MULTI-OBJECTIVE APPROACH FOR INTRUSION DETECTION IN RPL-BASED INTERNET OF THINGS

RPL TABANLI NESNELERİN İNTERNETİNDE İZİNSİZ GİRİŞ TESPİTİ İÇİN ÇOK KRİTERLİ YAKLAŞIM

ALİ DEVECİ

PROF. DR. SEVİL ŞEN AKAGÜNDÜZ

Supervisor

ASST. PROF. DR. SELİM YILMAZ

2nd Supervisor

Submitted to

Graduate School of Science and Engineering of Hacettepe University

as a Partial Fulfillment to the Requirements

for the Award of the Degree of Master of Science

in Computer Engineering

January 2023

ABSTRACT

MULTI-OBJECTIVE APPROACH FOR INTRUSION DETECTION IN RPL-BASED INTERNET OF THINGS

Ali DEVECİ

Master of Science, Computer Engineering

Supervisor: Prof. Dr. Sevil ŞEN AKAGÜNDÜZ

2nd Supervisor: Asst. Prof. Dr. Selim YILMAZ

January 2023, 98 pages

In the last decade, humanity has witnessed a tremendous increase in the number of life-saving applications such as smart home or automated industrial systems. This is mainly due to recent advances in resource-constrained sensory devices that have low power and low computational capabilities, as well as the achievements in communication technology between such devices using the IPv6 protocol. Recent rapid integration has led to the development of a new paradigm known as the Internet of Things (IoT). The Internet of Things enables resource-constrained and heterogeneous devices to communicate and access information. This special IoT network called Low Power and Lossy Networks (LLN) enable this communication effectively. The low throughput and packet loss of LLNs characterize them as lossy links. Until now, many routing protocols have been proposed to ensure effective routing between the heterogenous devices in LLN. It is widely considered that the Routing Protocol for Low Power and Lossy Networks (RPL) are the most reliable routing protocols for LLNs, and therefore it has widely been adopted in a diverse range of IoT applications today. Proposed by the IETF-ROLL group in RFC-6550 documents, multipoint-to-point (MP2P) communications are ensured by RPL. Moreover, RPL enables

two different kind of communication called point-to-point (P2P) that enables communication one device to one device, and called point-to-multipoint (P2MP) that enables communication between one device to more than one devices (also called nodes), too.

Although effective and efficient routing can be found by RPL, it is very susceptible to malicious attacks that mainly stem from the intruders. This is because the security measurements specified in the protocol are not sufficient, and even they can easily be evaded by the attackers today. When considering the life-threatening consequences of insider attacks, it is of very high importance to develop reliable security solutions, which is the major reason researchers are working on it for a long while now.

Being an indispensable part of security systems, Intrusion Detection Systems (IDSs) have also been integrated into the LLNs operated by RPL. However, most of these solutions disregard the constrained nature of devices and the network, leading the IDS to be too costly, particularly in terms of memory and power consumption. This becomes even problematic as more and more nodes are in charge of the detection task. Therefore, in this thesis, we propose a centralized IDS in which a central node as well as collaborator nodes participate. In contrast to existing solutions, our objective is to make the our proposed IDS model lightweight in terms of battery and memory consumption so that not only effectiveness but also efficiency are guaranteed to secure LLN against four types of RPL attacks, including version number, hello flood, worst parent, and decreased rank.

This thesis employs Genetic Programming (GP), which is an evolutionary-based algorithm, as well as Non-Dominated Sorting Genetic Algorithm-II (NSGA-II), in order to simultaneously achieve the objectives with the lightweight IDS model generated (i.e., effectiveness and efficiency). Here, the performance of the proposed IDS model is extensively explored in a large number of network scenarios with varying topologies and mobility patterns. The results showed the applicability of GP to evolve a low-cost IDS model against various RPL-specific attacks.

Keywords: RPL, RPL Attacks, IoT, intrusion detecting, communication cost, genetic programming, multi-objective approach,

ÖZET

RPL TABANLI NESNELERİN İNTERNETİNDE İZİNSİZ GİRİŞ TESPİTİ İÇİN ÇOK KRİTERLİ YAKLAŞIM

Ali DEVECİ

Yüksek Lisans, Bilgisayar Mühendisliği

Danışman: Prof. Dr. Sevil ŞEN AKAGÜNDÜZ

Eş Danışman: Dr. Selim YILMAZ

Ocak 2023, 98 sayfa

Son yıllarda (son on yılda) insanlık, akıllı ev veya otomatik endüstriyel sistemler gibi hayat kurtaran uygulamaların sayısında muazzam bir artışa tanıklık etmiştir. Bunun başlıca nedeni, düşük güce ve düşük hesaplama yeteneklerine sahip kısıtlı kaynaklara sahip sensör cihazlardaki son gelişmeler ile IPv6 protokolünü kullanan bu tür cihazlar arasındaki iletişim teknolojisindeki başarılardır. Son hızlı entegrasyon, Nesnelerin İnterneti (IoT) olarak bilinen yeni bir paradigmanın gelişmesine yol açmıştır. Nesnelerin İnterneti, kaynakları kısıtlı ve heterojen cihazların iletişim kurmasını ve bilgilere erişmesini sağlar. Low Power and Lossy Networks (LLN) adı verilen bu özel IoT ağı, bu iletişimi etkin bir şekilde sağlar. LLN'lerin düşük çıktısı ve paket kaybı, onları kayıplı bağlantılar olarak nitelendirir. Şimdiye kadar, LLN'deki heterojen cihazlar arasında etkili yönlendirmeyi sağlamak için birçok yönlendirme protokolü önerilmiştir. Güç açısından düşük ve aynı zamanda kayıplı ağlarda kullanıma uygun olarak geliştirilen yönlendirme protokolünün (RPL), LLN'lerde kullanılan en güvenilir yönlendirme protokolleri olduğu yaygın olarak kabul edilmekte ve bu nedenle günümüzde çok çeşitli IoT uygulamalarında geniş çapta benimsenmektedir. IETF-ROLL grubu tarafından RFC-6550 belgelerinde önerilen, çok noktadan noktaya (MP2P) iletişim

RPL ile sağlanır. Ayrıca RPL, noktadan noktaya diye adlandırılan (P2P) yani bir cihazdan diğer bir cihaza iletişim sağlayan ve noktadan çok noktaya (P2MP) diye adlandırılan yani bir cihazdan birden fazla cihaza (veya düğümlere) iletişimi de sağlayabilmektedir.

Etkili ve verimli yönlendirme RPL tarafından sağlanabilse de, esas olarak ağ içindeki düğümlerden kaynaklanan kötü niyetli saldırılara karşı çok hassastır. Bunun nedeni, protokolde belirtilen güvenlik ölçümlerinin yeterli olmaması ve hatta günümüzde saldırganlar tarafından kolayca atlatılabilmesidir. İçeriden gelen saldırıların yaşamı tehdit eden sonuçları düşünüldüğünde, araştırmacıların şu an için üzerinde çalışmalarının en büyük nedeni olan güvenilir güvenlik çözümleri geliştirmek oldukça önemlidir.

Güvenlik sistemlerinin vazgeçilmez bir parçası olan Saldırı Tespit Sistemleri (IDS'ler), RPL tarafından işletilen LLN'lere de entegre edilmiştir. Bununla birlikte, bu çözümlerin çoğu, cihazların ve ağın kısıtlı doğasını göz ardı ederek, IDS'nin özellikle bellek ve güç tüketimi açısından çok maliyetli olmasına yol açar. Giderek daha fazla düğüm tespit görevinden sorumlu olduğu için bu durum daha da sorunlu hale gelmektedir. Bu nedenle, bu tezde, işbirlikçi düğümlerin yanı sıra merkezi bir düğümün katıldığı merkezi bir saldırı tespit sistemi öneriyoruz. Amacımız, mevcut çözümlerin aksine, önerilen IDS modelimizi enerji ve bellek tüketimi açısından düşük maliyetli hale getirmektir. Böylece LLN'yi sürüm numarası, merhaba sel, en kötü ebeveyn ve decreased rank dahil olmak üzere dört tür RPL saldırısına karşı korumak için yalnızca etkinlik değil, verimlilik de garanti edilmektedir.

Bu tez çalışmasında, oluşturulan hafif IDS modeli ile hedeflere (yani etkinlik ve verimlilik) eş zamanlı olarak ulaşmak için, evrim tabanlı bir algoritma olan Genetik Programlama (GP) ve Sıralamayı Baskınlık içermeden icra eden Algoritma-II(NSGA-II) kullanıldı. Burada, önerilen IDS modelinin performansı, değişen topolojilere ve hareketlilik modellerine sahip çok sayıda ağ senaryosunda kapsamlı bir şekilde araştırılmıştır. Sonuçlar, GP'nin çeşitli RPL'ye özgü çeşitli saldırılara karşı düşük maliyetli bir IDS modeli geliştirmek için uygulanabilirliğini gösterdi.

Keywords: RPL, RPL Atakları, Nesnelerin İnterneti, Izinsiz Giriş Tespiti, Haberleşme Maliyeti, Genetik Programlama, Çok Kriterli Yaklaşım.

ACKNOWLEDGEMENTS

To begin with, I would like to thank my supervisor, Prof. Dr. Sevil Şen Akagündüz, for her guidance, inspiration, and criticism throughout the process. Her motivating approach, comprehensive vision, sincere attitude motivated me and gave me strength. I would also like to thank her for the support she has given me throughout my graduate education.

For his excellent advice, I would like to express my gratitude to my 2nd advisor, Dr. Selim Ylmaz, who guided me throughout my research with practical solutions and outstanding academic knowledge. I was able to develop practical solutions to problems because of his patience and solution-oriented approach.

Finally, I would like to express my deep appreciation for my wife Aynur and my children Selvi Su and Nehir Göksu, who always supported me before and during the preparation of this thesis, as well as in every aspect of my life.

CONTENTS

Page

TABLES

FIGURES

ABBREVIATIONS

ACCR	:	**A**ccuracy
6LoWPAN	:	IPv**6**over**L**owPower**W**ireless**P**ersonal**A**rea**N**etwork
CC	:	**C**ommunication-**C**ost
DAG	:	**D**irected**A**cyclic**G**raph
DAO	:	**D**estination-**A**dvertisement**O**bject
DR	:	**D**ecreased**R**ank
DRA	:	**D**ecreased**R**ank**A**ttack
DIO	:	**D**ODAG**I**nformation**O**bject
DIS	:	**D**ODAG**I**nformation**S**olicitation
DODAG	:	**D**estination**O**riented**D**irected**A**cyclic**G**raph
DoS	:	**D**enial**o**f**S**ervice
FANET	:	**F**lying**A**d-Hoc**N**etwork
FP	:	**F**alse**P**ositive
FPR	:	**F**alse**P**ositive**R**ate
GUI	:	**G**raphical**U**ser**I**nterface
GA	:	**G**ennetic**A**lgorithm
GP	:	**G**ennetic**P**rogramming
HF	:	**H**ello**F**lood
HFA	:	**H**ello**F**lood**A**ttack
ID	:	**I**ntrusion**D**etection
IDS	:	**I**ntrusion**D**etection**S**ystem
IEEE	:	**I**nstitute-of-**E**lectrical and**E**lectronics**E**ngineers
IETF	:	**I**nternet**E**ngineering**T**ask**F**orce
IoT	:	**I**nternet**o**f**T**hings
IV	:	**I**ncreased**V**ersion
IVA	:	**I**ncreased**V**ersion-**A**ttack

IVN	:	**I**ncremented**V**ersion-**N**umber
LLN	:	**L**ow-Power-**L**ossy-**N**etworks
MAC	:	**M**edia**A**ccess**C**ontrol
MANET	:	**M**obile**A**d-Hoc-**N**etwork
MRHOF	:	**M**inimum**R**ank with **H**ysteresis **O**bjective**F**unction
MP2P	:	**M**ulti**P**oint**t**o**P**oint
NoF	:	**N**umber-**o**f**F**eatures
NSGA-II	:	**N**on-dominated-**S**orting**G**enetic**A**lgorithm**II**
OFO	:	**O**bjective**F**unction**Z**ero
OSI	:	**O**pen-**S**ystem**I**nterconnection
PF	:	**P**areto**F**ront
PFS	:	**P**areto**F**ront**S**et
P2MP	:	**P**oint**t**o**M**ulti**P**oint
P2P	:	**P**oint**t**o**P**oint
PDR	:	**P**acket**D**elivery**R**atio
RC	:	**R**esource**C**onstraint
RFC	:	**R**equest**F**or**C**omments
RPL	:	**R**outing**P**rotocol-for-**L**ow-Power and Lossy-Network
TP	:	**T**rue**P**ositive
TPR	:	**T**rue**P**ositive**R**ate
UDP	:	**U**ser**D**atagram**P**rotocol
VN	:	**V**ersion**N**umber
VNA	:	**V**ersion**N**umber**A**ttack
WP	:	**W**orst**P**arent
WPA	:	**W**orst**P**arent**A**ttack

1. INTRODUCTION

IoT, which enables a variety of devices to be connected to each other, is one of the most breakthrough advancements in our era. A great deal of IoT applications has found use in various domains including smart homes, smart cities, logistic monitoring, e-health, and the like. That is why the number of smart devices (nodes) enabling such IoT applications has long been increasing. Totaly, in ten years, installed base of these nodes is estimated to achieve 75 billion, a five-fold increase, and the connections of machine-to-machine (M2M) are estimated to constitute 50% of world's hybrid connections until 2030 [8, 9]. As a result of the rapid development of the IoT paradigm, attackers have moved their targets to IoT-enabled devices, arising some security issues in IoT-based networks. Therefore, in recent years, the importance of securing the IoT has increased significantly [10]. Having too many devices connected to IoT necessitates keeping a lot of data together securely, which can be achieved with robust security mechanisms.

The Internet of Things consists of a wide variety of heterogeneous devices, such as mobile devices, smart watches, and wearables. They are characterized as resource-constrained IoT devices (also IoT nodes) in terms of battery, power and storage. In the literature, traditional network protocols are shown to be not suitable for such resource-constrained devices in IoT. The Internet Protocol (IP), which is one of the most common and well-known communication protocols, is not applicable for IoT. Therefore, the need for a less complex mechanism and less resources is achieved by developing a layer called 6LoWPAN [11]. Thanks to this layer, which works in accordance with IEEE's 802.15.4 [12] standards, resource-constrained devices can communicate and connect to the Internet using fewer resources.

LLNs are a type of IoT that provides lossy communication between IPv6-enabled resource-constrained devices. They are characterized by their constrained communication with high packet loss, low throughput, and limited frame size [13]. In a typical LLN implementation, each resource-constrained node communicates with another, but also connects to a special more powerful device called the LLN Border Router (LBR) in order

to connect to the internet. In order to build routes among nodes in such a constrained network, Routing Protocol for Low-Power and Lossy Network (RPL) was developed by IETF-ROLL in 2012 [13], and is today adopted as a standard routing protocol for LLNs today. This important protocol is designed to optimally select the routing path and avoidance of unnecessary routing loops for resource-constrained devices in the IoT network. In addition, RPL works efficiently in these IoT devices, which have low bandwidth, low computing power, and limited energy, and also work in harmony with the changing network topology.

Although RPL is good at building efficient routes between nodes in an LLN, it is still very susceptible to attacks, especially insider attacks. Attacks based on resources, topology, and data traffic can damage the system [14]. The results of such attacks can be vital considering the applications of LLNs in critical systems such as healthcare, and smart home. Therefore, researchers have been working on developing effective intrusion detection systems for RPL-based IoT. However, most studies in the literature deal mainly with detecting RPL specific attacks and overlook the suitability of developed IDS to such LLNs. However, the proposed security solutions for such resource-constrained devices require a balance between security and the usage of limited resources. On the other hand, the existence of many different types of devices introduces complexity to the problem, naturally leading to the formation of complex security solutions. Currently, there is no adequate security solution for IoT networks in the literature. Therefore, there is still a need for new, effective and efficient security solutions. Therefore, the main aim of this thesis is to explore the development of intrusion detection systems that show both high accuracy and low cost.

In our proposed ID program, a collaborative and centered ID system architecture is explored in which a global ID node is placed at the root node and some other nodes called monitoring nodes participate for intrusion detection sending their local information to the root node. Although involving the monitoring nodes brings about an additional burden to the network and devices, they enable the global ID node to capture intrusions on a global scale, hence more effectively. In this study, the data required for intrusion detection is collected not only from the root node but also from 1-3 hop neighbor nodes and 4-6 hop neighbor nodes

of the root node. Here, the size of the information collected and sent by the monitoring nodes becomes important in terms of resource consumption. In addition to increasing communication cost, large packets sent by monitoring nodes can also lead to fragmentation. Therefore, in this thesis, while developing intrusion detection systems, beside their accuracy, the information used and sent by the monitoring nodes is taken into account. For simplicity, this information regarding the cost of intrusion detection and communication is taken as the number of features used for training in this study. Therefore, the detection accuracy and the number of features extracted from both the ID node and the monitoring nodes must be tuned simultaneously to generate an effective and efficient IDS, which is the main motivation of this study. It is emphasized that a trade-off between detection accuracy and cost of communication has not yet been discovered in the literature [1]. The motivation point of the study is to fill this gap by developing effective intrusion detection algorithms taking into account communication cost. In order to overcome this complex and difficult multi-objective optimization problem, we employ genetic programming (GP) due to its ability to explore search space efficiently for complex environments such as LLNs and to also handle multiple objectives (i.e., accuracy and the number of features in this study) simultaneously. GP's main goal is to evolve a detection program (or model) that finds a good trade-off between accuracy and the minimum number of features used. Only evolved features are extracted and sent by the monitoring ID nodes to the central ID node, which then periodically runs the evolved program. To handle multiple objectives by GP, we employ Non-dominated Sorting Genetic Algorithm II (NSGA-II) [5], one of the most popular Pareto-based evolutionary multi-objective algorithms. The following four attacks are covered in this study: "worst parent", "hello flood", "increased version" and "decreased rank" attacks. Various network scenarios with these four attacks in which attackers are placed in different locations are evaluated and discussed. As expected, the experimental results show that the increase in the number of nodes and the number of data packets used in intrusion detection also increases the number of features (NoF) used, resulting in an increase in power consumption and a decrease in network performance. For WP, HF, and IV type attacks, GP can produce a satisfactory ID program with an average detection accuracy of 94%. On the other hand, limiting the number of features has an adverse impact on the detection of DR.

