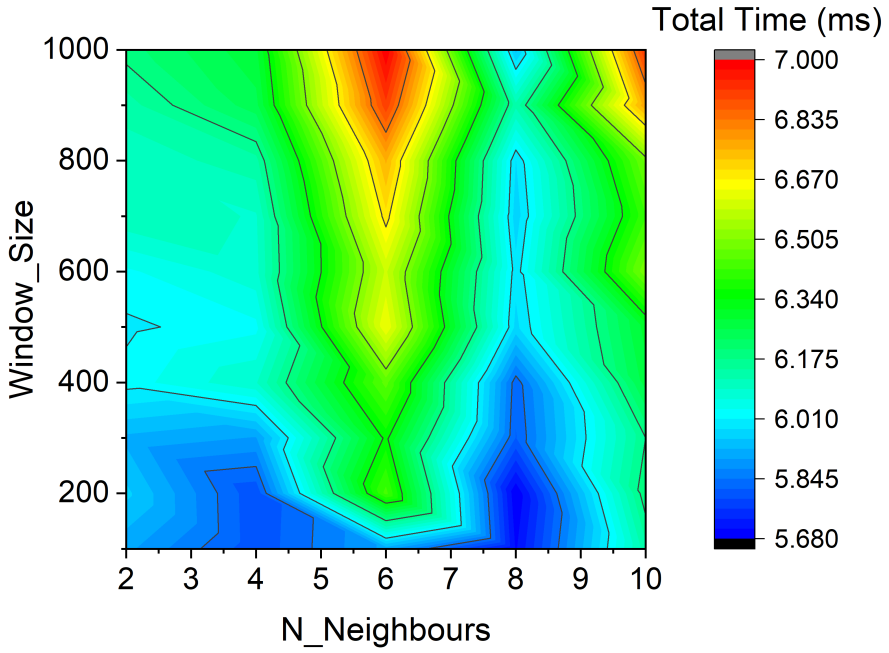


Figure 5.12: *KNNADWIN prediction time complexity.*

were associated as malicious.

Experiment 6. In this experiment, we evaluate the performance of the proposed scheme in detecting the RL-based (DQN) black-box and grey-box adversaries (discussed in section 5.4). Here, we implement the adversarial ML-based attacks to deteriorate the performance of the proposed scheme. The adversary uses the query-response pairs $Q_{adv}(s, a)$ to craft an adversarial intrusion and find optimal destabilisation policy. Hence this experiment is divided into two sub-experiments. In both experiments, the primary aim of the adversary is to maximise its total reward by malevolently luring the defender to mis-classify. The detection performance of proposed scheme is illustrated in Fig. 5.13. In the black-box adversary, we consider the adversary has access to several basic features (src_ip, dst_ip, packet_type, DODAG version number, and advertised rank). Our proposed scheme received 96.3% accuracy and recall, 96.29% precision, and 96.29% F1 against a black-box adversary (average results of ten runs), while it receives 92.2% accuracy and recall, 92.22% precision, and 92.19% F1 against grey-box adversary (average results of ten runs). Next, we benchmark the performance of different standard ML algorithms along with SMOTE oversampling approach against a grey-box adversarial ML-based attacks, shown in Fig. 5.15.

Experiment 7. In this experiment, we first compare the performance of the proposed scheme against different standard ML algorithms (e.g. support vector machine, decision tree, random forest, and k-nearest neighbors). We apply different oversampling and undersampling techniques to change the composition of the training dataset and enhance model performance against imbalanced training data; namely, the Synthetic Minority Over-sampling Technique (SMOTE), Adaptive Synthetic (ADASYN), Random Over Sampling (ROS), and Random Under Sampling (RUS). Our testing environment has evolving/dynamic 6LoWPAN,

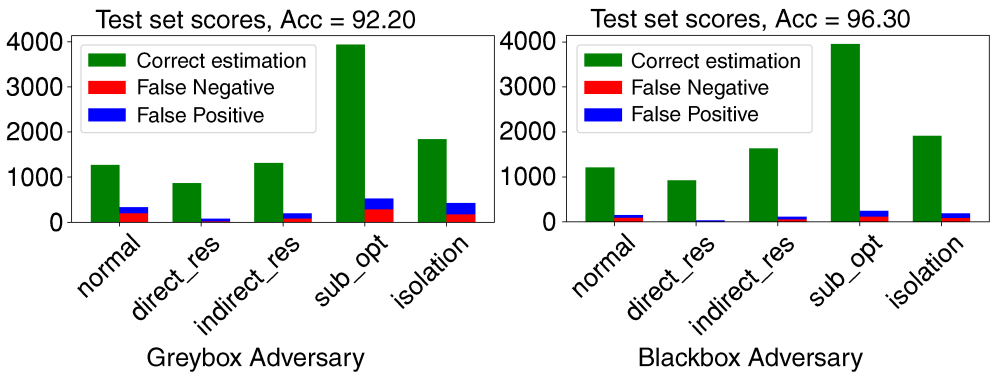


Figure 5.13: *KNNADWIN against adversarial ML-based attacks (Y-axis shows number of instances)*

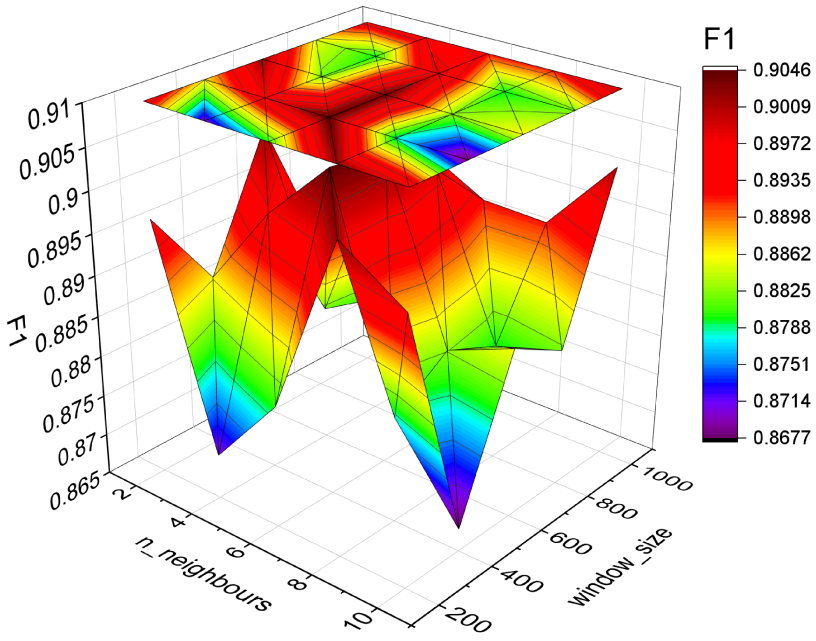


Figure 5.14: *KNNADWIN against a grey-box adversary.*

where 40% $\sim$ 80% of nodes are mobile and nodes transfer different application packets over the time. Fig. 5.16, illustrates the outcomes of this experiment where ML classifiers perform the binary classification task. Contrary to our expectations, batch-trained SVM and KNN were more generalised to outperform XGBoost and Random Forest. In Fig. 5.15, we evaluate the performance of standard ML classifiers' performance along with SMOTE against the grey-box adversary.