1.1. Scope Of The Thesis

In this thesis, genetic programming (GP) is employed to develop and evolve intrusion detection algorithms. The main reason for using GP is that it is good at discovering complex environments such as RPL. Moreover, it is good at discovering trade-offs such as accuracy and communication cost that is aimed to be optimized in this study. GP is a population-based algorithm which optimal (close to optimal) solutions for the current problem and usually the best one is chosen for testing [1]. This feature of GP ensures that the evolution is completed by transferring the subpopulation of the best individuals of each generation to the next generation. Mutation and cross-over operators of GP also make a tremendous contribution to providing better and fitter individuals in each generation. Although there are a number of studies in the literature for prevention and detection of RPL attacks, they do not consider limited resources of devices. In this thesis, a GP-based multi-objective approach is proposed to not only detect the following RPL-specific attacks but also do that efficiently: *i*) Worst Parent, *ii*) Hello Flood *iii*) Increased Version, and *iv*) Decreased Rank. To the best of our knowledge, it is the first study in the literature to discover trade-offs between communication cost and accuracy of intrusion detection algorithms developed for RPL-based IoT networks.

1.2. Contributions

Contributions of this study include the following:

- In the study, the detection of RPL-specific attacks by using GP based on the multi-objective approach is explored. According to the results, GP is able to evolve effective detection algorithms for “hello-flood”, “increased-version”, “decreased-rank”, and “worst parent” attacks.
- Our evolved intrusion detection program has been proposed not only to detect the RPL-specific attacks (WP attack, HF attack, IV attack, and DR attack), but also to do that efficiently. Minimizing the communication cost is of vital importance in

resource-constrained devices and networks created with these devices. In this study, it is aimed to minimize communication costs and therefore to use less resources.

- As far as we know, there is no study that covers both accuracy and communication-cost objectives together in this research area yet.

1.3. Organization

Following is a description of the thesis' organization:

- We outline our motivation, scope, contributions, and organizational structure in Chapter 1.
- Chapter 2 gives basic information about RPL, RPL repair mechanism, specific RPL Routing Attacks, optimization, multiobjective-optimization, intrusion detection pragrams (systems), classification of intrusion detection systems, genetic programming (GP), non-dominated sorting genetic algorithm-II(NSGA-II).
- Chapter 3 presents related studies in the literature related to intrusion detection and attack analysis in RPL.
- Chapter 4 introduces the multi-objective based intrusion detection system (ID program), focused RPL-specific attacks ("decreased-rank-attack", "increased-version-attack", "hello-flood-attack", "worst-parent-attack"), intrusion detection architecture, evolving intrusion detection algorithms (the features, the representation, fitness function, detection accuracy, communication cost) in detail.
- Chapter 5 presents the evaluation of the proposed approach, experimental settings, simulation environment, simulation environment with mobile attackers, performance metrics, experimental results, the performance of IDS on the targeted attacks, and general discussion.
- Summarizing and concluding the thesis is presented in Chapter 6.

2. BACKGROUND OVERVIEW

The IoT, Internet of Things, is a state-of-the-art network structure that allows various devices (also nodes) to connect to each other and open the Internet according to certain rules and protocols, and also embraces very different applications [15, 16]. In 1999, the concept of IoT was proposed by Kevin Ashton and was referred to as unique objects identified by radio frequency identification (RFID) [17]. With each passing day, the number of IoT devices and networks that these devices connect increases rapidly. According to Statista [18], the number of devices connected to the Internet of Things (IoT) worldwide is expected to reach around 25 billion by 2030, representing a three-fold increase in eleven years. Although some IoT applications are still in their infancy, advanced IoT applications are emerging in many areas and excite the world. However, the main IoT architectures are still being defined today, IoT basically aims to keep devices or objects in interaction [19]. As a result of this interaction, objects are brought together and, after that, data can be shared, and eventually communication with the outside world can be established [20].

The Low-Power and Lossy-networks (LLNs) are the special networks that consist of many integrated nodes with low power, limited memory and resources, interconnected by lossy links. It has a wide range of uses, from smart devices to space technologies. These LLN networks have resource constraints such as memory and battery (energy). In addition, such resource-constrained devices, unfortunately, constantly consume resources to keep their connection active and to become a member of the network. These disadvantageous features of LLNs necessitate the emergence of new communication protocols. LLN routing protocols are often designed to simplify the design and hence reduce energy consumption [21]. The Low Power and Lossy Network Routing Protocol (RPL) [13] is a well qualified routing protocol designed specifically for LLN networks [13], which has become a standard protocol today [22]. Therefore, this section presents RPL specifications and discusses specific RPL routing attacks. Furthermore, the fundamental information about intrusion detection systems is covered in the section. Finally, the optimization algorithms which are used for

evolving effective and efficient ID programs (IDSs) for RPL-based networks, namely GP and NSGA-II, are given in this section.

2.1. RPL

A loop-free and standart state-of-the-art distance vector routing protocol, RPL, is also a proactive routing protocol that was defined and explained in RFC 6550 [13]. Since RPLs are designed for low power and lossy networks (LLNs), in an IoT architecture that has limited resources such as energy and bandwidth, RPL can be used effectively [23]. RPL allows new paths to be established when existing paths are no longer available, and it allows one to optimize existing paths.

RPL enables these types of communication: *i*) "point-to-point (P2P)", *ii*) "point-to-multipoint (P2MP)" and *iii*) "multipoint-to-point (MP2P)". In an IoT network, communication between sensor nodes is called *P2P*, communication between root node and sensor nodes is called *P2MP*, and communication between sensor nodes and root node is called *MP2P*. The main aim of RPL is to create a DODAG topology. DODAG is a special and well-qualified combination of both mesh and tree topology. The LLN nodes can connect to each other and 6BR thanks to RPL. Three different types of nodes are defined in this protocol: *i*) "LLN border router (LBR)", *ii*) "router", and *iii*) "host". A node's operations in the DODAG topology are shown in Figure 2.1 LBR, the root node of DODAG, collects data from the network. This collection point also creates a "Destination Acyclic Graph" (DAG). In addition, the LBR provides a connection between the remaining nodes and Internet. As part of the IoT, routers produce data traffic and send data packets. Destination Acyclic Graph (DAG) cannot be created by routers, but they can be joined by network messages. However, the host is responsible for producing data traffic over a network. In a DODAG topology, each node has some information and features: *i*) an IPv6 address of the node called node ID, *ii*) a rank value, *iii*) a neighbours list, and *iv*) a parent node selected according to the Objective Function (OF). As illustrated in Figure 2.2, RPL has two different route discovery modes: *i*) mode of storing and *ii*) mode of non-storing. Only the 6BR device keeps the

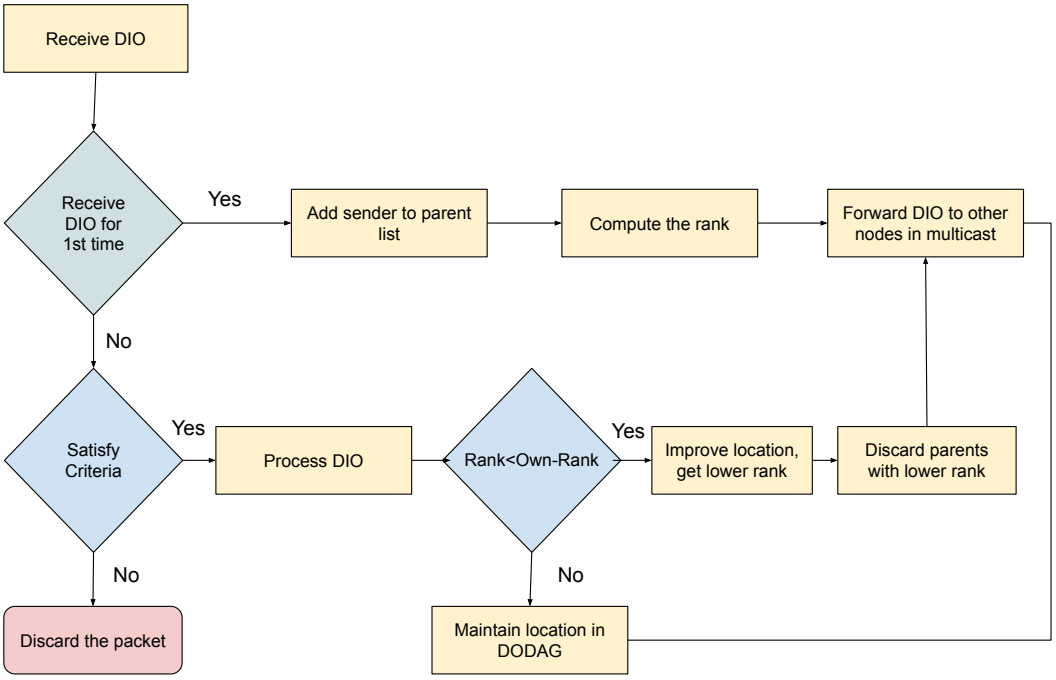
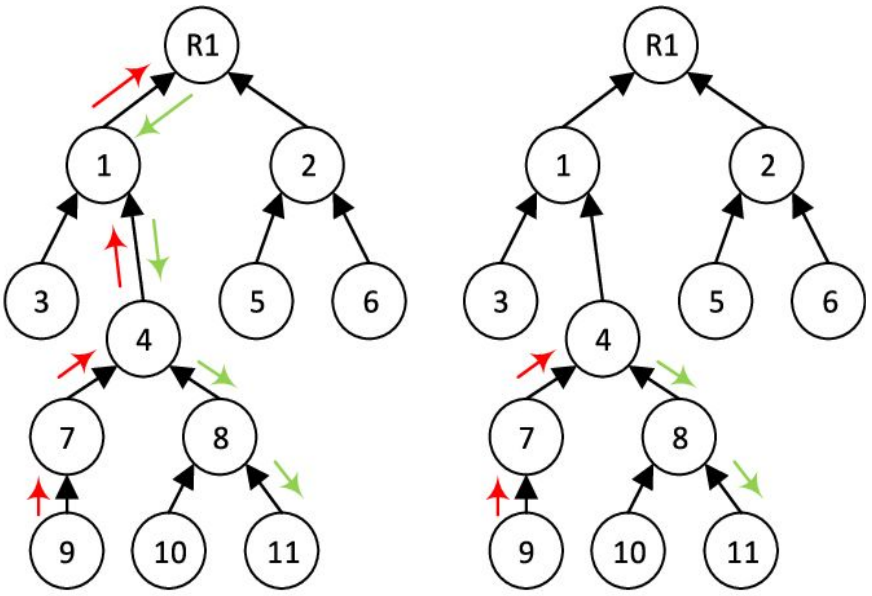


Figure 2.1 A node's operations in a DODAG topology.

routing information, in non-storing mode. This mode contains all the path information that the packet follows to navigate the network. However, in storage mode, all nodes have routing information for nodes in their subgraph. A DODAG topology is created using RPL control messages. Here, the control messages: *i*) DODAG-Information-Object (DIO), *ii*) DODAG-Information-Solicitation (DIS), *iii*) Destination-Advertisement-Object (DAO), and *iv*) Destination-Advertisement-Object-Acknowledgment (DAO-ACK).

- DIO : Root node, responsible for creating the DODAG structure. It initially broadcasts this control message to all other DODAG's devices in this topology. The DIO messages carry some network information, such as rank value, node ID, version number, and OF. All nodes receiving DIO messages from root node add the information of sender to their parent list and calculate own rank. Nodes receiving DIO control message also forward the message to its neighbors. Nodes in the graph are ranked based on how far they are from the root node. Therefore, the rank value has to do with the quality of their



(a) RPL non-storing mode (b) RPL storing mode

Figure 2.2 RPL non-storing and storing mode [2].

routes to the root. In order to calculate the rank value, each node uses OF that defines rule on the parent node selection so that the optimum route to the root node in DODAG is established according to driven routing metric [23] [24]. While calculating the rank, OF may not only consider the physical path length. For example, the most important parameter in a network may be the number of hops, load balancing, or another metric. Consequently, due to the OF metric, even if the nodes are physically near the node, it may not select this node as its parent node due to OF. Nodes are required to update their DIO status messages when their parent or rank changes. In addition, DIO messages play a prominent role to built default upward paths.

- DIS: In order to request a DIO from a node, DIS control message may be used. Candidate nodes hoping to join a DODAG topology broadcast DIS control messages to their neighbors. Here, a DIO message is sent as a response to the nodes that receive the DIS message. In addition, a node can use DIS messages to search its neighborhood for nearby DODAGs [13].

- DAO: It is used to build downward routes between root node and the sensor nodes. Here, the preferred mode of operation determines a child nodes send unicast DAO

control messages to the selected parent node or to the root node. Thanks to the DAO control messages, this stores downward paths into own routing table.

- DAO-ACK: After receiving a DAO message from a node, a child node sends DAO-ACK message as an acknowledgement response to the sender node.

As a way of building upward paths, in RPL, nodes send messages to their neighbours when they get a message. However, if this node is not in its list, the packet is forwarded to the parent nodes. Of course, in order to prevent increasing routing loops, the rank value is used. The rank value also allows nodes to know the parent and child nodes in the neighbourhood. If the existing connection is lost with the currently selected parents, new parents can be found using the neighbour list available on all nodes. The new route is found according to the metrics defined in OF. As a result, to create a DODAG topology, all nodes decide which node their parent is, based on the best path they have chosen up to the root node.

2.1.1. RPL Repair Mechanisms

Two different repair systems are specified in RPL: “local-repair mechanism” and “global-repair mechanism”. If any failure occurs with any node or routing topology in the network, RPL runs its mechanism, called local repair. After the nodes disconnect from the parent nodes, they check the neighbour lists and try to select alternatives. However, if this node cannot find an alternative for the parent node, it uses a neighbouring node that has the same rank value to forward packets. If network sustainability cannot be achieved with the local repair mechanism, the global repair mechanism can be used. In order to control this global repair mechanism, the root node uses the DODAG version number. If a root node chooses to run global repair mechanisms in the RPL network, it increments the DODAG version number. The incremented version number is carried by the DIO message that is broadcast to the network by the root node. A node that receives the DIO control message with the incremented version number executes the parent selection algorithm. As a result of the operation of the parent selection algorithm, the parent node is updated. In fact, the trigger

for the global RPL repair mechanism can vary depending on the implementation created [25]. Therefore, users should define the trigger status of the global repair mechanism in the RPL network. This trigger can be either threshold-based or specification-based.

2.1.2. Routing Attacks against RPL

Even though RPL has some security specifications specified in the RFC [13], that is still open to internal attacks such as worst-parent and increased-version attacks [1]. The taxonomy of specific RPL attacks is shown in Figure 2.3 [3]. Such attacks may be categorised into three types [26]:

- **Attacks targeting resources:** In these attacks, the malicious node(s) force the victim node to take continuous actions different from their aims. In this way, the malicious node(s) goal to consume the available restricted resources such as battery, power and memory, while at the same time causing the established connections to be disabled. As stated above, the resources in such networks are unfortunately very limited.

- **Attacks targeting topology:** In this type of attack, the attacker(s) dramatically affect(s) the construction of RPL topology in a non-optimal way. Here, the attacker(s) aim to distort the RPL topology. The "'Worst Parent Attack", a type of rank-specific attack, is one of the attacks in this group. In this type of attack, the attacker selects the node with the non-real rank as its parent node, thus ensuring that the nodes now have a path containing the malicious node when they select a path up to the root node. The topology is a vital element for the systematic operation of a network. Attackers who aim for this attack can sometimes also isolate legitimate nodes within the network. In this way, they can achieve their goals by keeping the legitimate nodes they want out of the network, albeit temporarily. Another attack type in this group is "blackhole attack". In this attack type, an attacker announces that he has the shortest route (i.e. the best rank) to attract victim nodes for sending their packets through the attacker. Therefore, this node will always be able to respond to the routing request and will

capture and retain data packets. Sometimes, the attacker node does not allow the packets passing over itself to go to the root node and causes the packets to drop. Hence, it can isolate many legitimate nodes from the network, especially if the attacker is at a critical position in the network.

- **Attacks targeting traffic:** With this type of attack, the attackers are trying to disrupt the network traffic by attempting to interfere with the traffic. This can change the traffic patterns and enable attackers to achieve their goals. In fact, this type of attacks' basic aim is to prepare the infrastructure for more complex attacks.

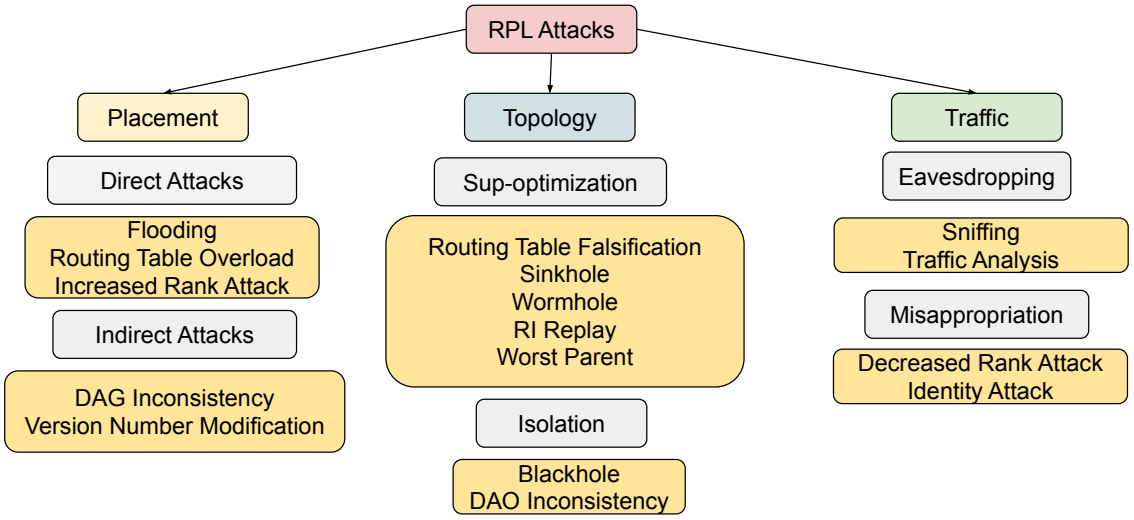


Figure 2.3 The taxonomy of attacks against RPL [3].

2.2. Intrusion Detection Systems (IDSs)

It has been stated in previous sections that some security mechanisms are needed for protection LLNs from attacks. The first priority in the security mechanisms is the prevention of both internal and external attacks. However, there is a trade-off between usability and security. Therefore, a system might not prevent the system from all types of attacks. Moreover, such mechanisms cannot prevent insider attackers who have legitimate access to the system. Therefore, Intrusion Detection Systems (IDSs), which detect attacks and

responds, are introduced as a second line of defence. As shown in Figure 2.4, we can basically divide IDSs into three main categories, according to *i*) placement, *ii*) technique, and *iii*) architecture[4].

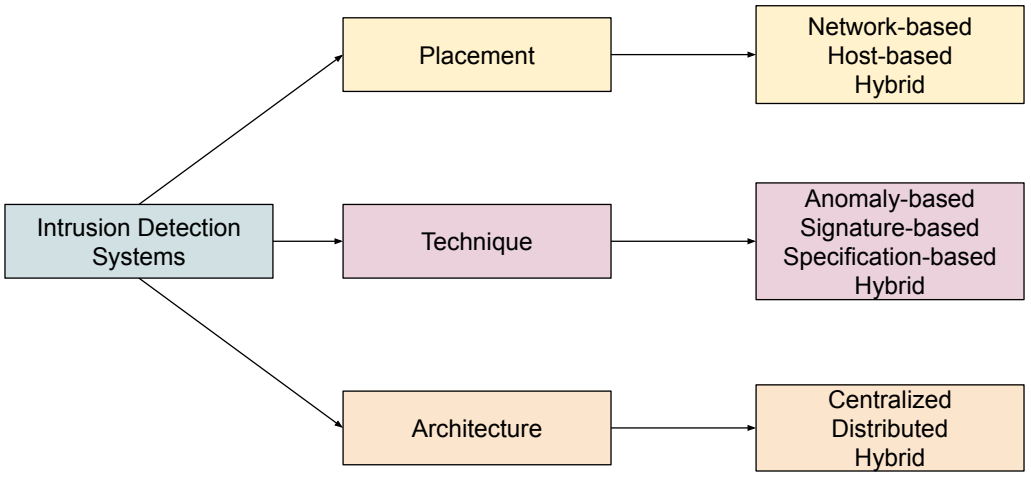


Figure 2.4 The classification of IDSs [4].

2.2.1. Intrusion Detection Placement

Depending on its location, intrusion detection systems fall into two categories: *i*) based on networks and *ii*) based on hosts. This type of IDS is placed in an area to monitor the entire network or parts of it. While such systems have global knowledge, they do not have individual nodes' information such as their resource consumption. On the other hand, host-based detection systems focus only on collecting network information from a node, and they do not have information about the entire network. Hence, they might not detect distributed and collaborative attacks.

2.2.2. Intrusion Detection Techniques

There are three main classes of intrusion detection techniques: *i*) anomaly-based, *ii*) signature-based, and *iii*) specification-based [27].

Malicious activities are detected on networks by first identifying normal network behavior, patterns, and rules, and then comparing the current system profile to this baseline in anomaly-based systems. Therefore, observations that deviate more than the normal range of normal profile patterns are marked as abnormal behaviors [28]. Anomaly-based attack detection systems have capabilities that can be adapted to all attack types. Anomaly-based attack detection systems can detect new types of attacks. However, it is difficult to define normal behaviour which can changed over time. Signature-based detection systems, compares the behaviour of a system with signatures created based on known attacks. Therefore, this technique cannot detect new attacks, hence the signature database should be updated frequently. In specification-based systems, the specifications of a system are defined, and violations of such specifications are tagged as attacks. For example, whenever any node in an IoT network fails to follow the routing protocol specifications, IDS generates an attack alarm. In spite of this, defining specifications for every protocol used in the system is not an easy task.

2.2.3. Intrusion Detection Architectures

We can divide intrusion detection architectures into two main groups: *i*) centralized and *ii*) distributed systems. In centralized systems [29], the monitoring tools are located in central locations to observe the network. In an IoT network, taking into account the resource constraints of the devices, the IDS is placed in the most appropriate location, often a higher power node, for the running of the intrusion detection (ID) program. A centralized intrusion detection system has two options: *i*) a centralized device monitors the network data flow that passes through it, or *ii*) the central node collects data from other nodes. The first approach is largely suitable for a wired network because the IDS can be placed on a border router to monitor the entire network data flow. However, this approach can be quite unsuccessful in highly mobile wireless networks such as MANETs [30] and FANETs [31] (Mobile Ad Hoc Networks and Flying Ad Hoc Networks). Since the entire network flow cannot be monitored by a central node. Although RPL is proposed mainly for static networks, many devices in RPL-based networks are mobile in real life. In the second approach, where the central

node collects data from other nodes, communication overhead may occur due to excessive data flow, since the monitoring nodes have to send information to the central node. Here, communication overhead is the main disadvantage. In this approach, malicious nodes can block the monitoring information from reaching the central node. As a result, this can greatly reduce the performance of IDS.

In distributed systems, all nodes work as monitoring nodes [32]. Since all nodes are used and the resource constraints of the devices are taken into account, the system should be carefully designed. With this approach, system overhead could increase as the nodes share information between themselves. However, if the system is not collaborative, there is no communication overhead, but each node makes its own evaluation based on its own observation. The main aim of these hybrid systems is to make use of centralized and distributed systems and take advantage of these systems [33].

2.3. Optimization

Optimization is a search for the best solution or at least accepted from an infinite number of candidates in a search space, which may also be limited by certain equality or inequality constraints [34]. The optimization problem that involves equality and inequalities limitations can be written as follows:

$$
\begin{aligned}
&\underset{X}{\textit{minimize}} \;\; f(\vec{\boldsymbol{x}}), \\
&subject\ to \\
&\qquad g(\vec{\boldsymbol{x}}) = [g_1(\vec{\boldsymbol{x}}), g_2(\vec{\boldsymbol{x}}), \ldots, g_p(\vec{\boldsymbol{x}})] \leq 0, \\
&\qquad h(\vec{\boldsymbol{x}}) = [h_1(\vec{\boldsymbol{x}}), h_2(\vec{\boldsymbol{x}}), \ldots, h_q(\vec{\boldsymbol{x}})] = 0
\end{aligned}
$$

As seen from the formulation, the objective function (f) characterizes the search space, and therefore, plays a key role for an optimization algorithm to find local optimum ($\vec{\boldsymbol{x^*}} = [\boldsymbol{x_1^*}, \boldsymbol{x_2^*}, \ldots, \boldsymbol{x_n^*}]$) of the optimization problem by gradually developing a given decision

variable vector ($\vec{\boldsymbol{x}}$). Apart from the objective function, the p inequality constraints ($g(\vec{\boldsymbol{x}})$) and the q equality constraints ($h(\vec{\boldsymbol{x}})$) define the *feasible region* (Ω) of the search space, and any solution vector within the feasible region (i.e., $\vec{\boldsymbol{x}} \in \omega$) is called a *feasible solution* [35].

The optimization problems in real life are characterized by a number of complexities including non-differentiability, non-linearity, and the like. That's why, it is very hard, and even impossible, to solve such optimization problems by using the traditional optimization algorithms. In order to effectively deal with these complexities, nature-inspired metaheuristic algorithms have gained popularity over the past few decades. Based on the metaphor of natural processes, several metaheuristics have been developed until now, and the majority of them are inspired by intelligent behavior of the creatures. Depending on the number of objectives described for the optimization problem, these algorithms are mainly classified into *single-objective* and *multi-objective* heuristics. Here, multi-objective heuristic optimization methods are often can be classified: swarm- and evolutionary based [35]. In this study, a multi-objective heuristic approach is used to find an intrusion detection model called the ID program in RPL. Background information regarding the concept of multi-objective optimization, and the optimization algorithms adopted in this thesis are introduced briefly in the subsequent sections.

2.3.1. Multi-objective Optimization

Unlike the single-objective optimization where the aim is to find the global minimum (for minimization problems) or maximum (for maximization problem) for a given single-objective function, in multi-objective optimization the handled problem may involve a number of conflicting objectives and the main aim here is to solve multiple objectives simultaneously. The search algorithms for handling multi-objective problems can be pretty much in the literature and are categorized according to the way that these algorithms adopt. Among them, the Pareto-based strategy is the most popular. Here, the solution developed for a targeted multi-obective optimization problem isn't only a single solution, but also an optimal solution vectors set or *Pareto optimal* solutions (nondominated solutions).

Therefore, the determination of dominated solutions is the core operation of this strategy. Here, a solution vector $\vec{\boldsymbol{x}}$ 'dominates' another solution vector $\vec{\boldsymbol{y}}$ as long as no criterion in the objective vector of $\vec{\boldsymbol{x}}$, $f(\vec{\boldsymbol{x}})$, is worse than that of $f(\vec{\boldsymbol{y}})$ and at least one criterion is better [35]. By adopting this domination-based comparison operator, one can reach a final set in the end that contains only the solutions that are not dominated. So, the main goal of the Pareto-based approaches is to have Pareto optimal set. Taking into account the Pareto front set, decision-makers choose the best solution depending on their requirements after evaluating different trade-offs among conflicting objectives. An example of the basic concept definition used in a Pareto-based MO optimization using the minimization problem is presented below:

Pareto Dominance: An n-dimensional solution $\vec{\boldsymbol{x}} = [x_1, \ldots, x_n]$ dominates another n-dimensional solution $\vec{\boldsymbol{y}} = [y_1, \ldots, y_n]$ (denoted as $\vec{\boldsymbol{x}} \prec \vec{\boldsymbol{y}}$); if $f(\vec{\boldsymbol{x}})$ is partially less than $f(\vec{\boldsymbol{y}})$, i.e., $\forall_i : f_i(\vec{\boldsymbol{x}}) \leq f_i(\vec{\boldsymbol{y}}) \wedge \exists_i : f(\vec{\boldsymbol{x}}) < f(\vec{\boldsymbol{y}}) \mid i \in \{1, \ldots, m\}$; $\vec{\boldsymbol{x}}$, $\vec{\boldsymbol{y}} \in \Omega$.

Pareto Optimal: If no other solution dominates $\vec{\boldsymbol{x}}$ in Ω, then $\vec{\boldsymbol{x}}$ is said to be Pareto-optimal solution; i.e., $\neg\exists\vec{\boldsymbol{x'}} \in \Omega : \vec{\boldsymbol{x'}} \prec \vec{\boldsymbol{x}}$. The Pareto optimal solutions comprise Pareto optimal set ($\mathcal{P}^*$).

Pareto Front: A set $\mathcal{PF}^*$ containing all objective vectors from Pareto optimal set ($\mathcal{P}^*$), i.e., $\mathcal{PF}^* = \{f(\vec{\boldsymbol{x}}) \mid \vec{\boldsymbol{x}} \in \mathcal{P}^*\}$ [35]

2.3.2. Genetic Programming (GP)

GP is a popular evolutionary-based optimization algorithm and this is inspired from the *Darwinian survival-of-the-fittest* theory. It relies on a population in which a pre-defined number of agents, called individuals, take part, and each individual is in charge of the optimization task by performing a search in the space. Individuals are encoded with a tree structure, called GP tree, where terminal and non-terminal types of node take part. The terminals and non-terminals form the leaf and intermediate nodes of the GP tree, respectively. The individuals are attributed by a *fitness* score, which is an indicator in GP to measure

how well the solution is reached by an individual. The population is sorted by ranking the individuals, and for the next generation only the fittest individuals, called elite individuals, will survive. In this study, the elite individual rate is set 10% of the population size. In addition, it is more likely for fitter individuals to breed their offspring, which enables GP to find the better solution at the end of generations. Here, this solution is expected to be the best solution. The main parts of a genetic programming are shown in Algorithm 1 [1].

Algorithm 1: Main parts of GP [1].

```
Initialize population randomly;
repeat
    Calculate all individuals' fitness;
    Sort and rank populations by fitness value;
    Reproduce new population using genetic operators (mutation, crossover etc.);
until a termination criterion is satisfied;
return best-of-run individual
```

As also shown from this algorithm, in the current generation the parent individuals undergo three consecutive genetic functions: *i*) selection, *ii*) crossover, *iii*) mutation to evolve new children for the next generations. In selection, a pair of individuals is selected, and the fitness value of an individual plays a key role here to determine if it is reproduced in the next generation. The selection is made depend on the operators of selection process, for instance roulette wheel, tournament-, elite-, random-selection, and the like. Among them, tournament selection is widely adopted in the literature because it well balances a trade-off between the random selection and elite selection depending on the tournament size. In this strategy, a number of individuals is picked randomly first, then the fittest individual is selected from that subpopulation. At a point called crossover point, two new offsprings are produced by crossover operator with replacing the subtrees randomly determined. By the way, subtrees of the offspring individuals are also replaced by, contrary to crossover, randomly generated new subtrees in mutation. Hence, better individuals are aimed to be evolved through generations. GP reaches the end of the generation once the termination condition is satisfied. There may be different conditions such as reaching the total number of generations, approximating well to the ideal or optimum solutions, and the like.

2.3.3. Non-Dominated Sorting Genetic Algorithm-II (NSGA-II)

This algorithm was developed on the basis of NSGA [36] to ensure a more effective and efficient search procedure than that in NSGA and NSGA-II algorithm is one of the multi-objective evolutionary-based approach proposed in [5]. Because multi-objective problems' solution spaces are not affected, the genetic operators can also be applied to the population in NSGA-II. However, in objective space, unlike single-objective evolutionary algorithms NSGA-II uses Pareto domination and crowding distance, an additional diversity metric, of the solution sets to determine which individuals will survive and breed for the next generations. Therefore, individuals are attributed with these metrics in this algorithm.

In order to evaluate the population in the objective space, the individuals are first ranked into different Pareto fronts according to their level of dominance. While the non-dominated individuals are ranked as 0 and placed at the first front, the non-dominated individuals are ranked as 1 and placed at the second front, and so on.

After the individuals are ranked into different Pareto fronts, NSGA-II allows only the individuals that belong to the better Pareto fronts, and where the count of individuals of a Pareto front surpassed the size of population, only a portion of individuals in that front are accepted according to their evulation of crowding distance metrics. The crowding distance in NSGA-II is used to show how close two individuals are to each other. That's why, this metric is very useful to indicate the variety of population in the objective space. Here, the population is first consecutively sorted according to every dimension in the objective space for calculating the individuals' crowding distance score. Individuals having extreme objective values (i.e., minimum or maximum) are assigned to an *Infinite* crowding distance value, so NSGA-II selects more likely them, that also increases the divergence of all population belongs to their solution space. For other individuals between the extreme points, a distance value is calculated with respect to the nearest two individuals. Here, an individual's total crowding distance metric score is can be found with adding the distances calculated for each value in the objective space. The selection strategy proposed in NSGA-II that based on crowding distance metric is demonstrated in Figure 2.5.