5.6 Summary

The broad connectivity of LLN nodes has exposed the 6LoWPAN to various routing threats (discussed in Chapter 2). Furthermore, the 6LoWPAN has an evolving and imbalanced data

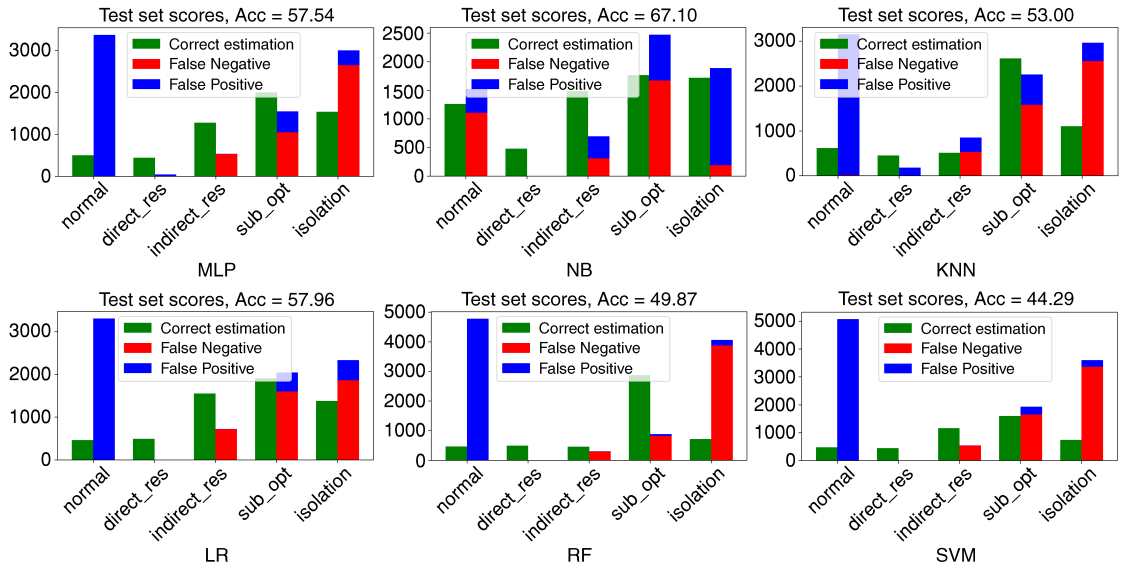


Figure 5.15: *Standard ML classifiers (using SMOTE) against a grey-box adversary.*

environment; hence, any security mechanism needs to be robust, efficient, and adaptive to perform its primary task in this network accurately. The proposed adversarial reinforcement learning plays a vital role in generating robust and generalised incremental ML-based IDSs for 6LoWPAN. The outcome of our experiments show that the proposed scheme can enhance the performance of an IDS agent against various RPL attacks. The proposed scheme can detect and also adopt changes in computer network traffic. Our experimental outcomes confirm that the proposed RL-based adversarial environment can efficiently train incremental ML-based IDSs over more challenging observations. This approach leads to the generation of more robust, generalised, and resource-efficient IDS. In this chapter, we measure the performance of our proposed scheme against intelligent ML-based RPL attacks. Our experiment outcomes confirm that the developed IDS can identify adversarial ML-based RPL attacks that are hard to identify by any standard ML classifier.

On the other hand, the adversary may also initiate a white-box attack against 6LoWPAN, where the adversary is assumed to have the complete knowledge about the IDS configurations and also knows what training data is used by the IDS classifier to develop its detection engine. Using this knowledge, the adversary can create a clone of the IDS in its system to find the IDS's vulnerabilities before initiating any intrusion. Since, identifying a white-box adversary is more challenging, in this dissertation the protection against the white-box attack has not been considered and is left as potential future work.

Moreover, it is not clear that up to what extent the proposed adversarial reinforcement learning environment can enhance the performance of other incremental ML classifiers in identifying RPL attacks. In future, we will evaluate the performance of our proposed scheme using hoeffding tree and adaptive random forest algorithms.

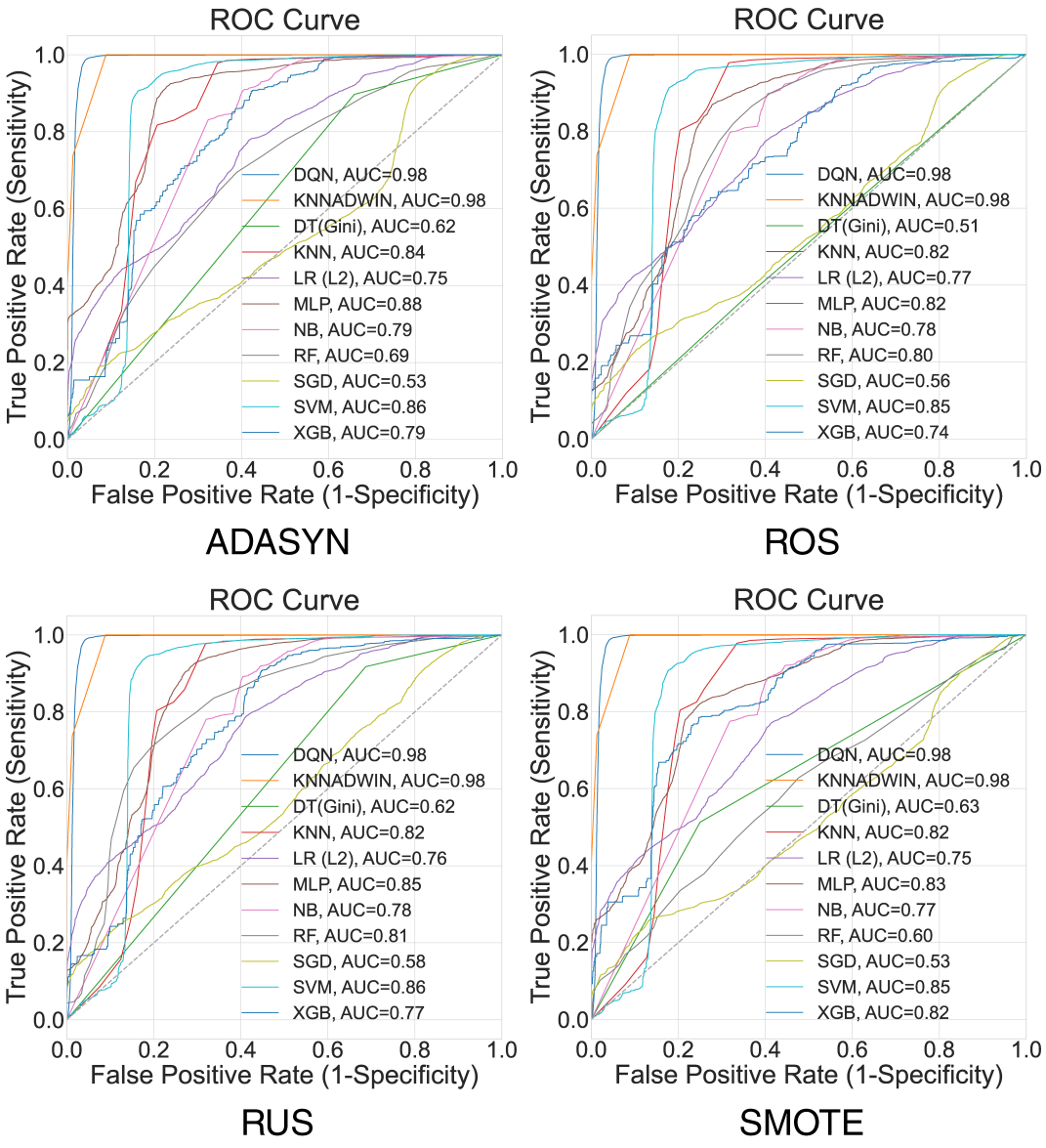


Figure 5.16: *Bench-marking the proposed scheme with standard ML classifiers (binary classification) in a dynamic environment.*

Chapter 6

Conclusions and future work

In this chapter, we summarise this dissertation and provide conclusions drawn from it. Remaining issues and potential future work are discussed.

6.1 Context

The IPv6 over low-power wireless personal area networks (6LoWPAN) standard enables resource-constrained devices to connect and interact over the Internet. Because of the seamless connectivity and significant computational constraints of Low power and Lossy Network (LLN) nodes, a new routing protocol called the Routing Protocol for low power Lossy networks (RPL) has been proposed to associate routes between the LLN nodes and the IPv6 Border Router (6BR). In this context, the routing relies on the construction of the suitable Destination Oriented Directed Acyclic Graphs (DODAGs) using node rank values to structure the graphs. Although, the ranking system enables various properties such as route discovery, loop prevention, and overhead management, it is vulnerable to several attacks (as discussed in Chapter 2) which may significantly degrade resource utilisation, routing mechanisms and general network performance. Therefore, protecting against any attacks on RPL is vital; however, the computational limitations of the LLN nodes are a barrier to the adoption of highly promising leading-edge approaches such as those based on machine learning (ML). Moreover, 6LoWPAN has an evolving data environment, where node movement and inaccessibility alter the data distribution unpredictably, and so the IDS needs to update its model incrementally or retrain it using recently observed batches of data. Therefore, IDS must employ intelligent algorithms to monitor the evolvement in the network environment and update itself effectively. In this context, even though adaptivity plays a major role in the robustness of IDS in 6LoWPAN, extant articles have not considered it.