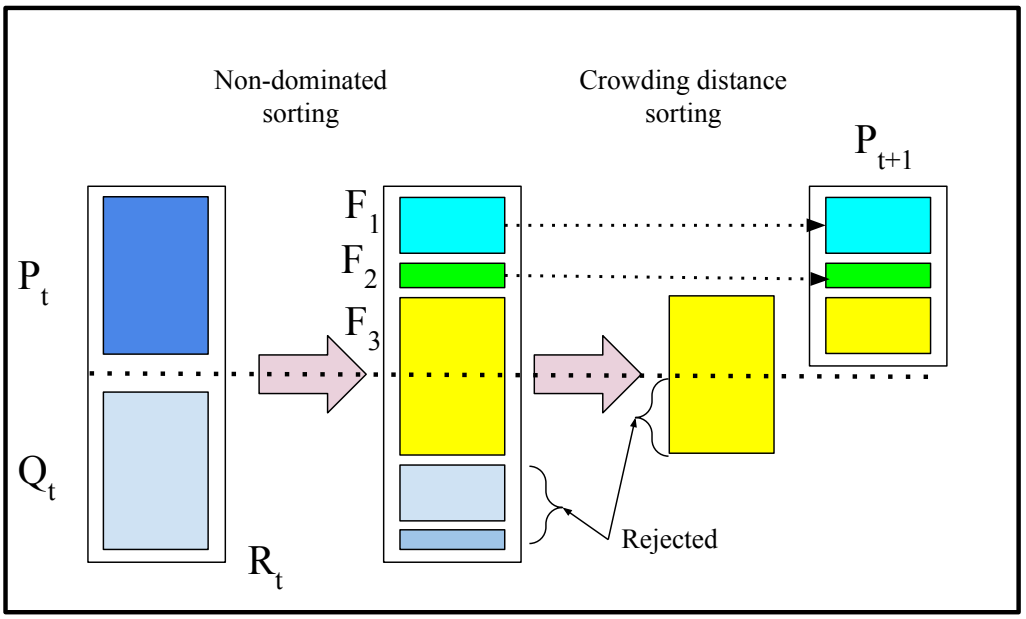


Figure 2.5 NSGA-II procedure [5].

Here, P_t is the parent population and Q_t is the offspring population at generation t obtained from P_t using genetic operators. The parents and offspring are merged into a set denoted R_t. Individuals are then classified into different fronts (e.g., F_1, F_2 and so on). While all the individuals at F_1, F_2 survive to the next generation by allowing them to pass P_{t+1}, only a portion of the individuals at F_3 is selected according to the value of crowding distance metric. However, the rest of the lower front is eliminated.

3. RELATED WORK

Due to the lack of adequate security mechanisms in its specifications, many studies have been carried out for securing RPL-based IoT networks in the literature. Until now, for wired and wireless networks, although many security solutions such as IDSs have been proposed, such traditional IDSs are not suitable for LLNs due to their very nature. Therefore, new solutions are proposed or existing solutions are adapted to these low-power and lossy-networks in this research area. We summarize the proposals in this section and show the contribution of the current study. In this section, we firstly present the studies carried out on analysis of RPL-specific attacks, then the proposals for detecting such attacks.

3.1. Attack Analysis for RPL-based IoT Networks

There are many analysis studies on different routing attack types in the literature and one of the most focused attacks are Version number attacks (VNAs). The performance of mobile RPL-based IoT networks under version number attack is analyzed in [37]. Here, the performance of these RPL-based network is analyzed in terms of PDR, delay, and battery consumption. Contiki Cooja simulator is used and different scenarios are generated by putting the attacker node at different locations. In addition, the study includes a number of nodes (50% mobile nodes, and 50% are not mobile nodes) and multiple attackers. Hence, because of mobility, a hybrid network is created. Different numbers of nodes are used when performing experiments (ranging from ten to fifty in increments of 10). With increasing number of malicious nodes, PDR drops dramatically. Here, the version number is changed, if the malicious node is not far away from the root node and this root node forcefully repairs the DODAG topology. Not surprisingly, mobility with the attacker or other nodes significantly reduces the PDR. Having no attack results in a very low delay; on the other hand, at different hops level the malicious nodes number increases and also the latency increases. The results also show that when mobile networks and static networks are compared to each other, generally the power consumption for mobile networks tends to increase. Furthermore, not surprisingly, the attackers at 1-hop distance are more likely to consume energy.

In [6], the performance of RPL-based IoT is analyzed against multiple version number attackers (VNA). Version-number attacks are a kind of damaging DoS attack that targets IoT network availability. Therefore, it can easily affect network performance and quickly misuse resources. Here, the Contiki Cooja simulator [38] is used for the experiments. Various criteria, except from the performance metrics, are taken into account for analyzing and understanding multiple version number attacks, including architecture and attacker mechanism. Here, a grid network architecture consisting of 16 static Tmote Sky nodes is constructed, and UDP protocol is utilized. Here, the created DODAG topology is limited to three malicious nodes corresponding to 20% of the nodes. The position of the attacker is considered to find the best effective place. They developed a novel strategy called DODAG leveling according to the position of nodes, since thousands of experiments were required to determine the most suitable location for the attacker. Therefore, the root node is always at level 1, while its child nodes are at level 2. Here, they acted on the assumption that attackers at the same level logically have the same characteristics and adopted the brute-force approach. Therefore, the DODAG topology was logically leveled up into four different parts according to the parent-child relationship. In the study, the attackers are independent from each other and increment the version number in their DIO messages. Packet delivery ratio (PDR) are used to analyse the effects of the attacks. In addition the following metrics are used: averag-network delay and -power consumption(APC). These results, based on the attackers' position, show that the attackers cause the lowest PDR if they are at the center of topology. On the other hand, surprisingly, three attackers case cause shorter average delay than one or two attackers cases and the amount of successfully delivered packets results show that they have the relation of *one attacker* > *two attackers* > *three attackers*. In addition, the findings show that the effect of an attack is proportional to the position of the attacker.Hence, in order to effect the average delay, it cannot be said that the attacker should be close or far from each other. On the other hand, if the attacker nodes are placed on DODAG's edges, they cause more delays than attackers nodes inside of the topology. With regard to the attacker's position, results show that the closer the attacker's position, the more power consumption has an effect. Finally, in future work, an analysis of the effectiveness of VNA mitigation techniques and how multiple attackers affect more complex topologies is provided.

The rank value is changed by the malicious nodes in a rank attack. The rank system of RPL makes it vulnerable to internal attacks. Due to it's nature, RPL assumes that all nodes in the network are non-malicious therefore there is no mechanism to check the behavior of nodes. In this attack, the nodes are attracted to choose the attacker node as their parent node. First analysis that includes several types of internal attacks that target rank feature and the effects of them at network efficiency is given in [39]. This study focuses on various rank attack mechanisms depending on the attacker's location. Here, different types of rank attacks are analyzed. Contiki Cooja's simulator performs a 100 node grid topology. The following metrics are used for measuring the network performance: average end-to-end delay and delivery ratio. The experimental results for this study are as follows: *i*) When malicious nodes are in the right location, they downgrade network performance. *ii*) There is no RPL mechanism for children to verify their parents other than routing information carried by DIO packets. *iii*) The average-delay and delivery ratio is very susceptible to rank attacks. *iv*) If the malicious node is choosen as a node's parent, all nodes in its range will suffer adversely from performance.

The objective function plays a critical role in determining the optimal route in RPL. Therefore, it can be targeted and impacted by attacks. A comprehensive analysis is given in [23] to investigate how the performance of RPL with different objective functions is affected under varying types of attack. In this study, the four performance metrics are used for evaluating the experimental results (PDR, traffic overhead, latency, battery) and three different attacks against RPL are analyzed: VN attack, DIS flooding attacks, WP attacks. A variety of attacks scenarios are implemented in simulated networks to analyze the effect of specific routing attacks, and to simulate each scenario the Cooja Simulator is used. In the simulation environment these objective functions are selected: OF0, MRHOF-ETX and MRHOF-ENERGY and the experiment are conducted under version number, DIS flooding, and worst parent attacks. In addition, to simulate real lossy links environment in LLN, UDGM is used. Here, ten different networks are simulated with different percentages of attackers (0%, 2%, 6%, and 10% percentages of attackers) for these three types of attacks. The experimental results show that version number attacks result in a big increase

according to the control packets' number. On the other hand, MRHOF-ENERGY produces the biggest control messages number for all attacks types, particularly DIS flooding attacks. In the network, when there are more attackers MRHOF-ETX results in less overhead than objective-function-zero. The number of attackers that affects MRHOF-ETX is less than OF0. OF0 and MRHOF-ETX values have the approximately satisfactorily PDR scores if the network is under no attack. In a benign network, there are no significant performance differences between objective functions when it comes to average latency. However, when the network under version number or DIS flooding attacks, the average latency values are greater than the worst parent attack. The overall results show that MRHOF-ENERGY uses more power than MRHOF-ETX. In addition, energy consumption of version number and DIS flooding attacks is bigger than the worst parent attack. Another study [40] was carried out to evaluate the effect of local repair attack, worst parent attack, neighbour attack, and HF attack(hello flood). Here, the performance metrics are: *i*) end-to-end delay, *ii*) delivery ratio and *iii*) control overhead. In this study the results provide that, compared to the others, the hello flood attack has the most harmful effects on the RPL-based IoT network.

3.2. Intrusion Detection for RPL-based IoT Networks

In the literature, the first lightweight IDS study, SVELTE [41], proposed one of the security architecture for sinkhole and selective forwarding attacks. The study is based on a distributed monitoring architecture, and the proposed solutions have three parts: *i*) "6LoWPAN-mapper" , *ii*) "intrusion-detection", *iii*) "distributed-mini-firewall". Here, the first part, the Root Node requests some information from all other nodes. The information are "rank", "parent and node ID" and "information of neighbours". Here, in order to detect the intrusion, this information is used. Intrusion detection rules are predefined and integrated into the model. These rules are checked for intrusion detection. If suspicious intrusions or illegal attempts are observed, the malicious nodes are deactivated from network topology. Last part of this IDS, the distributed mini-firewall, is a kind of filtering mechanism that filters the malicious data flow. In the study, the proposed model is evaluated in lossless and lossy networks with nodes from 8 to 32. The results of the study show that in a lossless network with eight nodes,

the TPR is calculated as 100%. When the settings are a lossy network with 32 nodes, the TPR is calculated as 65-70%. While the study admitted that there were false positives, no numerical data were disclosed.

Another anomaly-based study based on a game-theoretic model is proposed in [42]. It relies on two parts: *i*) for detection: stochastic game, *ii*) for confirmation: an evolutionary game. Here, this stochastic model for game calculates the general rules of RPL such one zero-sum-game, and a proposed scheme confirms the accuracy value of detection by applying evolutionary methods. Another study based on evolutionary computation [43] use genetic programming in order to detect HF attacks and VN attacks. Here, this study's first goal is to show the effectiveness of "evolutionary-computation-based techniques" in RPL-based IoT networks. When comparing centralised with distributed models, especially for the Hello flood attack, the centralised model performs better than the distributed model.

In [44], an attack detection system based on machine learning approach is explored. Here, this model consists of five parts: *i*) dataset creation and analysis, *ii*) pre-processing, *ii*) feature engineering, *iv*) model development, *v*) training. Contiki Cooja Simulator has used and three different models including different use cases in terms of node numbers and node mobility (20 nodes, 50 nodes, and mobile nodes), are developed: *i*) the benign model *ii*) protocol-specific attack models (PSAM), *iii*) Sensor Network (SN)-Inherited attack models. For PSAM simulations, two network models are developed depending on the uses case scenarios and the number of nodes. For the first scenario, the first use case, there are 20 nodes including 1 malicious node, and for the second use case, 50 nodes including 2 malicious nodes are created. The second is based on state, static or mobile, of the node. Here, the rank attacks for the protocol-specific attack category are simulated. Similar scenarios are created for SN-inherited attacks, with the same use cases as protocol-specific attacks. Here, a wormhole attack is simulated from the SN-inherit attack category. Wireshark [45] is used to analyze the deep network traffic. After data-set creation and data preparation part of the model, an extensive dataset called LIoTN-RPL is created. Here, a machine learning technique called "one-hot encoding" is used to convert categorical data into numerical data. In the classification performance analysis, the "light gradient boosting machine model"

developed by Microsoft for binary classification in 2016 [46], is used. In order to evaluate the proposed approach, the following performance metrics are used: confusion matrix, accuracy, precision, recall, cross-entropy, Cohn's kappa, and Matthews correlation coefficient. The information for the overall performance of the model is given as: 99.7% accuracy, 0.927% MCC, 0.93% Cohn's Kappa, 99% precision, 0.116% cross-entropy, and 99.7% detection rate.

In order to detect version number and the hello flood attack, an anomaly-based IDS model has been proposed by Muller et al. [47]. Here, the KDE ("kernel density estimation") algorithm is used. The number of different types of routing control messages is used as their features. Simulations were carried out using a maximum of 12 nodes and one time interval of 800–1400 seconds. For the hello flood attack, the TPR results reach 90% and 96% for the version number attack.

Another version number attack detection model is proposed in [48]. Here, in RPL-based networks, an approach depend on a distributed monitoring structure [49] is proposed to label malicious nodes and detect VN attacks. The proposed architecture passively observes the network. Two participating node types are used in this study: "monitored-" , "monitoring-nodes". Monitored nodes are the regular nodes that are highly constrained. On the other hand "monitoring nodes" are the higher-order nodes that have higher capabilities. Monitoring nodes can detect abnormal traffic and monitor network flow without affecting their ability to function normally in the network. Hence, these nodes can analyze the data packets coming from the regular nodes. In this model, monitoring nodes have the monitoring ability to monitor only their neighbours. A distributed detection system relays the collected observing data from the monitoring nodes to the root node in intervals of time. Here, to avoid consuming more resources, these nodes use a second network architecture called a "monitoring-network". These two different topologies can work independently of each other. Here, if a monitoring node detects any incremented version number (IVN), this monitoring node can not decide whether this is the result of an attack or not. Here, two algorithms are proposed: the local assessment algorithm and the distributed detection algorithm. The first algorithm detects the intrusion and collects whole information come from the monitoring

nodes. After analysing the information collected, the second algorithm is run to identify the attacker according to this information. In the study, Contiki Cooja Simulator, which creates a 20 nodes network grid topology, is used. In order to perform the proposed model, two metrics are defined: Cov-i (defines how many regular nodes are monitored by exactly i nodes) and Ca-i (measures the percentage of regular nodes monitored by i or more monitors). Additionally, for each simulated scenario, the FPR is calculated. Based on the results, detecting version number attacks with the proposed model performs satisfactorily.

Rank and black hole attacks are among the most effective attacks on disrupting the topology of RPL. In order to detect rank and black hole attacks a Security-, Mobility-, and Trust-based model (SMTrust) is proposed in [50]. Here, a trust-based approach is proposed for the detection and isolation of attacker nodes. In the study, the term "trust" is used for affiliating between two endpoints. Trust metrics, *i*) "success rate", *ii*) "energy level", *iii*) "historical observations", *iv*) "location and link stability", *v*) "mobility", and *vi*) "recommended trust" are selected to make secure RPL in a mobile environment. Here, the model evaluation metrics are "topology stability", "packet loss rate", "throughput", and "power consumption". The proposed model mainly has two parts: "trust formation" and "attack detection". The trust formation part includes *i*) "trust metric identification", *ii*) "trust metric calculation", *iii*) "trust index calculation" *iv*) "trust rating" and *v*) "trust monitoring". The second part includes *i*) detection and *ii*) isolation of malicious nodes. In order to evaluate the trust rating, a fuzzy threshold-based mechanism is used. After evaluating fuzzy judgment, the trust index is defined as trust levels: "No trust", "Poor Trust", "Fair Trust", "Good Trust" and "Full Trust". Here, an attack detection procedure is used to attack detection. For detection of black hole attacks, the success threshold and the trust index are used. Whereas, for the detection of rank attacks, the DIO sequence number and the rank of nodes are checked. Therefore, if a selected parent node is detected as a malicious node or the selected parent node is not trustworthy, it is added to the suspicious parent list. Therefore, only legitimate nodes are allowed to run in an IoT network. In the experiments, the Contiki Cooja Simulator is used and a centralised smart hospital architecture is simulated. Compared to MRHOF, SecTrust, DCTM, and MRTS, the proposed model provides better performance [51, 52].

In [53], an ensemble-based machine learning IDS model called ELNIDS is proposed in order to detect "clone ID attacks","sink-hole attacks", "black-hole attacks", "selective forwarding attacks", "hello flood attacks" and "local repair attack". Here, the classifiers "Boosted Trees", " Bagged Trees", "Subspace Discriminant" and "RUSBoosted Trees" are used. The results show that the boosted trees reach the highest accuracy of 94.5% and the Subspace Discriminant classifier has the lowest accuracy of 77.8%.

Artificial Neural Networks can be effectively used for detecting the intrusion in the RPL-based networks [54]. A recent neural network-based IDS is proposed in [4]. In this study, compared to previous studies, feature extraction is done by focusing not only on the routing layer, but also on the link layer. In this model, the false positive value is reported to be decreased as a result of including the link layer-related features.

In another ML-based IDS, a deep ANN approach has been used in [55] for the detection of DR attacks, HF attacks, and VN attacks. Moreover, they introduce a dataset called IRAD. Here, the model has five hidden layers that use features, especially, packet counts (received and transmitted), and number of DIO/DAO packet in 1000-ms window size. Contiki Cooja simulator [38] and the varying number of nodes from 10 to 1000 are used. The results show that the precision and recall are 94% when the network is under version number attack, 97% under hello flood attack. Another deep learning-based model is developed in order to detect HF attacks in [56]. Here, the Gated Recurrent Unit (GRU)-based deep learning model with a Recurrent-Neural-Network(RNN) approach has been used to classify nodes. Another neural network-based system is proposed in [57] to identify the normal behaviour of the nodes.

Very recently, for smart and RPL-based industries, a "fog-assisted deep-learning-empowered IDS" called "Cu-DNNGRU" is proposed [58]. Here, this proposed model uses the N-BaIoT dataset [59]. This model has three layers: *i*) "edge layer", *ii*) "fog layer", *iii*) "cloud layer". This layer (edge layer) is a kind of basic layer that consists of interconnected industrial IoT devices and runs under the fog layer (FL). The devices (nodes) in the edge layer are similarly connected to the Fog Nodes which are evolved in this fog-layer (FL). Here, this FL, acts like the first security part of the layers, runs under the Cloud Layer

and has some data flow management operations. The process of these operations is as follows: “packet capturing”, “processing”, “packet analysis”, “IDS”, and finally “labeling (malicious or benign)”. By doing so, it manages the data flows from the edge layer. In this layer, by using “Cu-LSTMDNN, Cu-BLSTM, Cu-GRU” classifiers, the data packets are labelled as malicious or benign. After that, the cloud layer, acts as the second security layer, consisting of the root node dismantled unaddressed security concerns for detecting malicious packets. The four performance metrics used in the study are: *i*) “accuracy”, *ii*) “precision”, *iii*) “recall”, and *iv*) “F1-score”. The results show that the evolved IDS Cu-DNNGRU reaches 99.09% precision, 99.21% F1 score, 98,89% recall, and 99.39% accuracy.

There are also rule-based security solutions in the literature. A recent rule-based study proposes an IDS model using “Logistic Regression”, “Gaussian Naive Bayes”, “Artificial Neural Networks”, “Support Vector Machine”, and simulated data in [60]. Here, three attacks against RPL are targeted: *i*) rank attack, *ii*) selective forwarding attack, and *iii*) DoS attack. Here the IEEE-IoT-IDS dataset (is developed by Avast AIC lab.) [61], WSN-DS dataset (developed by Iman et al.) [62], and simulated data are used. Here, the WSN data is simulated using the Contiki Cooja simulator. The WSN data include both benign and malicious activity, and MRHOF has been selected for OF. The WSN data set includes 390,230 samples, and 362,521 samples are normal. Since M2P flow is required for the experiments, the RPL is set to “NO-DOWNWARD-ROUTE”. Multiple attack detection results show that the Selecting Forwarding attack has an 88.4% detection rate and 90.4% accuracy, DoS Attack has an 84.7% detection rate and 85.2% accuracy, the Rank Attack has 89.3% detection rate and 88.4% accuracy.

Transfer-learning-based approaches have also been proposed for intrusion detection in the IoT. A deep transfer learning (DTL) approach called MultiMaximum Mean Discrepancy AE (MMD-AE), based on AutoEncoder (AE) and allows transfer knowledge, is proposed in [63]. Here, although no IoT protocol is targeted, general attacks such as TCP/UDP flooding attacks are targeted, and a labelled dataset is transferred to an unlabelled dataset in accordance with the proposed model. The results clearly indicate that the developed transfer learning model produced better experimental results (Area Under the Curve (AUC) score) than the classic

models. In [1], knowledge has been transferred to detect new attack models and to evolve ID algorithms for new types of devices with different limited resources. Here, when the devices' energy usage is minimized, the detection accuracy is maximized.

Although there are some studies based on evolutionary computation in the literature [1, 43], the current study differs from those by exploring different trade-offs between detection intrusion accuracy and the cost of the developed algorithm in terms of power usage and cost of communication. Therefore, a different intrusion detection architecture is explored here, and communication cost is taken into account for the first time.

4. MULTI OBJECTIVE-BASED INTRUSION DETECTION SYSTEM

This study is aimed at developing a suitable IDS for RPL-based IoT-networks. Therefore, a central ID node is placed at the root-node, which runs our evolved intrusion detection algorithm. In order to analyse the data traffic far from this central ID node, some monitoring nodes in which periodically collect the local data in their neighbourhood and sent to the central ID node are participated in intrusion detection. Although these monitoring nodes enable the central ID node to detect attacks with a more satisfactory performance, it can have an adverse impact on the average lifetime of the network due to collecting their local information and on the communication cost due to their sending such information regularly to the central node. Therefore, the trade-offs between detection accuracy and cost need to be investigated. Hence, this study aims to evolve a lightweight ID model in terms of communication cost and energy consumption while effectively detecting malicious network traffic. Hence, both accuracy and communication cost among IDS nodes are taken into consideration in this study by employing multi-objective evolutionary computation. Here, firstly, the attacks targeted in this study are given. Then, the details of the proposed approach including the features used, the representation of individuals, and the fitness functions are described.

4.1. RPL-specific attacks

The following four specific attacks against RPL are aimed to be detected in this study :

- Decreased Rank Attack (DRA),
- Worst Parent Attack (WPA),
- Increased Version Attack (IVA) and
- Hello Flood Attack (HFA).

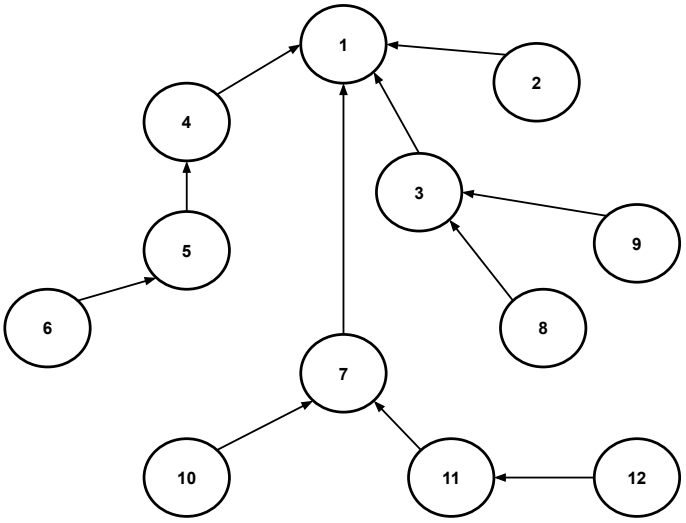


Figure 4.1 An exemplar RPL topology under no attack [2].

An exemplar RPL network topology, which is under no attack, is shown in Figure 4.1. The attacks will be explained as taken this topology as the baseline.

4.1.1. Decreased Rank Attack (DR)

This type of attack, which is illustrated in Figure 4.2, directly targets the DODAG topology and goals to change the topology, and this is one of the most dangerous attacks in this category. Each node advertises its rank value to its neighbour nodes. This value determines the quality of a path between the node and the DODAG root node. The rank value has very important effects, such as creating the most optimal topology in RPL and preventing unnecessary loop formation. Since RPL assumes that all nodes are trustworthy and does not have a security mechanism to protect that, this ranking value could be exploited by attackers. In rank attacks, the attacker manipulates the information of nodes' rank, then a message is sent to neighbours nodes with wrong rank value different from the actual value sent in control messages (DIO). In the decrease-rank attack case, illegitimately, the malicious node decreases the value and tries to attract nodes for them to select itself as the parent node. Hence, it is generally carried out as the first step of subsequent attacks, such as dropping attacks.

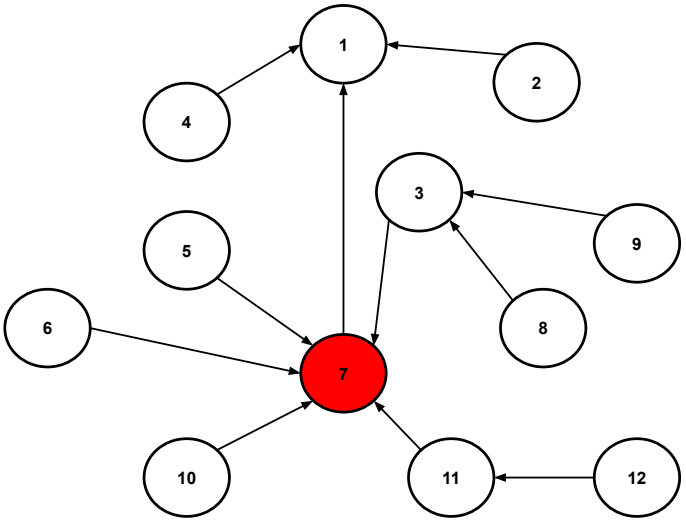


Figure 4.2 The topology under decreased rank attack [2].

4.1.2. Increased Version Attack (IV)

Version numbers are specified in the DIO messages in the RPL. A change in this number, which is only carried out by the root node, triggers DODAG global repair. Since the global repair reconstructs the DODAG topology, it consumes a lot of resources and puts a lot of overhead on the network due to excessive message traffic. In increased-version attack, malicious node(s) illegally alters this version number into given DIO control message and allows the DIO message to be distributed in this way. Therefore, re-creating the DODAG structure again and again causes unnecessary use of limited resources. As a result of the attack, performance and efficiency loss are experienced in the network, packet losses, and unwanted routing loops may occur in the network. Node 12 shown in red in Figure 4.3 [6] illegally increases the version number carried by DIO messages and allows it to be distributed in this way. It creates excessive message traffic by causing the DODAG global repair mechanism to work repeatedly.

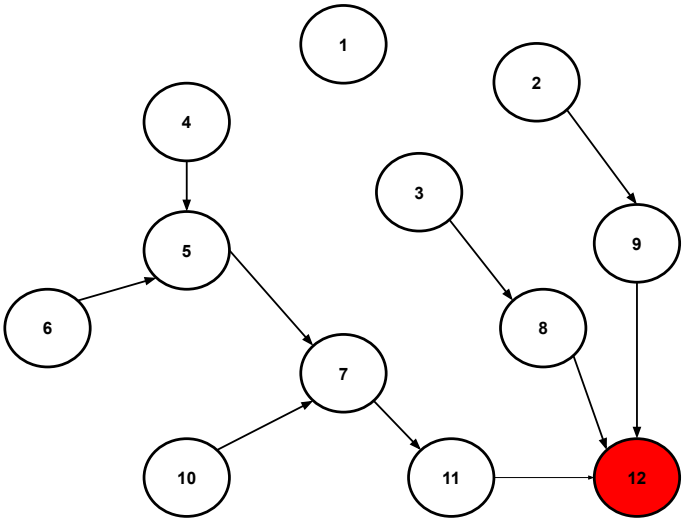


Figure 4.3 The topology under increased version attack [6].

4.1.3. Hello Flood Attack (HF)

In the DODAG structure, any candidate node that wants to join this DODAG, broadcasts DIS control messages to notify its neighbouring nodes. In RFC 6550 [25], there are no defined time intervals for DIS messages. Therefore, DIS messages are triggered as defined in the application layer, hence it may differ from one application to another. Hence, a new candidate node broadcasts DIS (HELLO) messages and waits for a response from neighbouring nodes in its transmission range. If a neighbouring node sends a DIO (HELLO response) message in response to the DIS message, the new candidate node responds with a DAO message and joins the network. Here, the attacker node behaves like as the new candidate device for joining the DODAG topology and continuously broadcasts DIS messages to neighbour nodes to force them responds with DIO messages, with more power than the root node at a certain time frequency. In this way, the attacker node forces its neighbour nodes not only to respond with a DIO message, but also to reset their trickle timers. In this attack scenario, which is illustrated in Figure 4.4, the main aim is to generate heavy network traffic and keep existing paths busy. In Figure 4.4 [7], the attacker node broadcasts DIS messages with more power than the root node (left side of this figure). Directions of the responses sent to this fake DIS control messages are illustrated at the right side of this figure. Here, the nodes

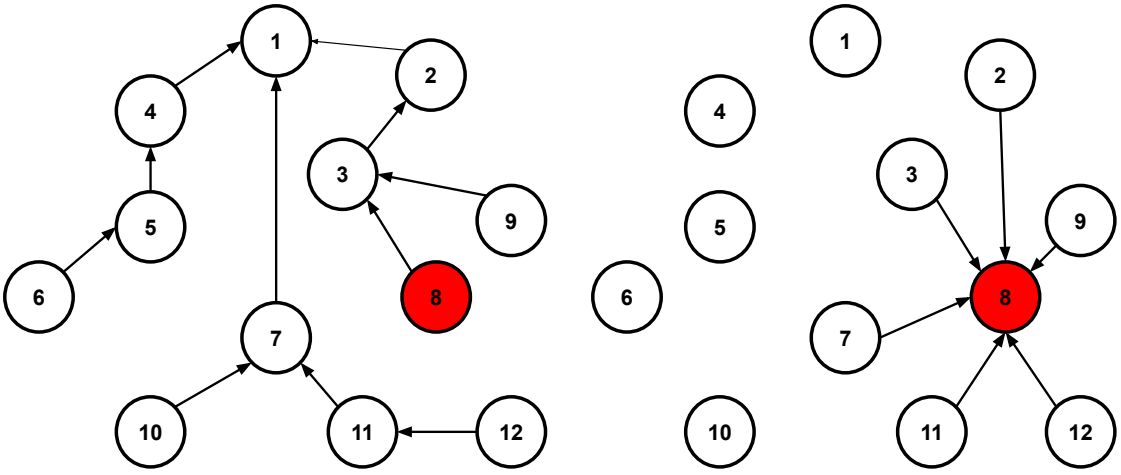


Figure 4.4 The topology under hello flood attack [7].

assume that the attacker node is a new legitimate node. Thus, the resources of the network are unnecessarily consumed.

4.1.4. Worst Parent Attack (WP)

Worst Parent Attack, which is illustrated in Figure 4.5 [2], is a kind of rank attack and goals to change the DODAG network topology. As stated above, the quality of paths to the root node is determined according to rank value. This rank value is calculated by all nodes. A DODAG topology has very important mechanisms such as maintaining optimal paths and preventing loop formation, and the continuity of these mechanisms is provided by the calculated rank values. In this type of attack, the malicious node sents any different rank score than their current rank values in the DIS values. By sending higher rank values, differently than in decreased rank attack, it causes non-optimal paths to be built.

In this type of attack, the goal is to choose the worst parent node (this node is the attacker node as illustrated with number 3 and red color in Figure 4.5) to change the network topology according to the objective function. Since the node chooses a path that it should not normally choose according to the objective function (OF), the performance of the network decreases and the DODAG structure is adversely affected by this process. Since neighbouring nodes

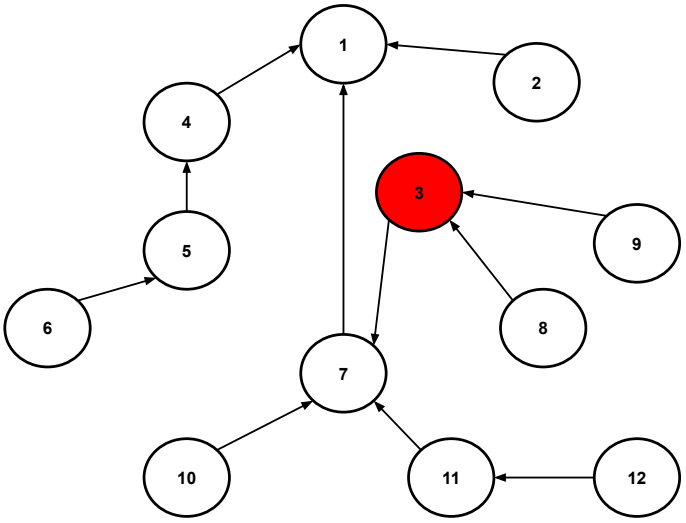


Figure 4.5 The topology under worst parent attack [2].

can only see the rank value of a node, they cannot decide what its true rank should be. Therefore, monitoring and detecting worst parent attack is very difficult.

4.2. Intrusion Detection Architecture

In order to create an effective and highly accurate IDS, data collected only from the root node may be insufficient. Therefore, in order to benefit from the data collected from other nodes (monitoring nodes) and increase the efficiency of the proposed IDS, the data was collected not only from the root node but also from monitoring nodes. Here, the main purpose is to control the entire network with a global ID node, it is also to determine how the monitoring nodes affect the performance of the evolved IDS. Here, a grid topology-based network model of 27 sensor nodes, 1 of which is the root node, and 3 malicious nodes is used in this study. The nodes in the model are grouped as the first group nodes (close nodes) and the second group nodes (far nodes), as illustrated in Figure 4.6. Here root-node is shown in blue. The first group of nodes shown in yellow are the nodes 1-3 hops close to the root node. The second group nodes shown in purple are the nodes 4-6 hops far from this root node. According to this grouping, the feature values collected from the nodes belonging to the relevant group were brought together just before the simulation run, and their average

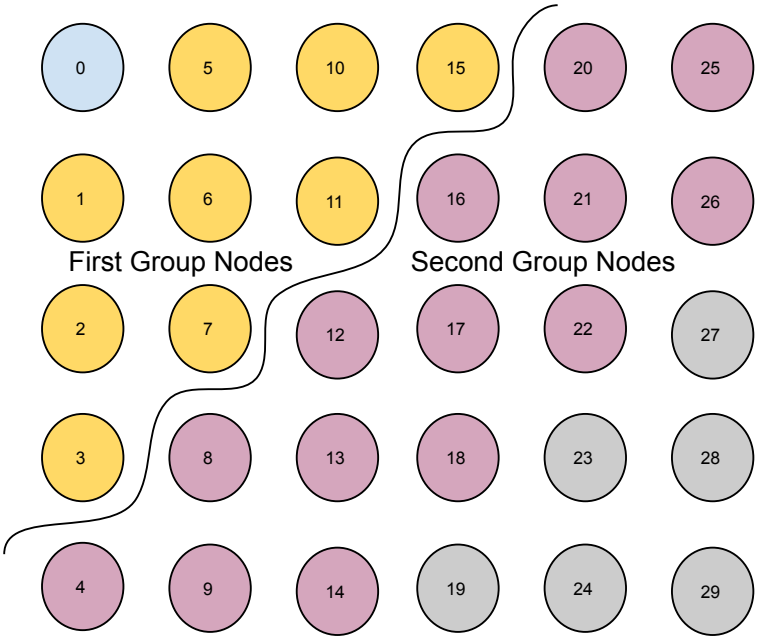


Figure 4.6 Grouping of nodes according to hop numbers.

values were calculated. For this, a node was selected from each node group and the average values of the data of these selected nodes were calculated.