This dissertation has sought to determine the best possible approaches in the development of a robust and generalised IDS for 6LoWPAN. We aimed to develop an adaptive IDS that can secure 6LoWPAN against various types of RPL attacks (including overlooked and less researched ones) and robustly adjust to the evolving data environments and learn new types of intrusion. In Chapter 3, we proposed an adaptive heterogeneous hybrid IDS (which we termed the central IDS) that is placed on the 6BR to secure the 6LoWPAN against internal and external intrusions. Since the proposed Central IDS (CIDS) on the 6BR could not observe network communications of distant nodes, the proposed scheme employs Anomaly-

based NIDS (ANIDS) agents to passively monitor control packets of their neighbours. The developed ANIDSs are responsible for reporting suspicious activities to the CIDS for further analysis. In this context, the obtained results (as shown in Section 3.3.2) support the "*Hypothesis 1: An adaptive heterogeneous hybrid IDS can be able to identify various internal and external RPL attacks in 6LoWPAN. The adaptivity through concept-drift detection should/will enable an incremental ML-based IDS to enhance its intrusion detection performance over time by adapting to unforeseen intrusions and data distributions*". To assess the detection performance of the proposed scheme, we conducted *six* experiments. The outcomes of our experiments (presented in Section 3.3.2, of Chapter 3) show that the proposed CIDS can accurately identify RPL attacks and adjust to concept drifts (changes in data distribution). However, in our proposed scheme the ANIDS agents operate an off-line model, hence they cannot maintain accuracy against evolving network environments.

Therefore, in Chapter 4, we proposed an reinforcement learning-based IDS for 6LoWPAN to analyse network communications in a large-scaled network (with up to 128 LLN nodes) using light-weight ML-IDS agents. The proposed RL-based IDS seeks to identify the strength of each neighbouring ML-IDS in classifying different types of RPL attacks. To reduce computational complexity of ML classifiers, each ML-IDS trains over a small proportion of training dataset using salient features that they select through the chi-square feature selection approach. In Chapter 4, Section 4.4, the outcomes of our experiments support "*Hypothesis 2: An RL-based IDS framework should/will be able to enhance the strength of distributed ML-based IDS in detecting RPL intrusions. An RL-based IDS will be able to accurately identify suspicious activities with the help of ML-based IDS agents*".

In Chapters 3 and 4, we proposed batch-trained anomaly-based and signature-based "IDS agents" to detect intrusions in 6LoWPAN; therefore, the IDS agents will not be able to autonomously adapt to any new intrusions or concept-drifts. Hence, in Chapter 5, we propose an adversarial reinforcement learning scheme to develop efficient IDS agents with an excellent performance against targeted RPL attacks. In this regard, we proposed an adversarial RL-based environment to develop resource-efficient and generalised incremental ML-based IDS agents. The outcomes of our experiments (as shown in Section 5.5.2) show the significant enhancement in the intrusion detection performance of the IDS developed through our proposed scheme compared with standard ML classifiers. We further evaluate the performance of the proposed IDS against adversarial ML-based RPL attacks, where the adversary is using RL to identify the weakness of the IDS. The outcomes of our experiments, in Section 5.5.2, support "*Hypothesis 3: A robust adversarial RL-based IDS framework will be able to generate efficient detectors using imbalanced training data. The integration of adversarial RL environment and incremental machine-learning should be able/will facilitate the formation of resource-efficient, generalised and robust IDS detectors*".

In Section 2.7.8, we stated that 86% of articles in the literature used a simulator to evaluate the detection performance of their proposed model in 6LoWPAN. In this dissertation, we employ a network simulator to evaluate the performance of our proposed scheme against RPL attacks in various scaled networks. Network simulation provides distinct advantages over a full-scale, real-world deployment: namely, fast implementation, more flexibility concerning network layout and communication parameters, and the capability of running network scenarios under identical conditions. The emulation capability of the adopted simulator enabled this dissertation to connect real hardware to the simulation environment and obtain more

realistic simulation outcomes. Moreover, we employed real micro-controller to measure the energy exhaustion of our proposed schemes.

6.2 Summary of Contributions

The major contributions of this thesis are outlined as follow:

- In Chapter 3, we proposed an adaptive heterogeneous hybrid ML-IDS, which can maintain its effectiveness against environmental change in different scaled 6LoWPAN. We constructed a set of features that can facilitate the identification of RPL attacks. Our experimental outcomes show that the proposed scheme can ensure 90∼100% accuracy in detecting RPL attacks (shown in table 3.4); However, the developed ANIDS (the first detection layer) provides 89.39% recall and 20.25% FPR (discussed in Section 3.3.2).
- In Chapter 4, we proposed an RL-IDS framework to enhance the strength of the distributed ML-IDS in detecting internal and external RPL intrusions. The proposed scheme is the first application of reinforcement learning for IDS in 6LoWPAN. Although the developed ML detectors are batch-trained ML classifiers and cannot detect and adapt to concept drifts, the proposed scheme can identify RPL attacks accurately (shown in Table 4.6).
- In Chapter 5, we proposed a robust adversarial RL-based IDS framework that can generate resource-efficient adjustable detectors using imbalanced training data. We presented the first application of reinforcement learning and incremental machine learning for IDS in 6LoWPAN. The proposed IDS is capable of detecting intelligent *ML-based combinational intrusions* against 6LoWPAN. The outcomes of our experiments (discussed in Section 5.5.2) show that the proposed scheme can develop IDSs to identify various RPL attacks accurately and efficiently in evolving environment of 6LoWPAN.
- It should be noted that, in Chapters 3, 4, and 5, we proposed effective IDSs that are capable of identifying and distinguishing a wide range of RPL attacks, including less researched ones; In this dissertation, we develop IDSs which are capable of detecting increase rank, DIO suppression, and replay attacks for the first time. The outcomes of our experiments show that the proposed scheme can accurately identify known and previously unseen RPL intrusions.

6.3 Extensions and Future Work

In this section, we explicitly suggest four potential future research ideas:

6.3.1 Development of a comprehensive, collaborative IDS

In Chapter 2, Fig. 2.14.A and Table 2.6 show that the majority of proposed methods employ active hybrid monitoring techniques. However, the conducted researches consider only one DODAG with a single border router in their scenarios. Securing 6LoWPAN against sophisticated intrusions (e.g. cooperative attacks) requires the development of a distributed

collaborative IDS to monitor several LLNs from a global perspective, with different LLNs informing each other of newly discovered intrusions.

6.3.2 Host-based and Network-based IDS development

We did not find any combination of HIDS and NIDS to secure IoT against both application layer and network layer attacks. IoT faces significant attacks from both levels and so fusing the best aspect of HIDS and NIDS seems essential.

6.3.3 Improve validation strategies

Validating a detection technique in order to design effective security measures for IoT networks and, more specifically, for RPL-based networks, requires realistic traces. The main two approaches to generate traces are simulation and testbeds. Several simulation tools available in this domain, some open-source and others with paid licenses, are compared in Table 2.10. Modeling the real world IoT-RPL environment requires a sophisitcated simulation tool. Having the right RPL behaviour will enable the researchers to simulate the aforementioned attacks and evaluate practical detection and mitigation techniques.

A physical testbed provides another validation means. However, researchers generally use very small-scale collections of devices which cannot mimic the actual IoT networks running RPL as the routing protocol. As indicated in Section 2.7.7, the average number of nodes of the testbeds implemented by researchers was 49. In this dissertation, we evaluate our proposed scheme in LLNs with up to 128 nodes. A large-scale testbed of, say, a smart city, that includes a large number of IoT devices would be a major resource.