4.3. Evolving Intrusion Detection Algorithms

The proposed approach conceptual scheme is shown in Figure 4.7. First, to construct the training and testing data sets, the pcap files that capture the network traffic are extracted by running Contiki Cooja Simulation[38]. As will be explained in Section 5.1., the simulation was run both in attack mode and without attack mode (benign mode), and the data were labelled malicious (the label is 1) and benign (the label is 0) using the generated pcap and log files. Then, the features collected from monitoring nodes and the root node are extracted using these files.

After running simulation files and obtaining the pcap and log files, feature extraction is carried out. Then the candidate solutions based on these features and some operators as explained in the subsequent sections are generated randomly. The GP algorithm is run and tries to evolve better solutions, in other words, intrusion detection (ID) programs in

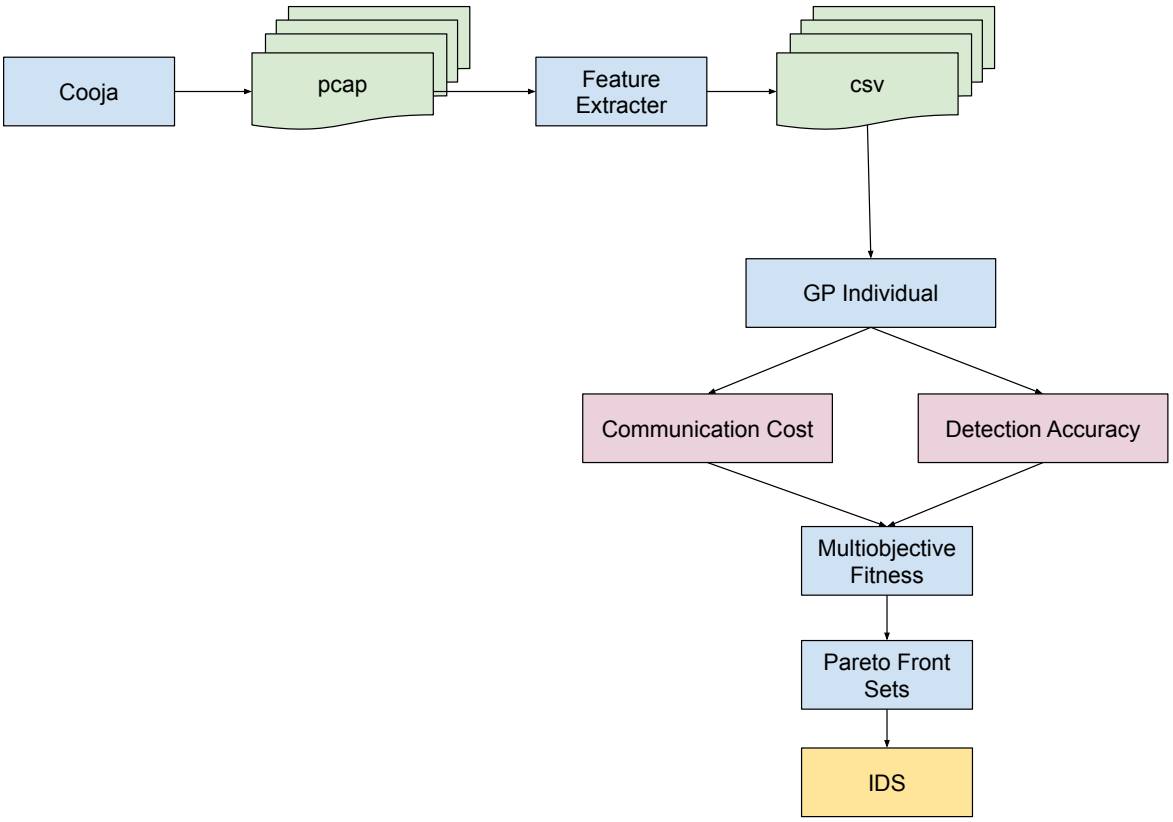


Figure 4.7 The conceptual scheme of the proposed approach.

each generation. In order to evaluate the evolved solutions, two fitness functions are used: *detection accuracy* and *communication cost* (i.e. number of features). Finally, GP outputs the best individual according to its fitness value. This output is our *intrusion detection program*. In the following, the details of the evolution process, including the features and operations used for representation of the candidate solution, the GP algorithm, and the fitness functions, are given in detail.

4.3.1. The Features

Choosing appropriate features in solving of complex problems is vital for training machine learning algorithms. In order to develop security mechanisms and detect attacks against RPL-based IoT networks, selecting features that could help to distinguish malicious traffic

from benign one, is very critical and important. On the other hand, the use of more features than necessary might negatively affect the training of the machine learning algorithm or increase its training time [4]. Since each attack type might require different features besides generic features such as the number of data packets, all features that is considered to be useful for intrusion detection are included in this study. Hence, in this study, all the features proposed in a recent study [1] shown in Table 4.1 are used.

Table 4.1 The feature set [1].

No	Feature	Description
1	DataPacketCount	The total count of data packets
2	CountDIO	The total count of DIO control packets
3	CountDIS	The total count of DIS control packets
4	CountDAO	The total count of DAO control packets
5	CountDAOACK	The total count of DAO-ACK control packets
6	MaxVersion	The versions' maximum value
7	MinVersion	The versions' minimum value
8	AvgVersion	The versions's average value
9	Maxrank	The ranks' maximum value
10	MinRank	The ranks' minimum value
11	AvgRank	The ranks' average value
12	MaxIntervalMin	The maximum value of interval min. for DIO
13	MinIntervalMin	The minimum value of interval min. for DIO
14	AvgIntervalMin	The average value of interval min. for DIO
15	MaxIntervalDoub	The greatest value for DIO interval doublings
16	MinIntervalDoub	The lowest value for DIO interval doublings
17	AvgIntervalDoub	The average value for DIO interval doublings
18	MaxRplIns	The rpl instances' maximum value
19	MinRplIns	The rpl instances' minimum value
20	AvgRplIns	The rpl instances' average value
21	MaxTimeBtwDATA	The greatest time value
22	MinTimeBtwDATA	The lowest time value
23	AvgTimeBtwDATA	The average time for data packets
24	MaxTimeBtwDIO	The greatest time value for DIO messages
25	MinTimeBtwDIO	The lowest time value between DIO messages
26	AvgTimeBtwDIO	The average time between DIO messages
27	MaxTimeBtwDIS	The greatest time value for DIS messages
28	MinTimeBtwDIS	The lowest time value for DIS messages
29	AvgTimeBtwDIS	The average time for DIS messages
30	MaxTimeBtwDAO	The greatest time value for DAO messages
31	MinTimeBtwDAO	The lowest time value for DAO messages
32	AvgTimeBtwDAO	The average time between DAO messages
33	MaxTimeBtwDAOACK	The greatest time value for DAO-ACK messages
34	MinTimeBtwDAOACK	The lowest time value for DAO-ACK messages
35	AvgTimeBtwDAOACK	The average time for DAO-ACK messages

4.3.2. The Representation

As stated above, the solutions are represented by *GP trees* in genetic programming. Therefore, a candidate solution, which is an intrusion detection (ID) program in this study, is represented by a GP tree and essentially returns a conditional (if) statement to detect the attack and raise an alarm. In this tree representation, the terminal nodes, i.e. leaf nodes in the GP tree, are the features collected by the nodes. As mentioned above, in this study we used 35 different features [1] that are extracted from the RPL control and data packets. Traffic flows are used to build these features. To do that, the flows are first windowed within the specific time intervals by both the root and the monitoring nodes. Here, the optimum interval time of the window is found to be 60 s. The windowed feature data is then collected by the monitoring nodes and aggregated at the root node and provided to the GP tree as input data. In addition to RPL-related features, randomly generated numbers are also assigned to leaf nodes to enable a more effective search by GP individuals. The non-terminal nodes correspond to arithmetic, comparison, and logical operators given in Table 4.2. It is worth stating that this GP tree's root node can be either a comparison or a logical operator, since it returns a *Boolean* value. An example of a GP tree that represents a *candidate solution* is given in Figure 4.8, and the *ID program* corresponding to this tree is given in Algorithm 2.

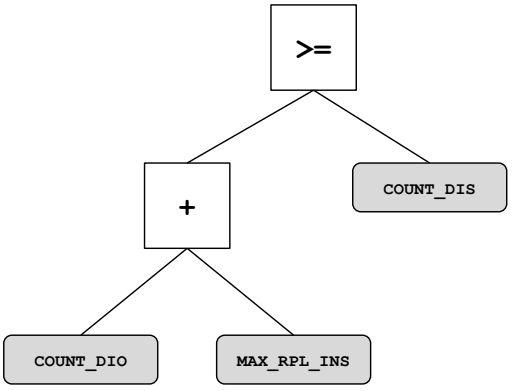


Figure 4.8 An example GP tree.

Eventually, the evolution process should come to an end, for instance after the specified steps of generations. In this study, GP-algorithm is run for 1000 generations. A total of 100 individuals are kept in each generation. The other GP parameters used in this study are given

Algorithm 2: A conditional tree algorithm of GP.

if *((COUNT_DIO + MAX_RPL_INS) >= COUNT_DIS)* **then**
alert(attack)
stop

in Table 4.2. For the GP implementation, "Evolutionary Computing Toolkit (ECJ)" [64], a kind of java-based toolkit, is used. Please note that, Table 4.2 does not show other parameters that are normal settings for ECJ. Fitness functions are used to evaluate each individual as explained in the following sections.

Table 4.2 The values for the parameters of GP.

Parameter	**Value**
Non-terminals	Non-terminals in [1]
Generations	1,000
Population Size	100
Elite Fraction	10%
Crossover probability	0.9
Mutation probability	0.1
Selection strategy	Tournament with the size of 7
Max. depth of the tree	20
Terminals	features in [1] and rnd(0,1)

4.3.3. Fitness Function

The fitness functions are used to assign a fitnes value to individuals. This value determines which are the most effective solutions in solving the problem at hand. In GP, as stated earlier, in order to create a new generation individuals are selected from the existing population according to their fitness value and then modifying these individuals via genetic operators such as recombination, mutation. Following that, the algorithm is iterated again with the new set of candidate solutions. The GP algorithm terminates when a population or algorithm reaches a specified number of generations or a desired satisfactory fitness level [65, 66]. In this study, in order to evaluate evolved algorithms, two fitness functions are employed: *detection accuracy* and *communication cost (number of features)*.

4.3.3.1. Detection Accuracy This metric is used to measure the effectiveness of the evolved ID programs. It can be said that the higher the attack detection accuracy of the ID program, the more effective it is. The formula of this metric is given in Equation-1 :

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

Here, TP corresponds to true positives, FP corresponds to false positives, TN corresponds to true negatives, and FN corresponds to false negatives.

4.3.3.2. Communication Cost When developing ID programs, communication cost, another objective is of critical importance. Because if communication costs are not minimized on a network composed of resource-limited devices, the network can die much faster. As communication cost increases, the amount of energy and resources consumed by the network also increases. However, since our IoT network is already resource-constrained, our aim is to propose a lightweight solution as much as possible in this study. Therefore, communication cost is also taken into account as another objective.

The number of features employed in the ID model is of very high importance in determining to what extent the model leads to *additional cost* in terms of the memory and battery usage of the monitoring-nodes and also communication load in any networks. Moreover, the high number of features might result in fragmentation and the increase in the number of packets [4]. As shown in the results of the preliminary experiment given in Figure 4.9, the overall power consumption of the nodes increases linearly with increasing data packets. Therefore, in addition to the detection accuracy, the number of distinct features employed in the intrusion detection algorithm is taken as the second objective which should be optimized simultaneously for ensuring an efficient and high-performance ID model. To optimize these objectives simultaneously, as stated earlier, the NSGA-II's selection and survival strategies [5] are integrated into the GP algorithm. Hence, each individual that represents a candidate program is evaluated according to the *Pareto dominance*.

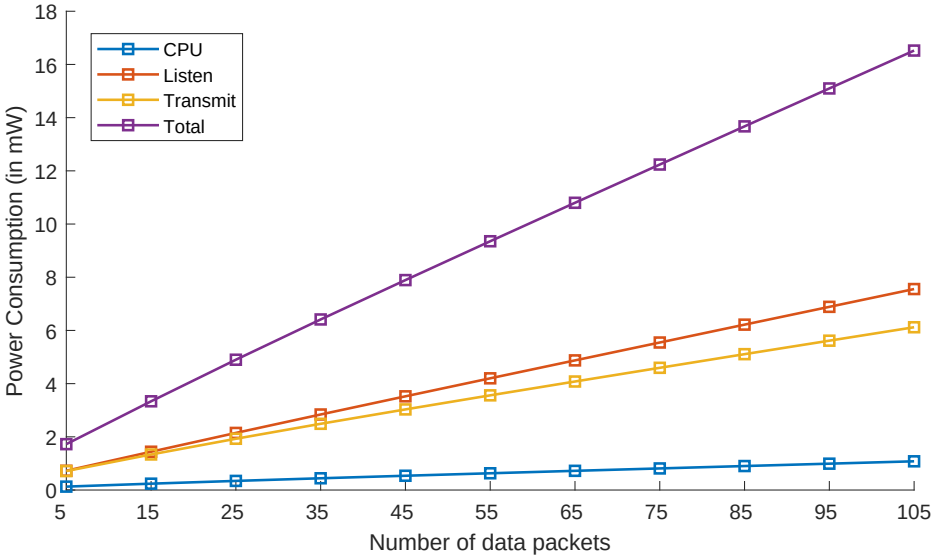


Figure 4.9 The change in average power consumption with varying number of packets.

5. EVALUATION OF THE PROPOSED APPROACH

Experimental settings and information about the simulation environment are given in detail in this section, and simulation parameters and performance metrics are explained. Finally, the experimental results are thoroughly presented and discussed.

5.1. Settings for Experiments

5.1.1. Simulation Environment

In this study, a java based simulator, Cooja Simulation [67] running on the Contiki OS [68] is used to simulate RPL. Contiki OS is a "state-of-the-art open-source" OS primarily targeted at sensor networks and other embedded network domains [69]. Cooja is a Java-based simulation that has flexible and ubiquitous network capability and can emulate various nodes such as the Z1 mote [70], the Sky mote [71], and the AVR mote [72]. Thanks to Cooja, researchers can debug their software, observe or test the behaviour of the systems they create, and, with Cooja, have the opportunity to control code without using hardware. Cooja allows developers to simulate different RPL features. In order to simulate the most realistic RPL mechanism, the RPL protocol, called ContikiRPL in Contiki OS, can be implemented in Cooja. The RPL attack framework [73] is integrated into the Cooja simulation to simulate attack environments.

In the experiments, a grid topology, shown in Figure 5.1, is used with 30 nodes, including the root node. Among them, three nodes (10% of the total nodes) are set as attacker nodes, where their positions are *randomly* chosen. The nodes in the topology are positioned in the network so that each node is 20 m away from another node with a maximum distance (transmission range) of 25 m between each node, so that the nodes can communicate with their neighbour nodes. The arrows in Figure 5.1 represent the preferred parent of the child nodes in DODAG in a benign environment.

Table 5.1 Parameters for Simulation.

Parameter	Value
Tool	Contiki 2.7 Cooja Simulator
Mote type	Z1 Motes
Simulation time	180 mins
Sender Nodes	26
Malicious Nodes	3
Root Node	1
Radio medium	UDGM : Distance Loss
Transmission range	25m
Positioning	Grid Positioning
Framework	RPL Attacks Framework

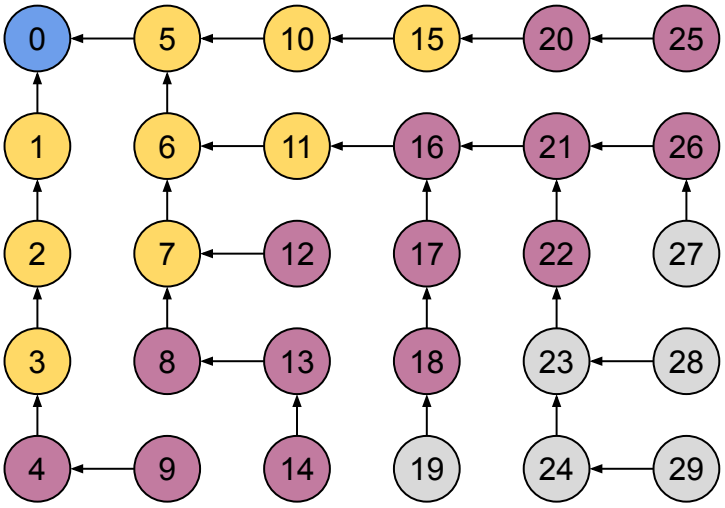


Figure 5.1 Simulation network.