6.3.4 White-box adversarial ML-based attack

In Chapter 5, we developed an IDS capable of identifying adversarial black-box and grey-box ML-based attacks; however, we did not evaluate our proposed scheme against a white-box attack where the adversary has complete knowledge about the IDS configurations and knows what training data is used by IDS classifier to develop its detection engine. Using this knowledge, the adversary can create a clone of the IDS in its system to find the IDS's vulnerabilities before initiating any intrusion. Hence, identifying a white-box adversary is more challenging and further research is needed to do so effectively.

Bibliography

[1] R. Alexander, A. Brandt, J. Vasseur, J. Huii, K. Pister, P. Thubert, P. Levis, R. Struik, R. Kelsey, and T. Winter, "RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks," RFC 6550, Mar. 2012. [Online]. Available: https://rfc-editor.org/rfc/rfc6550.txt

[2] D. Barthel, J. Vasseur, K. Pister, M. Kim, and N. Dejean, "Routing Metrics Used for Path Calculation in Low-Power and Lossy Networks," RFC 6551, Mar. 2012. [Online]. Available: https://rfc-editor.org/rfc/rfc6551.txt

[3] A. M. Pasikhani, J. A. Clark, P. Gope, and A. Alshahrani, "Intrusion detection systems in rpl-based 6lowpan: A systematic literature review," *IEEE Sensors Journal*, 2021.

[4] F. Hussain, R. Hussain, S. A. Hassan, and E. Hossain, "Machine learning in iot security: Current solutions and future challenges," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 3, pp. 1686–1721, 2020.

[5] G. Simoglou, G. Violettas, S. Petridou, and L. Mamatas, "Intrusion detection systems for rpl security: A comparative analysis," *Computers & Security*, p. 102219, 2021.

[6] L. Horwitz. The future of iot miniguide: The burgeoning iot market continues. Accessed: 20.03.2021. [Online]. Available: https://www.cisco.com/c/en/us/solutions/internet-of-things/future-of-iot.html

[7] T. Park, N. Abuzainab, and W. Saad, "Learning how to communicate in the internet of things: Finite resources and heterogeneity," 2016.

[8] D. Midi, A. Rullo, A. Mudgerikar, and E. Bertino, "Kalis — a system for knowledge-driven adaptable intrusion detection for the internet of things," in *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*, 2017, pp. 656–666.

[9] E. Fernandes, A. Rahmati, K. Eykholt, and A. Prakash, "Internet of things security research: A rehash of old ideas or new intellectual challenges?" *IEEE Security Privacy*, vol. 15, no. 4, pp. 79–84, 2017.

[10] B. B. Zarpelão, R. S. Miani, C. T. Kawakani, and S. C. de Alvarenga, "A survey of intrusion detection in internet of things," *Journal of Network and Computer Applications*, vol. 84, pp. 25–37, 2017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1084804517300802

[11] P. Pongle and G. Chavan, "A survey: Attacks on rpl and 6lowpan in iot," in *2015 International Conference on Pervasive Computing (ICPC)*, 2015, pp. 1–6.

[12] D. Airehrour, J. Gutierrez, and S. K. Ray, "Secure routing for internet of things: A survey," *Journal of Network and Computer Applications*, vol. 66, pp. 198–213, 2016. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1084804516300133

[13] A. Raoof, A. Matrawy, and C. Lung, "Routing attacks and mitigation methods for rpl-based internet of things," *IEEE Communications Surveys Tutorials*, vol. 21, no. 2, pp. 1582–1606, 2019.

[14] L. Wallgren, S. Raza, and T. Voigt, "Routing attacks and countermeasures in the rpl-based internet of things," *International Journal of Distributed Sensor Networks*, vol. 9, no. 8, p. 794326, 2013. [Online]. Available: https://doi.org/10.1155/2013/794326

[15] A. Mayzaud, R. Badonnel, and I. Chrisment, "A taxonomy of attacks in rpl-based internet of things," *International Journal of Network Security*, vol. 18, no. 3, pp. 459–473, 2016.

[16] F. Hussain, R. Hussain, S. A. Hassan, and E. Hossain, "Machine learning in iot security: Current solutions and future challenges," *IEEE Communications Surveys Tutorials*, vol. 22, no. 3, pp. 1686–1721, 2020.

[17] A. Verma and V. Ranga, "Security of rpl based 6lowpan networks in the internet of things: A review," *IEEE Sensors Journal*, vol. 20, no. 11, pp. 5666–5690, 2020.

[18] H. Kim, J. Ko, D. E. Culler, and J. Paek, "Challenging the ipv6 routing protocol for low-power and lossy networks (rpl): A survey," *IEEE Communications Surveys Tutorials*, vol. 19, no. 4, pp. 2502–2525, 2017.

[19] A. Mosenia and N. K. Jha, "A comprehensive study of security of internet-of-things," *IEEE Transactions on Emerging Topics in Computing*, vol. 5, no. 4, pp. 586–602, 2017.

[20] O. Vermesan and P. Friess, *Internet of things: converging technologies for smart environments and integrated ecosystems*. River publishers, 2013.

[21] I. E. Korbi, M. Ben Brahim, C. Adjih, and L. A. Saidane, "Mobility enhanced rpl for wireless sensor networks," in *2012 Third International Conference on The Network of the Future (NOF)*, 2012, pp. 1–8.

[22] S. Raza, L. Wallgren, and T. Voigt, "Svelte: Real-time intrusion detection in the internet of things," *Ad Hoc Networks*, vol. 11, no. 8, pp. 2661–2674, 2013. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1570870513001005

[23] A. Aijaz, H. Su, and A.-H. Aghvami, "Corpl: A routing protocol for cognitive radio enabled ami networks," *IEEE Transactions on Smart Grid*, vol. 6, no. 1, pp. 477–485, 2014.

[24] S. Basagni, C. Petrioli, R. Petroccia, and D. Spaccini, "Carp: A channel-aware routing protocol for underwater acoustic wireless networks," *Ad Hoc Networks*, vol. 34, pp. 92–104, 2015.

[25] A. Le, J. Loo, Y. Luo, and A. Lasebae, "Specification-based ids for securing rpl from topology attacks," in *2011 IFIP Wireless Days (WD)*, 2011, pp. 1–3.

[26] T. Winter, P. Thubert, A. Brandt, J. W. Hui, R. Kelsey, P. Levis, K. Pister, R. Struik, J.-P. Vasseur, R. K. Alexander *et al.*, "Rpl: Ipv6 routing protocol for low-power and lossy networks." *rfc*, vol. 6550, pp. 1–157, 2012.

[27] J. Vasseur, M. Kim, K. Pister, N. Dejean, and D. Barthel, "Routing metrics used for path calculation in low-power and lossy networks," in *RFC 6551*. IETF, 2012, pp. 1–30.

[28] J. Martocci, P. De Mil, N. Riou, and W. Vermeylen, "Building automation routing requirements in low-power and lossy networks," in *Internet Engineering Task Force (IETF)*, 2010, vol. RFC5867.

[29] A. Brandt, J. Buron, G. Porcu, and T. Italia, "Home automation routing requirements in low-power and lossy networks," 2010.

[30] M. Dohler, T. Watteyne, T. Winter, and D. Barthel, "Routing requirements for urban low-power and lossy networks," 2009.

[31] K. Pister, P. Thubert, C. Systems, S. Dwars, and T. Phinney, "Industrial routing requirements in low-power and lossy networks," 2009.

[32] A. Khosla and T. C. Aseri, "Comparative analysis of objective functions in routing protocol for low power and lossy networks."

[33] J. V. V. Sobral, J. J. P. C. Rodrigues, R. A. L. Rabêlo, J. Al-Muhtadi, and V. Korotaev, "Routing protocols for low power and lossy networks in internet of things applications," *Sensors*, vol. 19, no. 9, 2019. [Online]. Available: https://www.mdpi.com/1424-8220/19/9/2144

[34] A. Musaddiq, Y. B. Zikria, S. W. Kim *et al.*, "Routing protocol for low-power and lossy networks for heterogeneous traffic network," *EURASIP Journal on Wireless Communications and Networking*, vol. 2020, no. 1, pp. 1–23, 2020.

[35] N. Pradeska, Widyawan, W. Najib, and S. S. Kusumawardani, "Performance analysis of objective function mrhof and of0 in routing protocol rpl ipv6 over low power wireless personal area networks (6lowpan)," in *2016 8th International Conference on Information Technology and Electrical Engineering (ICITEE)*, 2016, pp. 1–6.

[36] P. Thubert *et al.*, "Objective function zero for the routing protocol for low-power and lossy networks (rpl)," 2012.

[37] O. Gnawali and P. Levis, "The minimum rank with hysteresis objective function," *RFC 6719*, 2012.

[38] ——, "The etx objective function for rpl," *draft-gnawali-roll-etxof-01*, 2010.

[39] A. Brachman, "Rpl objective function impact on llns topology and performance," in *Internet of Things, Smart Spaces, and Next Generation Networking*, S. Balandin, S. Andreev, and Y. Koucheryavy, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 340–351.

[40] X. Liu, Z. Sheng, C. Yin, F. Ali, and D. Roggen, "Performance analysis of routing protocol for low power and lossy networks (rpl) in large scale networks," *IEEE Internet of Things Journal*, vol. 4, no. 6, pp. 2172–2185, 2017.

[41] A. Le, J. Loo, A. Lasebae, A. Vinel, Y. Chen, and M. Chai, "The impact of rank attack on network topology of routing protocol for low-power and lossy networks," *IEEE Sensors Journal*, vol. 13, no. 10, pp. 3685–3692, 2013.

[42] A. Aris, S. F. Oktug, and S. Berna Ors Yalcin, "Rpl version number attacks: In-depth study," in *NOMS 2016 - 2016 IEEE/IFIP Network Operations and Management Symposium*, 2016, pp. 776–779.

[43] A. Mayzaud, A. Sehgal, R. Badonnel, I. Chrisment, and J. Schönwälder, "A study of rpl dodag version attacks," in *Monitoring and Securing Virtualized Networks and Services*, A. Sperotto, G. Doyen, S. Latré, M. Charalambides, and B. Stiller, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014, pp. 92–104.

[44] P. Perazzo, C. Vallati, D. Varano, G. Anastasi, and G. Dini, "Implementation of a wormhole attack against a rpl network: Challenges and effects," in *2018 14th Annual Conference on Wireless On-demand Network Systems and Services (WONS)*, 2018, pp. 95–102.

[45] N. Song, L. Qian, and X. Li, "Wormhole attacks detection in wireless ad hoc networks: a statistical analysis approach," in *19th IEEE International Parallel and Distributed Processing Symposium*, 2005, pp. 8 pp.–.

[46] F. Medjek, D. Tandjaoui, M. R. Abdmeziem, and N. Djedjig, "Analytical evaluation of the impacts of sybil attacks against rpl under mobility," in *2015 12th International Symposium on Programming and Systems (ISPS)*, 2015, pp. 1–9.

[47] K. Zhang, X. Liang, R. Lu, and X. Shen, "Sybil attacks and their defenses in the internet of things," *IEEE Internet of Things Journal*, vol. 1, no. 5, pp. 372–383, 2014.

[48] S. Murali and A. Jamalipour, "A lightweight intrusion detection for sybil attack under mobile rpl in the internet of things," *IEEE Internet of Things Journal*, vol. 7, no. 1, pp. 379–388, 2020.

[49] A. Le, J. Loo, K. K. Chai, and M. Aiash, "A specification-based ids for detecting attacks on rpl-based network topology," *Information*, vol. 7, no. 2, 2016. [Online]. Available: https://www.mdpi.com/2078-2489/7/2/25

[50] P. Perazzo, C. Vallati, G. Anastasi, and G. Dini, "Dio suppression attack against routing in the internet of things," *IEEE Communications Letters*, vol. 21, no. 11, pp. 2524–2527, 2017.

[51] A. Verma and V. Ranga, "Cosec-rpl: detection of copycat attacks in rpl based 6lowpans using outlier analysis," *Telecommunication Systems*, vol. 75, pp. 43–61, 2020.

[52] C. Pu, "Mitigating dao inconsistency attack in rpl-based low power and lossy networks," in *2018 IEEE 8th Annual Computing and Communication Workshop and Conference (CCWC)*, 2018, pp. 570–574.

[53] C. Cervantes, D. Poplade, M. Nogueira, and A. Santos, "Detection of sinkhole attacks for supporting secure routing on 6lowpan for internet of things," in *2015 IFIP/IEEE International Symposium on Integrated Network Management (IM)*, 2015, pp. 606–611.

[54] H. Bostani and M. Sheikhan, "Hybrid of anomaly-based and specification-based ids for internet of things using unsupervised opf based on mapreduce approach," *Computer Communications*, vol. 98, pp. 52–71, 2017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0140366416306387

[55] M. Surendar and A. Umamakeswari, "Indres: An intrusion detection and response system for internet of things with 6lowpan," in *2016 International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET)*, 2016, pp. 1903–1908.

[56] M. N. Napiah, M. Y. I. Bin Idris, R. Ramli, and I. Ahmedy, "Compression header analyzer intrusion detection system (cha - ids) for 6lowpan communication protocol," *IEEE Access*, vol. 6, pp. 16 623–16 638, 2018.

[57] E. Kfoury, J. Saab, P. Younes, and R. Achkar, "A self organizing map intrusion detection system for rpl protocol attacks," *International Journal of Interdisciplinary Telecommunications and Networking (IJITN)*, vol. 11, no. 1, pp. 30–43, 2019.

[58] S. Choudhary and N. Kesswani, "Detection and prevention of routing attacks in internet of things," in *2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/ 12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)*, 2018, pp. 1537–1540.

[59] A. Verma and V. Ranga, "Elnids: Ensemble learning based network intrusion detection system for rpl based internet of things," in *2019 4th International Conference on Internet of Things: Smart Innovation and Usages (IoT-SIU)*, 2019, pp. 1–6.

[60] S. Choudhary and N. Kesswani, "Cluster-based intrusion detection method for internet of things," in *2019 IEEE/ACS 16th International Conference on Computer Systems and Applications (AICCSA)*, 2019, pp. 1–8.

[61] A. Althubaity, H. Ji, T. Gong, M. Nixon, R. Ammar, and S. Han, "Arm: A hybrid specification-based intrusion detection system for rank attacks in 6tisch networks," in *2017 22nd IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, 2017, pp. 1–8.

[62] J. Foley, N. Moradpoor, and H. Ochen, "Employing a machine learning approach to detect combined internet of things attacks against two objective functions using a novel dataset," *Security and Communication Networks*, vol. 2020, 2020.

[63] M. Sheikhan and H. Bostani, "A security mechanism for detecting intrusions in internet of things using selected features based on mi-bgsa," *International Journal of Information & Communication Technology Research*, vol. 9, no. 2, pp. 53–62, 2017.

[64] S. M. H. Mirshahjafari and B. S. Ghahfarokhi, "Sinkhole+cloneid: A hybrid attack on rpl performance and detection method," *Information Security Journal: A Global Perspective*, vol. 28, no. 4-5, pp. 107–119, 2019. [Online]. Available: https://doi.org/10.1080/19393555.2019.1658829

[65] U. Shafique, A. Khan, A. Rehman, F. Bashir, and M. Alam, "Detection of rank attack in routing protocol for low power and lossy networks," *Annals of Telecommunications*, vol. 73, no. 7, pp. 429–438, 2018.

[66] R. Stephen and L. Arockiam, "E2v: Techniques for detecting and mitigating rank inconsistency attack (RInA) in RPL based internet of things," *Journal of Physics: Conference Series*, vol. 1142, p. 012009, nov 2018. [Online]. Available: https://doi.org/10.1088/1742-6596/1142/1/012009

[67] L. Zhang, G. Feng, and S. Qin, "Intrusion detection system for rpl from routing choice intrusion," in *2015 IEEE International Conference on Communication Workshop (ICCW)*, 2015, pp. 2652–2658.

[68] T. Matsunaga, K. Toyoda, and I. Sasase, "Low false alarm rate rpl network monitoring system by considering timing inconstancy between the rank measurements," in *2014 11th International Symposium on Wireless Communications Systems (ISWCS)*, 2014, pp. 427–431.

[69] D. Airehrour, J. Gutierrez, and S. K. Ray, "Securing rpl routing protocol from blackhole attacks using a trust-based mechanism," in *2016 26th International Telecommunication Networks and Applications Conference (ITNAC)*, 2016, pp. 115–120.