Supported by Cooja for the emulated nodes, the *Contiki 0perating System* [67] (version 2.7) that also involves the implementation of RPL is used. Zolertia (Z1) [70] platform is chosen as the mote type for the nodes. By adopting this experimental environment and the settings, the performance of the proposed evolutionary-based IDS against these four targeted routing attacks has been thoroughly evaluated; *WP, HF, IV,* and *DR*. To do that, eight different scenarios are generated, and the learning is repeated *10 times for each scenario* that is individually simulated for *three hours*. The main motivation of the scenario-based evaluation is to thoroughly discuss how well the evolved ID program can detect when monitoring or attacker nodes are re-positioned after the learning step. Therefore, these scenarios differ

from each other in terms of the location of the monitoring and malicious nodes in training and testing simulation environment. These scenarios are outlined in Table 5.2. This table shows that the monitoring nodes in the network are grouped according to their proximity to *network's root* (node 0 in Figure 5.1). The monitoring nodes are either placed *closely* to root node (1-3 hops away from the root node, represented with yellow in Figure 5.1) or *far* (4-6 hops away, represented with purple in Figure 5.1) from the root node. It is worth pointing out here that each of the monitoring nodes is *randomly* chosen from a *different hop level* (that is, a single monitoring node is chosen per hop and the average of these nodes' data is used in the experiments). Hence, in each scenario 10% the nodes are responsible for monitoring. In Scenario-1, 3, 5 and 7, the attacker nodes were randomly selected and remained the same location in training and test environments. In addition, in order to see the effect of attackers' positions, in some scenarios, the attackers are placed randomly, but differently from its corresponding training setting.

Table 5.2 Position settings of the monitoring ID nodes and the attacker nodes used in the experiments.

Scenario	**Location of monitoring ID nodes**		**Location of attackers**
	In training	**In testing**	**in training and testing**
S1	close	close	same
S2	close	close	different
S3	far	far	same
S4	far	far	different
S5	close	far	same
S6	close	far	different
S7	far	close	same
S8	far	close	different

5.1.2. Simulation Environment with Mobile Attackers

The same experiments are repeated with mobile attackers. Even though RPL is not designed for mobility, most of the real-world scenarios include mobile attackers. To simulate real-world scenarios, we expanded our study by adding mobility characteristics to attacker nodes. Here, our aim is to evaluate the performance of our proposed ID program if the

malicious nodes have mobility and to show whether the proposed ID program still produces effective results.

A plug-in called "Mobility" [74] has been added to the Contiki Cooja simulator in order to make attacker nodes mobile and the simulator has been provided with a *mobility tool* [74] that is designed for simulating and testing MANET protocols. With this tool, the mobility in the experiments can be stopped at any time. Therefore, mobility can wait still for any second and then start running without reinstalling or generating new movement data [75]. This tool has the ability to read the relevant parameters from a file in order to make nodes mobile. The mobility parameters file is created by using *BonnMotion* [76]. Thanks to this tool, environments belonging to different models such as *"Boundless Simulation Area Model", "Chain Model", "Column Mobility ", "Disaster Area", "Random Street"*, and *"Random Walk Model"* can be created [76]. Here, a mobility model based on random speeds and random direction, called *random walk model*, which is most commonly preferred to simulate mobility in MANETs, is used in this mobility included experiments. In this model, nodes move from the their current location to the new location randomly. The new speed or location can be selected from predefined ranges consisting of [$speedmin$,$speedmax$] and [$0, 2\pi$]. The mobility of each node is defined with the following information [76]:

- "the number of node",
- "the time of simulation (in seconds)",
- "X and Y coordinates of the nodes' position".

Contiki Cooja Mobility tool uses this interval-by-line based file named positions.dat and takes the information about the mobility of the nodes from there as a parameter. This parameter file was first created in the BonnMotion environment in accordance with the random walk model and converted into a format that the Cooja Mobility tool can use. An example of the positions.dat file including mobility parameters are shown as follows:

$\#node$	$time(s)$	x	y
1	0.0	20	10
1	1.0	20	20
1	2.0	0	0
1	3.0	0	25
1	4.0	10	15
1	5.0	10	20

The first, second, third and fourth columns show respectively: the *"node number"*, the *"time-stamp"*, the *"x-coordinate"* and finally the *"y-coordinate"*[74].

The same scenarios for the location of attacker and monitoring nodes given in Table 5.2 are used. For four attack types, *ten simulations* for each scenario were run and the feature values of these simulations were *averaged*. Here, the attacker nodes are selected *randomly* and the mobility pattern of these nodes are determined based on the random walk model. Here, for Scenario-1 the nodes 6, 13, and 23 are randomly selected as attacker nodes as illustrated in Figure 5.2. The default parameters of speed (4 m/s) and directions ($[0, 2\pi$]) for the random walk model are used. Other than the coordinates of nodes due to their mobility, the same parameters given in Table 5.1 are used. Please note that in these experiments, nodes other than the attacker do not have mobility features.

The mobility areas of attacker nodes are shown in Figure 5.2. The point where the attackers start to move is the current location shown in Figure 4.6. Then, after the start of the simulation, the attacker nodes were allowed to move according to the random walk model so that they remain within the movement circle (they are limited to neighbouring nodes at a distance of 1-hop count). This mobility was continued until the end of the simulation.

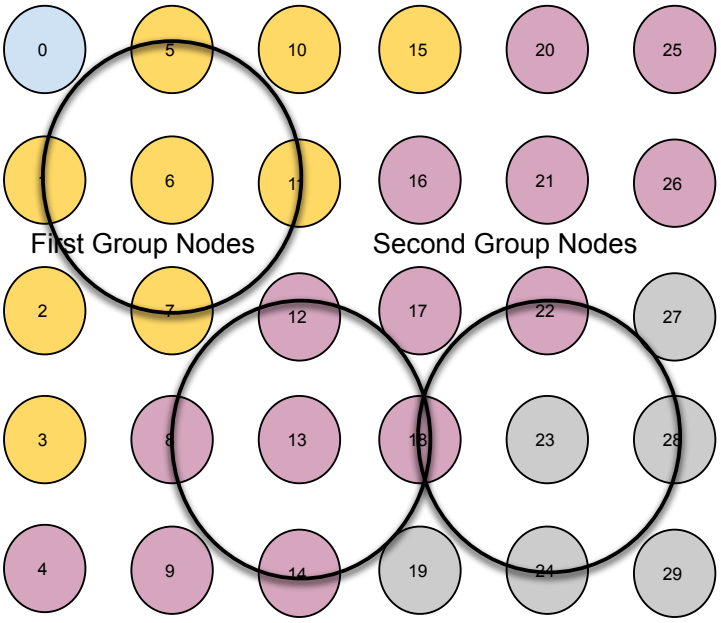


Figure 5.2 Mobility area of attacker nodes for Scenario-1.

5.1.3. Performance Metrics

In this study, we have considered the Pareto front set that represents the objectives of the Pareto dominant individuals found for each simulated scenario in order to evaluate the our evolved IDS's performance. In this study, eight different scenarios were studied for each type of attack. The GP algorithm was run 10 times for each attack type in each scenario. Here, accuracy (ACCR) and number of features (NoF) are used as performance metrics in order to show and compare the results.

- **ACCURACY:** The equation for calculatin accuracy of an ID program is given in *Equation-2*. Here, (TP) or (TN) prove that the predictions are correct. (FP) and (FN) similarly show that these predictions are incorrect [77].

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{2}$$

- **NUMBER OF FEATURES (NoF):** When creating Pareto-dominant individuals, the GP finds the optimum solutions according to the objective functions. In a GP tree, some features which corresponds to different GP nodes can be repeated. Therefore, this study considered the distinct number of features to calculate the NoF. For instance, for the GP tree formed as *((fA + fB) < (fA X fB))*, the number of features is calculated as 3. These features are fA, fB, and fC. Even though there are two nodes that have fA, it is counted as 1 unique feature. Similarly, the Number of Features value for the GP tree formed as *((fA + fB) < ((fC X fD) < fA))* is equal to 4 and these features are fA, fB, fC, and fD.

5.2. Experimental Results

This section of the study explores the performance of proposed IDS under the targeting attacks, namely worst parent, hello flood, increased version, and decreased rank. Here, for each attack type, all eight scenarios regarding the placements of monitoring nodes and attacker nodes listed in 5.2 are evaluated. In these scenarios, both attacker nodes and monitoring nodes are placed randomly. Please note that, for monitoring the most realistic packet flow of the network (data or control packets), the attackers were placed randomly. GP algorithm is run ten times for each case. Hence, in total, we performed 4 (attack type) x 8 (scenarios) x 10 (GP runs) x 2 (training & testing) x 2 (attackers are mobile & not mobile) = 1280 experiments. Then the Pareto Front sets, in other words the Pareto dominant individuals are extracted for evaluating the performance of IDS for each scenario. Results for each attack type are first given and general discussions are then presented.

5.2.1. The Performance of IDS on Detecting Worst Parent Attack

The best performances obtained with respect to each objective, namely accuracy (ACCR) and number of features (NoF) are presented for each scenario (S) in Table 5.3. The trade-offs (TO) between ACCR and NoF for each GP run for an exemplar scenario (S1) is given Table 5.4. As shown in the results, while the highest accuracy rate (94.5%) for scenario 1 (S1) is

achieved by using eight features, it drops to 85% when only one feature is used (1 (0.850)). Moreover, for each scenario we can obtain high accuracy for this attack type. Hence, the location of monitoring or attacker nodes does not affect the accuracy results considerably.

Table 5.3 The best performances with respect to accuracy (ACCR) and number of features (NoF) obtained in the Pareto front set for WPA.

Objective	**S1**	**S2**	**S3**	**S4**	**S5**	**S6**	**S7**	**S8**
ACCR	0.945	0.940	0.950	0.940	0.935	0.935	0.930	0.925
NoF	(8)	(7)	(11)	(8)	(8)	(12)	(9)	(8)
NoF	1	1	1	1	1	1	1	1
ACCR	(0.850)	(0.840)	(0.845)	(0.840)	(0.835)	(0.830)	(0.835)	(0.825)

Table 5.4 The trade-offs between objectives obtained in the Pareto set of each GP run for Worst Parent Attack.

GP Run	**Objective**	**TO1**	**TO2**	**TO3**	**TO4**	**TO5**	**TO6**	**TO7**	**TO8**
1	**ACCR**	0.850	0.850	0.850	0.850	0.855	0.855	0.855	-
	NoF	1	1	1	1	2	2	2	-
2	**ACCR**	0.870	0.870	0.870	0.870	0.870	0.870	-	-
	NoF	3	3	3	3	3	3	-	-
3	**ACCR**	0.870	0.870	0.870	0.872	0.872	0.872	0.873	-
	NoF	1	1	1	2	2	2	3	-
4	**ACCR**	0.870	0.870	0.870	0.870	0.870	0.870	-	-
	NoF	3	3	3	3	3	3	-	-
5	**ACCR**	0.940	0.940	0.945	0.945	0.945	0.945	0.945	0.945
	NoF	7	7	8	8	8	8	8	8
6	**ACCR**	0.870	0.870	0.870	0.870	0.870	-	-	-
	NoF	3	3	3	3	3	-	-	-
7	**ACCR**	0.905	0.905	0.905	0.930	0.937	0.937	0.937	0.937
	NoF	1	1	1	3	4	4	4	4
8	**ACCR**	0.870	0.870	0.870	0.870	0.870	0.935	0.935	0.940
	NoF	3	3	3	3	3	6	6	7
9	**ACCR**	0.870	0.870	0.870	0.870	0.870	0.935	0.940	0.940
	NoF	3	3	3	3	3	6	7	7
10	**ACCR**	0.940	0.940	0.945	0.945	0.945	0.945	0.945	-
	NoF	7	7	8	8	8	8	8	-

5.2.2. The Performance of IDS on Detecting Hello Flood Attack

The best performances obtained with respect to each objective, namely accuracy (ACCR) and number of features (NoF) are presented for each scenario (S) in Table 5.5. The trade-offs (TO) between ACCR and NoF for each GP run for an exemplar scenario (S1) is given Table 5.6. As shown in the results, while the highest accuracy rate (95%) for scenario 1 (S1) is achieved by using thirteen features, it drops to 86.5% when only one feature is used (1 (0.865)).

When comparing S1 with S7, and S2 with S8, a decrease of 2.5% is seen in the accuracy rates. Here, in the four scenarios' designs, the data collected from close node group are used in testing environment. On the other hand, in the training environment, in S1 and S2 data collected from the close node group and the location of the attackers are both the "same". In S7 and S8 data collected the nodes far from the root node and the location of attackers are both "different". In the light of the designs of the scenarios here, it can be said that the slight decrease is caused by the distance of the nodes collected in the training environment to the root node. In other words, it can be said that the accuracy performance of the ID program decreases as the node that is "close" in the test environment, the attacker position remains the same in the train and test environment, as it moves far from the root node in the train environment. In addition, S1, S2, S6 and S7 results show that more features are used for better performance than other scenarios in Table 5.5. In the S1 and S2 scenarios' designs, the data collected from the close node group are used both in training and testing environment. However, in S6 design, the data collected from the close node group in training environment and far node group for testing environment. In the S7 design, this data collected from far nodes group for learning and the data collected from the close node group for testing environment. These results can be evaluated that for this attack type, both attacker positions and monitoring nodes positions (even if they change in learning and test environment) affect the number of feature usage very slightly. Considering the number of features usage in all scenario results, it can be said that it is necessary to use approximately 10 (the average value

for NoF is 10,375) features for a good ID program performance. For each scenario, we can still obtain high accuracy for this attack type.

Table 5.5 The best performances with respect to accuracy (ACCR) and number of features (NoF) obtained in the Pareto front set for HFA.

Objective	**S1**	**S2**	**S3**	**S4**	**S5**	**S6**	**S7**	**S8**
ACCR	0.950	0.945	0.945	0.940	0.930	0.925	0.925	0.920
NoF	(13)	(12)	(10)	(9)	(9)	(11)	(12)	(7)
NoF	1	1	1	1	1	1	1	1
ACCR	(0.865)	(0.855)	(0.860)	(0.855)	(0.845)	(0.840)	(0.830)	(0.830)

Table 5.6 The trade-offs between objectives obtained in the Pareto set of each GP run for Hello Flood Attack.

GP Run	**Objective**	**TO1**	**TO2**	**TO3**	**TO4**	**TO5**	**TO6**	**TO7**	**TO8**
1	**ACCR**	0.865	0.865	0.865	0.870	0.870	-	-	-
	NoF	1	1	1	2	2	-	-	-
2	**ACCR**	0.935	0.935	0.935	0.940	0.940	0.950	0.950	0.950
	NoF	10	10	10	12	12	13	13	13
3	**ACCR**	0.865	0.865	0.865	0.870	0.870	-	-	-
	NoF	1	1	1	2	2	-	-	-
4	**ACCR**	0.950	0.950	0.950	0.950	-	-	-	-
	NoF	13	13	13	13	-	-	-	-
5	**ACCR**	0.945	0.945	0.950	0.950	0.950	-	-	-
	NoF	12	12	13	13	13	-	-	-
6	**ACCR**	0.865	0.865	0.870	0.870	0.870	0.870	-	-
	NoF	1	1	2	2	2	2	-	-
7	**ACCR**	0.945	0.945	0.950	0.950	0.950	0.950	0.950	0.950
	NoF	12	12	13	13	13	13	13	13
8	**ACCR**	0.865	0.865	0.870	0.870	0.870	0.870	-	-
	NoF	1	1	2	2	2	2	-	-
9	**ACCR**	0.865	0.870	0.890	0.915	0.935	0.940	0.950	0.950
	NoF	1	2	4	6	10	11	13	13
10	**ACCR**	0.890	0.890	0.915	0.915	0.940	0.940	-	-
	NoF	4	4	6	6	11	11	-	-

5.2.3. The Performance of IDS on Detecting Increased Version Attack

The best performances obtained with respect to each objective, namely accuracy (ACCR) and number of features (NoF) are presented for each scenario (S) in Table 5.7. The trade-offs (TO) between ACCR and NoF for each GP run for an exemplar scenario (S1) is given Table 5.8. As shown in the results, while the highest accuracy rate (95.5%) for scenario 1 (S1) is achieved by using twelve features, it drops to 87% when only one feature is used (1 (0.870)).

Interestingly, the number of features used in S1, S2 and S6 is twelve for accuracy performance of 94.5% and above. Here, S1, S2, S5 and S7 results show that more features are used for better performance than other scenarios in Table 5.7. In the S1 and S2 scenarios' designs, the data collected from the close nodes group are used both in the training and test environment. However, in S5 design, all data collected from the close nodes group for training environment and far nodes group for testing environment. In the S7 design, the data collected from the far nodes group for learning environment and the data collected from the close nodes group for testing environment. These results can be evaluated that for this attack type, both attacker positions and monitoring nodes positions (even if they change in learning and test environment) affect the number of feature usage very slightly. These results are also similar to the hello flood attack results described just before. Considering the number of features usage in all scenario results, it can be said that it is necessary to use approximately 9-10 (the average value for NoF is 9,875) features for a good ID program performance. In addition, for scenarios S1, S2, S5, S6, S7 and S8, attacker positions do not cause a serious change in the number of features (as well as accuracy values) used. Moreover, considering the NoF for scenarios S3 and S4, it is seen that the NoF values calculated for these two scenarios are below the average value (9) for the whole scenario (NoF=8 for S3 and NoF=6 for S4). This is because, in both scenarios, the collected data (for both learning and testing) may be collected from nodes far from the root node.

Table 5.7 The best performances with respect to accuracy (ACCR) and number of features (NoF) obtained in the Pareto front set for IVA.