[70] A. Sehgal, A. Mayzaud, R. Badonnel, I. Chrisment, and J. Schönwälder, "Addressing dodag inconsistency attacks in rpl networks," in *2014 Global Information Infrastructure and Networking Symposium (GIIS)*, 2014, pp. 1–8.

[71] D. Airehrour, J. Gutierrez, and S. K. Ray, "A trust-aware rpl routing protocol to detect blackhole and selective forwarding attacks," p. 50–69, 2017. [Online]. Available: https://search.informit.org/doi/10.3316/informit.752286025338502

[72] H. B. Patel and D. C. Jinwala, "Blackhole detection in 6lowpan based internet of things: An anomaly based approach," in *TENCON 2019 - 2019 IEEE Region 10 Conference (TENCON)*, 2019, pp. 947–954.

[73] A. Amouri, V. T. Alaparthy, and S. D. Morgera, "Cross layer-based intrusion detection based on network behavior for iot," in *2018 IEEE 19th Wireless and Microwave Technology Conference (WAMICON)*, 2018, pp. 1–4.

[74] E. G. Ribera, B. Martinez Alvarez, C. Samuel, P. P. Ioulianou, and V. G. Vassilakis, "Heartbeat-based detection of blackhole and greyhole attacks in rpl networks," in *2020*

12th International Symposium on Communication Systems, Networks and Digital Signal Processing (CSNDSP), 2020, pp. 1–6.

[75] S. Luangoudom, D. Tran, T. Nguyen, H. A. Tran, G. Nguyen, and Q. T. Ha, "svblock: mitigating black hole attack in low-power and lossy networks," *International Journal of Sensor Networks*, vol. 32, no. 2, pp. 77–86, 2020. [Online]. Available: https://www.inderscienceonline.com/doi/abs/10.1504/IJSNET.2020.104923

[76] F. Gara, L. Ben Saad, and R. Ben Ayed, "An intrusion detection system for selective forwarding attack in ipv6-based mobile wsns," in *2017 13th International Wireless Communications and Mobile Computing Conference (IWCMC)*, 2017, pp. 276–281.

[77] A. Nikam and D. Ambawade, "Opinion metric based intrusion detection mechanism for rpl protocol in iot," in *2018 3rd International Conference for Convergence in Technology (I2CT)*, 2018, pp. 1–6.

[78] C. Pu and S. Hajjar, "Mitigating forwarding misbehaviors in rpl-based low power and lossy networks," in *2018 15th IEEE Annual Consumer Communications Networking Conference (CCNC)*, 2018, pp. 1–6.

[79] P. Pongle and G. Chavan, "Real time intrusion and wormhole attack detection in internet of things," *International Journal of Computer Applications*, vol. 121, no. 9, 2015.

[80] S. Deshmukh-Bhosale and S. S. Sonavane, "A real-time intrusion detection system for wormhole attack in the rpl based internet of things," *Procedia Manufacturing*, vol. 32, pp. 840–847, 2019, 12th International Conference Interdisciplinarity in Engineering, INTER-ENG 2018, 4–5 October 2018, Tirgu Mures, Romania. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2351978919303282

[81] P. Shukla, "Ml-ids: A machine learning approach to detect wormhole attacks in internet of things," in *2017 Intelligent Systems Conference (IntelliSys)*, 2017, pp. 234–240.

[82] D. B. Gothawal and S. Nagaraj, "Intrusion detection for enhancing rpl security," *Procedia Computer Science*, vol. 165, pp. 565–572, 2019, 2nd International Conference on Recent Trends in Advanced Computing ICRTAC -DISRUP - TIV INNOVATION , 2019 November 11-12, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877050920300594

[83] P. Kaliyar, W. B. Jaballah, M. Conti, and C. Lal, "Lidl: Localization with early detection of sybil and wormhole attacks in iot networks," *Computers Security*, vol. 94, p. 101849, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S016740482030122X

[84] A. Mayzaud, R. Badonnel, and I. Chrisment, "A distributed monitoring strategy for detecting version number attacks in rpl-based networks," *IEEE Transactions on Network and Service Management*, vol. 14, no. 2, pp. 472–486, 2017.

[85] P. Ioulianou, V. Vasilakis, I. Moscholios, and M. Logothetis, "A signature-based intrusion detection system for the internet of things," in *Information and Communication Technology Form*, AUT, June 2018. [Online]. Available: http://eprints.whiterose.ac.uk/133312/

[86] A. Arış, S. B. Örs Yalçın, and S. F. Oktuğ, "New lightweight mitigation techniques for rpl version number attacks," *Ad Hoc Networks*, vol. 85, pp. 81–91, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1570870518307625

[87] E. Aydogan, S. Yilmaz, S. Sen, I. Butun, S. Forsström, and M. Gidlund, "A central intrusion detection system for rpl-based industrial internet of things," in *2019 15th IEEE International Workshop on Factory Communication Systems (WFCS)*, 2019, pp. 1–5.

[88] J. Cañedo and A. Skjellum, "Using machine learning to secure iot systems," in *2016 14th Annual Conference on Privacy, Security and Trust (PST)*, 2016, pp. 219–222.

[89] A. Verma and V. Ranga, "Mitigation of dis flooding attacks in rpl-based 6lowpan networks," *Transactions on Emerging Telecommunications Technologies*, vol. 31, no. 2, p. e3802, 2020, e3802 ett.3802. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/ett.3802

[90] B. Farzaneh, M. A. Montazeri, and S. Jamali, "An anomaly-based ids for detecting attacks in rpl-based internet of things," in *2019 5th International Conference on Web Research (ICWR)*, 2019, pp. 61–66.

[91] P. P. Ioulianou and V. G. Vassilakis, "Denial-of-service attacks and countermeasures in the rpl-based internet of things," in *Computer Security*, S. Katsikas, F. Cuppens, N. Cuppens, C. Lambrinoudakis, C. Kalloniatis, J. Mylopoulos, A. Antón, S. Gritzalis, F. Pallas, J. Pohle, A. Sasse, W. Meng, S. Furnell, and J. Garcia-Alfaro, Eds. Cham: Springer International Publishing, 2020, pp. 374–390.

[92] P. Kasinathan, C. Pastrone, M. A. Spirito, and M. Vinkovits, "Denial-of-service detection in 6lowpan based internet of things," in *2013 IEEE 9th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, 2013, pp. 600–607.

[93] S. O. Amin, M. S. Siddiqui, C. S. Hong, and S. Lee, "Rides: Robust intrusion detection system for ip-based ubiquitous sensor networks," *Sensors*, vol. 9, no. 5, pp. 3447–3468, 2009. [Online]. Available: https://www.mdpi.com/1424-8220/9/5/3447

[94] P. Kasinathan, G. Costamagna, H. Khaleel, C. Pastrone, and M. A. Spirito, "Demo: An ids framework for internet of things empowered by 6lowpan," in *Proceedings of the 2013 ACM SIGSAC Conference on Computer amp; Communications Security*, ser. CCS '13. New York, NY, USA: Association for Computing Machinery, 2013, p. 1337–1340. [Online]. Available: https://doi.org/10.1145/2508859.2512494

[95] V. Pandu, J. Mohan, and T. P. Kumar, "Network intrusion detection and prevention systems for attacks in iot systems," in *Countering Cyber Attacks and Preserving the Integrity and Availability of Critical Systems*. IGI Global, 2019, pp. 128–141.

[96] B. Farzaneh, M. Koosha, E. Boochanpour, and E. Alizadeh, "A new method for intrusion detection on rpl routing protocol using fuzzy logic," in *2020 6th International Conference on Web Research (ICWR)*, 2020, pp. 245–250.

[97] B. Ghaleb, A. Al-Dubai, E. Ekonomou, M. Qasem, I. Romdhani, and L. Mackenzie, "Addressing the dao insider attack in rpl's internet of things networks," *IEEE Communications Letters*, vol. 23, no. 1, pp. 68–71, 2019.

[98] M. C. Belavagi and B. Muniyal, "Multiple intrusion detection in rpl based networks." *International Journal of Electrical & Computer Engineering (2088-8708)*, vol. 10, no. 1, 2020.

[99] A. Mayzaud, A. Sehgal, R. Badonnel, I. Chrisment, and J. Schönwälder, "Using the rpl protocol for supporting passive monitoring in the internet of things," in *NOMS 2016 - 2016 IEEE/IFIP Network Operations and Management Symposium*, 2016, pp. 366–374.

[100] A. Mayzaud, A. Sehgal, R. Badonnel, I. Chrisment, and J. Schönwälder, "Mitigation of topological inconsistency attacks in rpl-based low-power lossy networks," *International Journal of Network Management*, vol. 25, no. 5, pp. 320–339, 2015. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/nem.1898

[101] C. Pu, "Sybil attack in rpl-based internet of things: Analysis and defenses," *IEEE Internet of Things Journal*, vol. 7, no. 6, pp. 4937–4949, 2020.

[102] D. Airehrour, J. A. Gutierrez, and S. K. Ray, "Sectrust-rpl: A secure trust-aware rpl routing protocol for internet of things," *Future Generation Computer Systems*, vol. 93, pp. 860–876, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167739X17306581

[103] F. Medjek, D. Tandjaoui, I. Romdhani, and N. Djedjig, "A trust-based intrusion detection system for mobile rpl based networks," in *2017 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)*, 2017, pp. 735–742.

[104] A. Verma and V. Ranga, "Addressing flooding attacks in ipv6-based low power and lossy networks," in *TENCON 2019 - 2019 IEEE Region 10 Conference (TENCON)*, 2019, pp. 552–557.

[105] A. Saeed, A. Ahmadinia, A. Javed, and H. Larijani, "Intelligent intrusion detection in low-power iots," *ACM Trans. Internet Technol.*, vol. 16, no. 4, Dec. 2016. [Online]. Available: https://doi.org/10.1145/2990499

[106] T. Tsao, R. Alexander, M. Dohler, V. Daza, A. Lozano, and M. Richardson, "A security threat analysis for the routing protocol for low-power and lossy networks (rpls)," *RFC 7416 (Informational), Internet Engineering Task Force*, 2015.

[107] O. Lounis and B. Malika, "A new vision for intrusion detection system in information systems," in *2015 Science and Information Conference (SAI)*, 2015, pp. 1352–1356.

[108] S. Sonavane, "Design and implementation of rssi based intrusion detection system for rpl based iot network," 2020.

[109] T. Jones, A. Dali, M. R. Rao, N. Biradar, J. Madassery, and K. Liu, "Towards a layered and secure internet-of-things testbed via hybrid mesh," in *2018 IEEE International Congress on Internet of Things (ICIOT)*, 2018, pp. 17–24.

[110] Z. A. Khan and P. Herrmann, "A trust based distributed intrusion detection mechanism for internet of things," in *2017 IEEE 31st International Conference on Advanced Information Networking and Applications (AINA)*, 2017, pp. 1169–1176.

[111] E. Viegas, A. Santin, L. Oliveira, A. França, R. Jasinski, and V. Pedroni, "A reliable and energy-efficient classifier combination scheme for intrusion detection in embedded systems," *Computers Security*, vol. 78, pp. 16–32, 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167404818306175

[112] H. Sedjelmaci, S. M. Senouci, and M. Al-Bahri, "A lightweight anomaly detection technique for low-resource iot devices: A game-theoretic methodology," in *2016 IEEE International Conference on Communications (ICC)*, 2016, pp. 1–6.

[113] J. Li, Z. Zhao, R. Li, and H. Zhang, "Ai-based two-stage intrusion detection for software defined iot networks," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 2093–2102, 2019.

[114] J. Arshad, "Colide: a collaborative intrusion detection framework for internet of things," *IET Networks*, vol. 8, pp. 3–14(11), January 2019. [Online]. Available: https://digital-library.theiet.org/content/journals/10.1049/iet-net.2018.5036

[115] N. Beigi Mohammadi, J. Mišić, V. B. Mišić, and H. Khazaei, "A framework for intrusion detection system in advanced metering infrastructure," *Security and Communication Networks*, vol. 7, no. 1, pp. 195–205, 2014. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/sec.690

[116] D. B. Gothawal and S. Nagaraj, "Anomaly-based intrusion detection system in rpl by applying stochastic and evolutionary game models over iot environment," *Wireless Personal Communications*, vol. 110, no. 3, pp. 1323–1344, 2020.

[117] D. Shreenivas, S. Raza, and T. Voigt, "Intrusion detection in the rpl-connected 6lowpan networks," in *Proceedings of the 3rd ACM International Workshop on IoT Privacy, Trust, and Security*, ser. IoTPTS '17. New York, NY, USA: Association for Computing Machinery, 2017, p. 31–38. [Online]. Available: https://doi.org/10.1145/3055245.3055252

[118] A. Rghioui, A. Khannous, and M. Bouhorma, "Monitoring behavior-based intrusion detection system for 6lowpan networks," *International Journal of Innovation and Applied Studies*, vol. 11, no. 4, p. 894, 2015.

[119] A. L. Buczak and E. Guven, "A survey of data mining and machine learning methods for cyber security intrusion detection," *IEEE Communications surveys & tutorials*, vol. 18, no. 2, pp. 1153–1176, 2015.

[120] M. H. Bhuyan, D. K. Bhattacharyya, and J. K. Kalita, "Survey on incremental approaches for network anomaly detection," *arXiv preprint arXiv:1211.4493*, 2012.

[121] J. Gama, I. Žliobaitė, A. Bifet, M. Pechenizkiy, and A. Bouchachia, "A survey on concept drift adaptation," *ACM computing surveys (CSUR)*, vol. 46, no. 4, pp. 1–37, 2014.

[122] J. Gama, P. Medas, G. Castillo, and P. Rodrigues, "Learning with drift detection," in *Brazilian symposium on artificial intelligence*. Springer, 2004, pp. 286–295.

[123] R. S. Sutton, A. G. Barto *et al.*, *Introduction to reinforcement learning*. MIT press Cambridge, 1998, vol. 135.

[124] A. Mitrokotsa and A. Karygiannis, "Intrusion detection techniques in sensor networks," *Wireless Sensor Network Security*, vol. 1, no. 1, pp. 251–272, 2008.

[125] V. Verendel, "Quantified security is a weak hypothesis: A critical survey of results and assumptions," in *Proceedings of the 2009 Workshop on New Security Paradigms Workshop*, ser. NSPW '09. New York, NY, USA: Association for Computing Machinery, 2009, p. 37–50. [Online]. Available: https://doi.org/10.1145/1719030.1719036

[126] G. Kumar, "Evaluation metrics for intrusion detection systems-a study," *Evaluation*, vol. 2, no. 11, pp. 11–7, 2014.

[127] M. H. Bhuyan, D. K. Bhattacharyya, and J. K. Kalita, "Network anomaly detection: methods, systems and tools," *Ieee communications surveys & tutorials*, vol. 16, no. 1, pp. 303–336, 2013.

[128] H. M. Gomes, J. P. Barddal, F. Enembreck, and A. Bifet, "A survey on ensemble learning for data stream classification," *ACM Computing Surveys (CSUR)*, vol. 50, no. 2, pp. 1–36, 2017.

[129] J. Maerien, P. Agten, C. Huygens, and W. Joosen, "Famos: A flexible active monitoring service for wireless sensor networks," in *Distributed Applications and Interoperable Systems*, K. M. Göschka and S. Haridi, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 104–117.

[130] L. Ben Saad, C. Chauvenet, and B. Tourancheau, "Simulation of the RPL Routing Protocol for IPv6 Sensor Networks: two cases studies," in *International Conference on Sensor Technologies and Applications SENSORCOMM 2011*. Nice, France: IARIA, Sep. 2011. [Online]. Available: https://hal.inria.fr/hal-00647869

[131] G. I. Webb, R. Hyde, H. Cao, H. L. Nguyen, and F. Petitjean, "Characterizing concept drift," *Data Mining and Knowledge Discovery*, vol. 30, no. 4, pp. 964–994, 2016.

[132] P. Kaliyar, W. B. Jaballah, M. Conti, and C. Lal, "Lidl: Localization with early detection of sybil and wormhole attacks in iot networks," *Computers & Security*, vol. 94, p. 101849, 2020.

[133] M. A. Kareem and S. Tayeb, "Ml-based nids to secure rpl from routing attacks," in *2021 IEEE 11th Annual Computing and Communication Workshop and Conference (CCWC)*. IEEE, 2021, pp. 1000–1006.

[134] N. Martindale, M. Ismail, and D. A. Talbert, "Ensemble-based online machine learning algorithms for network intrusion detection systems using streaming data," *Information*, vol. 11, no. 6, p. 315, 2020.

[135] X. Yuan, R. Wang, Y. Zhuang, K. Zhu, and J. Hao, "A concept drift based ensemble incremental learning approach for intrusion detection," in *2018 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)*. IEEE, 2018, pp. 350–357.

[136] R. Singh, H. Kumar, and R. Singla, "An intrusion detection system using network traffic profiling and online sequential extreme learning machine," *Expert Systems with Applications*, vol. 42, no. 22, pp. 8609–8624, 2015.

[137] L. A. Maglaras and J. Jiang, "A real time ocsvm intrusion detection module with low overhead for scada systems," *International Journal of Advanced Research in Artificial Intelligence (IJARAI)*, vol. 3, no. 10, 2014.

[138] S. C. Tan, K. M. Ting, and T. F. Liu, "Fast anomaly detection for streaming data," in *Twenty-Second International Joint Conference on Artificial Intelligence*, 2011.

[139] B. Wang and J. Pineau, "Online bagging and boosting for imbalanced data streams," *IEEE Transactions on Knowledge and Data Engineering*, vol. 28, no. 12, pp. 3353–3366, 2016.

[140] N. C. Oza and S. Russell, "Experimental comparisons of online and batch versions of bagging and boosting," in *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, 2001, pp. 359–364.

[141] A. Bifet and R. Gavalda, "Adaptive learning from evolving data streams," in *International Symposium on Intelligent Data Analysis*. Springer, 2009, pp. 249–260.

[142] N. C. Oza and S. J. Russell, "Online bagging and boosting," in *International Workshop on Artificial Intelligence and Statistics*. PMLR, 2001, pp. 229–236.

[143] A. Bifet, G. Holmes, B. Pfahringer, R. Kirkby, and R. Gavalda, "New ensemble methods for evolving data streams," in *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2009, pp. 139–148.

[144] A. Bifet and R. Gavalda, "Learning from time-changing data with adaptive windowing," in *Proceedings of the 2007 SIAM international conference on data mining*. SIAM, 2007, pp. 443–448.

[145] R. Elwell and R. Polikar, "Incremental learning of concept drift in nonstationary environments," *IEEE Transactions on Neural Networks*, vol. 22, no. 10, pp. 1517–1531, 2011.

[146] H. Wang, W. Fan, P. S. Yu, and J. Han, "Mining concept-drifting data streams using ensemble classifiers," in *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, 2003, pp. 226–235.

[147] O. Gaddour and A. Koubâa, "Rpl in a nutshell: A survey," *Computer Networks*, vol. 56, no. 14, pp. 3163–3178, 2012.

[148] S. Otoum, B. Kantarci, and H. Mouftah, "Empowering reinforcement learning on big sensed data for intrusion detection," in *Icc 2019-2019 IEEE international conference on communications (ICC)*. IEEE, 2019, pp. 1–7.

[149] Y.-F. Hsu and M. Matsuoka, "A deep reinforcement learning approach for anomaly network intrusion detection system," in *2020 IEEE 9th International Conference on Cloud Networking (CloudNet)*. IEEE, 2020, pp. 1–6.

[150] M. Lopez-Martin, B. Carro, and A. Sanchez-Esguevillas, "Application of deep reinforcement learning to intrusion detection for supervised problems," *Expert Systems with Applications*, vol. 141, p. 112963, 2020.

[151] G. Caminero, M. Lopez-Martin, and B. Carro, "Adversarial environment reinforcement learning algorithm for intrusion detection," *Computer Networks*, vol. 159, pp. 96–109, 2019.

[152] K. Sethi, E. S. Rupesh, R. Kumar, P. Bera, and Y. V. Madhav, "A context-aware robust intrusion detection system: a reinforcement learning-based approach," *International Journal of Information Security*, vol. 19, no. 6, pp. 657–678, 2020.

[153] Z. S. Stefanova and K. M. Ramachandran, "Off-policy q-learning technique for intrusion response in network security," *World Academy of Science, Engineering and Technology, International Science Index*, vol. 136, pp. 262–268, 2018.

[154] S. Khalid, T. Khalil, and S. Nasreen, "A survey of feature selection and feature extraction techniques in machine learning," in *2014 science and information conference*. IEEE, 2014, pp. 372–378.

[155] X. Jin, A. Xu, R. Bie, and P. Guo, "Machine learning techniques and chi-square feature selection for cancer classification using sage gene expression profiles," in *International Workshop on Data Mining for Biomedical Applications*. Springer, 2006, pp. 106–115.

[156] S. Adam, L. Busoniu, and R. Babuska, "Experience replay for real-time reinforcement learning control," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 42, no. 2, pp. 201–212, 2011.

[157] H. Van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double q-learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 30, no. 1, 2016.

[158] K. Arulkumaran, M. P. Deisenroth, M. Brundage, and A. A. Bharath, "A brief survey of deep reinforcement learning," *arXiv preprint arXiv:1708.05866*, 2017.

[159] I. Osband, C. Blundell, A. Pritzel, and B. Van Roy, "Deep exploration via bootstrapped dqn," *arXiv preprint arXiv:1602.04621*, 2016.

[160] J. J. Davis and A. J. Clark, "Data preprocessing for anomaly based network intrusion detection: A review," *computers & security*, vol. 30, no. 6-7, pp. 353–375, 2011.

[161] V. Losing, B. Hammer, and H. Wersing, "Knn classifier with self adjusting memory for heterogeneous concept drift," in *2016 IEEE 16th international conference on data mining (ICDM)*. IEEE, 2016, pp. 291–300.

[162] R. Heartfield, G. Loukas, A. Bezemskij, and E. Panaousis, "Self-configurable cyber-physical intrusion detection for smart homes using reinforcement learning," *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 1720–1735, 2021.

[163] X. Ma and W. Shi, "Aesmote: Adversarial reinforcement learning with smote for anomaly detection," *IEEE Transactions on Network Science and Engineering*, vol. 8, no. 2, pp. 943–956, 2021.

[164] S. Yılmaz, E. Aydogan, and S. Sen, "A transfer learning approach for securing resource-constrained iot devices," *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 4405–4418, 2021.

[165] A. Mohammadi Pasikhani, A. J. Clark, and P. Gope, "Reinforcement-learning-based ids for 6lowpan," 2021.

[166] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602*, 2013.

[167] M. Delange, R. Aljundi, M. Masana, S. Parisot, X. Jia, A. Leonardis, G. Slabaugh, and T. Tuytelaars, "A continual learning survey: Defying forgetting in classification tasks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[168] B. C. Da Silva, E. W. Basso, A. L. Bazzan, and P. M. Engel, "Dealing with non-stationary environments using context detection," in *Proceedings of the 23rd international conference on Machine learning*, 2006, pp. 217–224.

[169] Y.-C. Lin, Z.-W. Hong, Y.-H. Liao, M.-L. Shih, M.-Y. Liu, and M. Sun, "Tactics of adversarial attack on deep reinforcement learning agents," *arXiv preprint arXiv:1703.06748*, 2017.

[170] M. A. Ayub, W. A. Johnson, D. A. Talbert, and A. Siraj, "Model evasion attack on intrusion detection systems using adversarial machine learning," in *2020 54th Annual Conference on Information Sciences and Systems (CISS)*. IEEE, 2020, pp. 1–6.

[171] I. G. Terrizzano, P. M. Schwarz, M. Roth, and J. E. Colino, "Data wrangling: The challenging yourney from the wild to the lake." in *CIDR*, 2015.

[172] Y. Sun, A. K. Wong, and M. S. Kamel, "Classification of imbalanced data: A review," *International journal of pattern recognition and artificial intelligence*, vol. 23, no. 04, pp. 687–719, 2009.