Objective	S1	S2	S3	S4	S5	S6	S7	S8
ACCR	0.955	0.950	0.935	0.925	0.940	0.945	0.935	0.910
NoF	(12)	(12)	(8)	(6)	(10)	(12)	(10)	(9)
NoF	1	1	1	1	1	1	1	1
ACCR	(0.870)	(0.870)	(0.860)	(0.850)	(0.830)	(0.825)	(0.835)	(0.810)

Table 5.8 The trade-offs between objectives obtained in the Pareto set of each GP run for Increased Version Attack.

GP Run	Objective	TO1	TO2	TO3	TO4	TO5	TO6	TO7	TO8
1	**ACCR**	0.870	0.870	0.872	0.872	0.872	0.873	0.873	-
	NoF	1	1	2	2	2	3	3	-
2	**ACCR**	0.905	0.905	0.905	0.937	0.937	0.945	-	-
	NoF	1	1	1	2	2	5	-	-
3	**ACCR**	0.945	0.945	0.945	0.945	0.945	0.945	0.945	0.945
	NoF	5	5	5	5	5	5	5	5
4	**ACCR**	0.945	0.945	0.945	0.950	0.950	0.955	0.955	0.955
	NoF	5	5	5	8	8	12	12	12
5	**ACCR**	0.870	0.870	0.872	0.872	0.872	0.873	0.873	-
	NoF	1	1	2	2	2	3	3	-
6	**ACCR**	0.945	0.945	0.945	0.950	0.950	0.955	0.955	0.955
	NoF	5	5	5	8	8	12	12	12
7	**ACCR**	0.870	0.870	0.872	0.872	0.872	0.873	0.945	0.945
	NoF	1	1	2	2	2	3	5	5
8	**ACCR**	0.945	0.945	0.945	0.950	0.950	0.955	0.955	0.955
	NoF	5	5	5	8	8	12	12	12
9	**ACCR**	0.870	0.870	0.872	0.872	0.873	0.873	0.873	0.947
	NoF	1	1	2	2	3	3	3	6
10	**ACCR**	0.935	0.945	0.947	0.950	0.950	0.955	-	-
	NoF	4	5	6	8	8	12	-	-

5.2.4. The Performance of IDS on Detecting Decreased Rank Attack

The best performances obtained with respect to each objective, namely accuracy (ACCR) and number of features (NoF) for detecting decreased rank attacks are presented for each scenario (S) in Table 5.9. The trade-offs (TO) between ACCR and NoF for each GP run for an exemplar scenario (S1) are given in Table 5.10. As shown in the results, while the highest

accuracy rate (88%) for scenario 1 (S1) is achieved by using thirteen features, it drops to 81% when only one feature is used (1 (0.810)).

When comparing the S1 with the S7 and S8, it is seen that the S7 and S8 have higher performance. It would be a more consistent assessment to mention that such a result occurs because the attacker locations are close or far from the root node, changing the transmission load. As stated in [39], the cooperation of the attackers can significantly reduce network performance when the attackers are positioned "high-forwarding load area". Moreover, in this type of attack, it is seen that the entire network is affected rather than the local area effect. Here, the S7 and S8 results show that more features are used for better performance than other scenarios in Table 5.9. In the S7 and S8 scenario designs, the data collected from the far nodes group for the training environment and close nodes group for the testing environment. Considering the NoF values in all scenario results, it can be said that it is necessary to use approximately 10 (the average value for NoF is 10,5) features for a good ID program performance. In this study, it is also seen that it is very difficult to detect the DR attack type compared to other attack types.

Table 5.9 The best performances with respect to accuracy (ACCR) and number of features (NoF) obtained in the Pareto front set for DRA.

Objective	**S1**	**S2**	**S3**	**S4**	**S5**	**S6**	**S7**	**S8**
ACCR	0.880	0.875	0.870	0.865	0.880	0.875	0.900	0.890
NoF	(11)	(10)	(10)	(8)	(11)	(10)	(12)	(12)
NoF	1	1	1	1	1	1	1	1
ACCR	(0.810)	(0.800)	(0.820)	(0.820)	(0.835)	(0.810)	(0.800)	(0.805)

Table 5.10 The trade-offs between objectives obtained in the Pareto set of each GP run for Decreased Rank Attack.

GP Run	Objective	TO1	TO2	TO3	TO4	TO5	TO6	TO7	TO8
1	**ACCR**	0.810	0.810	0.810	0.872	0.872	0.872	0.873	0.873
	NoF	1	1	1	2	2	2	3	3
2	**ACCR**	0.810	0.872	0.872	0.873	0.873	0.875	0.877	-
	NoF	1	2	2	3	3	5	6	-
3	**ACCR**	0.810	0.872	0.872	0.873	0.873	0.875	0.877	0.880
	NoF	1	2	2	4	4	5	6	11
4	**ACCR**	0.810	0.872	0.873	0.875	0.877	0.878	0.880	-
	NoF	1	2	3	4	5	6	11	-
5	**ACCR**	0.878	0.878	0.878	0.880	0.880	0.880	-	-
	NoF	6	6	6	11	11	11	-	-
6	**ACCR**	0.875	0.875	0.878	0.880	0.880	0.880	-	-
	NoF	5	5	6	11	11	11	-	-
7	**ACCR**	0.810	0.810	0.810	0.872	0.872	0.872	0.873	0.873
	NoF	1	1	1	2	2	2	3	3
8	**ACCR**	0.810	0.872	0.872	0.873	0.873	0.875	0.877	0.880
	NoF	1	2	2	4	4	5	6	11
9	**ACCR**	0.810	0.872	0.873	0.875	0.877	0.878	0.880	-
	NoF	1	2	3	4	5	6	11	-
10	**ACCR**	0.810	0.810	0.872	0.872	0.872	0.873	0.873	0.878
	NoF	1	1	2	2	2	3	3	6

5.2.5. The Performance of IDS on Detecting Worst Parent Attack with Mobile Attackers

The best performances obtained with respect to each objective, namely accuracy (ACCR) and number of features (NoF) for detecting worst parent attacks when the attackers are mobile are presented for each scenario (S) in Table 5.11. The trade-offs (TO) between ACCR and NoF for each GP run for an exemplar scenario (S1) is given Table 5.12. As shown in the results, while the highest accuracy rate (94%) for scenario 1 (S1) is achieved by using thirteen features, it drops to 84% when only one feature is used (1 (0.840)). A comparison with results when attacker nodes are not mobile will be provided shortly. Here, the accuracy results show a slight decrease, and the NoF values show a slight increase compared to the scenarios' results where the attackers are not mobile. However, the changes here are not significant changes. The average NoF values have a slight increase is seen from 8.875 to 10.125 and the

average accuracy values have a slight decrease from 93.75% to 92.70%. However, for each scenario, we can still obtain high accuracy for this attack type.

Table 5.11 The best performances with respect to accuracy (ACCR) and number of features (NoF) obtained in the Pareto front set for WPA with mobile attackers.

Objective	**S1**	**S2**	**S3**	**S4**	**S5**	**S6**	**S7**	**S8**
ACCR	0.940 (9)	0.940 (8)	0.950 (12)	0.945 (9)	0.930 (8)	0.935 (13)	0.930 (10)	0.925 (12)
NoF	1 (0.840)	1 (0.825)	1 (0.825)	1 (0.825)	1 (0.825)	1 (0.825)	1 (0.835)	1 (0.810)

Table 5.12 The results (R) of ten experiments in Scenario-1 when the network is under Worst Parent Attack and the attackers are mobile.

GP Run	**Objective**	**TO1**	**TO2**	**TO3**	**TO4**	**TO5**	**TO6**	**TO7**	**TO8**
1	**ACCR**	0.840	0.840	0.840	0.840	0.855	0.855	0.855	-
	NoF	1	1	1	1	2	2	2	-
2	**ACCR**	0.860	0.860	0.860	0.860	0.870	0.870	-	-
	NoF	3	3	3	3	4	4	-	-
3	**ACCR**	0.870	0.870	0.870	0.875	0.875	0.875	0.880	-
	NoF	1	1	1	2	2	2	3	-
4	**ACCR**	0.860	0.860	0.860	0.860	0.860	0.860	-	-
	NoF	5	5	5	5	5	5	-	-
5	**ACCR**	0.935	0.935	0.940	0.940	0.940	0.940	0.940	0.940
	NoF	8	8	9	9	9	9	9	9
6	**ACCR**	0.870	0.870	0.870	0.870	0.870	-	-	-
	NoF	4	4	4	4	4	-	-	-
7	**ACCR**	0.905	0.905	0.905	0.930	0.937	0.937	0.937	0.937
	NoF	2	2	2	4	5	5	5	5
8	**ACCR**	0.865	0.865	0.865	0.865	0.865	0.870	0.875	0.875
	NoF	3	3	3	3	3	6	6	7
9	**ACCR**	0.910	0.910	0.910	0.910	0.910	0.915	0.925	0.925
	NoF	3	3	3	3	3	6	7	7
10	**ACCR**	0.925	0.925	0.930	0.930	0.935	0.935	0.935	0.940
	NoF	7	7	8	8	8	8	8	9

5.2.6. The Performance of IDS on Detecting Hello Flood Attack with Mobile Attackers

The best performances obtained with respect to each objective, namely accuracy (ACCR) and number of features (NoF) are presented for each scenario (S) in Table 5.13. The trade-offs (TO) between ACCR and NoF for each GP run for an exemplar scenario (S1) is given Table 5.14. As shown in the results, while the highest accuracy rate (94.5%) for scenario 1 (S1) is achieved by using thirteen features, it drops to 85.5% when only one feature is used (1 (0.855)). A comparison with results when attacker nodes are not mobile will be provided shortly. Here, the accuracy results show a slight decrease, and the NoF values show a slight increase compared to the scenarios' results where the attackers are not mobile. However, the changes here are not significant changes. The average NoF values have a slight increase is seen from 10.375 to 11.625 and the average accuracy values have a slight decrease from 93.75% to 93.3%.

Table 5.13 The extreme results of experiments when the network is under Hello Flood Attack with Mobile Attackers.

Objective	**S1**	**S2**	**S3**	**S4**	**S5**	**S6**	**S7**	**S8**
ACCR	0.945 (14)	0.940 (13)	0.935 (12)	0.940 (12)	0.925 (9)	0.930 (12)	0.920 (12)	0.915 (9)
NoF	1 (0.855)	1 (0.845)	1 (0.855)	1 (0.845)	1 (0.835)	1 (0.835)	1 (0.830)	1 (0.815)

Table 5.14 The best performances with respect to accuracy (ACCR) and number of features (NoF) obtained in the Pareto front set for HFA with Mobile Attackers.

GP Run	Objectives	TO1	TO2	TO3	TO4	TO5	TO6	TO7	TO8
1	**ACCR**	0.840	0.840	0.840	0.840	0.855	0.855	0.855	-
	NoF	1	1	1	1	2	2	2	-
2	**ACCR**	0.870	0.870	0.870	0.870	0.870	0.870	-	-
	NoF	3	3	3	3	3	3	-	-
3	**ACCR**	0.870	0.870	0.870	0.872	0.872	0.872	0.873	-
	NoF	1	1	1	2	2	2	3	-
4	**ACCR**	0.870	0.870	0.870	0.870	0.870	0.870	-	-
	NoF	3	3	3	3	3	3	-	-
5	**ACCR**	0.940	0.940	0.940	0.940	0.945	0.945	0.945	0.945
	NoF	12	12	12	12	14	14	14	14
6	**ACCR**	0.870	0.870	0.870	0.870	0.870	-	-	-
	NoF	6	6	6	6	6	-	-	-
7	**ACCR**	0.915	0.915	0.915	0.930	0.935	0.935	0.935	0.935
	NoF	2	2	2	3	4	4	4	4
8	**ACCR**	0.870	0.870	0.870	0.870	0.870	0.935	0.935	0.940
	NoF	3	3	3	3	3	6	6	7
9	**ACCR**	0.890	0.890	0.890	0.890	0.890	0.925	0.925	0.930
	NoF	4	4	4	4	4	6	6	7
10	**ACCR**	0.935	0.935	0.940	0.940	0.945	0.945	0.945	-
	NoF	12	12	13	13	14	14	14	-

5.2.7. The Performance of IDS on Detecting Increased Version Attack with Mobile Attackers

The best performances obtained with respect to each objective, namely accuracy (ACCR) and number of features (NoF) are presented for each scenario (S) in Table 5.15. The trade-offs (TO) between ACCR and NoF for each GP run for an exemplar scenario (S1) is given Table 5.16. As shown in the results, while the highest accuracy rate (94.5%) for scenario 1 (S1) is achieved by using thirteen features, it drops to 86% when only one feature is used (1 (0.860)). A comparison with results when attacker nodes are not mobile will be provided shortly. Here, the accuracy results show a slight decrease, and the NoF values show an increase compared to the scenarios' results where the attackers are not mobile. The average NoF values increase from 9.875 to 11.625, which means that IDS needs more features to detect the attack when

attackers are mobile. The average accuracy values have a slight decrease from 93.7% to 92.7%.

Table 5.15 The extreme results of experiments when the network is under Increased Version Attack.

Objective	S1	S2	S3	S4	S5	S6	S7	S8
ACCR	0.945 (13)	0.945 (14)	0.920 (10)	0.910 (8)	0.935 (12)	0.930 (14)	0.925 (10)	0.905 (12)
NoF	1 (0.860)	1 (0.860)	1 (0.855)	1 (0.845)	1 (0.825)	1 (0.820)	1 (0.835)	1 (0.820)

Table 5.16 The best performances with respect to accuracy (ACCR) and number of features (NoF) obtained in the Pareto front set for IVA with Mobile Attackers.

GP Run	Objective	TO1	TO2	TO3	TO4	TO5	TO6	TO7	TO8
1	**ACCR**	0.860	0.860	0.860	0.860	0.865	0.865	0.865	-
	NoF	1	1	1	1	2	2	2	-
2	**ACCR**	0.870	0.870	0.870	0.870	0.870	0.870	-	-
	NoF	4	4	4	4	4	4	-	-
3	**ACCR**	0.870	0.870	0.870	0.872	0.872	0.875	0.875	-
	NoF	4	4	4	5	5	7	7	-
4	**ACCR**	0.880	0.880	0.880	0.880	0.880	0.880	-	-
	NoF	4	4	4	4	4	4	-	-
5	**ACCR**	0.940	0.940	0.945	0.945	0.945	0.945	0.945	0.945
	NoF	11	11	13	13	13	13	13	13
6	**ACCR**	0.870	0.870	0.870	0.870	0.870	0.870	0.870	-
	NoF	4	4	4	4	4	4	4	-
7	**ACCR**	0.900	0.900	0.900	0.915	0.915	0.940	0.940	0.940
	NoF	1	1	1	3	3	4	4	4
8	**ACCR**	0.870	0.870	0.870	0.870	0.870	0.935	0.935	0.940
	NoF	3	3	3	3	3	6	6	7
9	**ACCR**	0.870	0.875	0.875	0.880	0.880	0.910	0.910	0.910
	NoF	3	4	4	5	5	6	6	6
10	**ACCR**	0.930	0.930	0.930	0.935	0.935	0.940	0.945	-
	NoF	6	6	6	11	11	11	13	-

5.2.8. The Performance of IDS on Detecting Decreased Rank Attack with Mobile Attackers

The best performances obtained with respect to each objective, namely accuracy (ACCR) and number of features (NoF) are presented for each scenario (S) in Table 5.17. The trade-offs (TO) between ACCR and NoF for each GP run for an exemplar scenario (S1) is given Table 5.18. As shown in the results, while the highest accuracy rate (87.5%) for scenario 1 (S1) is achieved by using fourteen features, it drops to 80% when only one feature is used (1 (0.800)). A comparison with results when attacker nodes are not mobile will be provided shortly. Here, the accuracy results show a noticeable decrease, and the NoF values show a slight increase compared to the scenario results where the attackers are not mobile. Average NoF values have increased from 10.5 to 12.125, and average accuracy values have a slight decrease from 87.9% to 87.7%. As stated previously, IDS cannot detect this type of attack as effectively as other types. On the other hand, the mobility of attackers have a lower impact on accuracy as well.

Table 5.17 The extreme results of experiments when the network is under Decreased Rank Attack with Mobile Attackers.

Objetive	**S1**	**S2**	**S3**	**S4**	**S5**	**S6**	**S7**	**S8**
ACCR	0.875 (14)	0.865 (12)	0.865 (11)	0.860 (9)	0.875 (13)	0.865 (12)	0.910 (12)	0.900 (14)
NoF	1 (0.800)	1 (0.790)	1 (0.810)	1 (0.810)	1 (0.830)	1 (0.805)	1 (0.800)	1 (0.815)

Table 5.18 The best performances with respect to accuracy (ACCR) and number of features (NoF) obtained in the Pareto front set for DRA with Mobile Attackers.

GP Run	Objective	TO1	TO2	TO3	TO4	TO5	TO6	TO7	TO8
1	**ACCR**	0.850	0.850	0.850	0.850	0.855	0.875	0.875	-
	NoF	4	4	4	4	5	14	14	-
2	**ACCR**	0.870	0.870	0.870	0.870	0.870	0.870	-	-
	NoF	12	12	12	12	12	12	-	-
3	**ACCR**	0.800	0.800	0.800	0.800	0.810	0.810	0.820	-
	NoF	1	1	1	1	2	2	3	-
4	**ACCR**	0.870	0.870	0.870	0.870	0.870	0.870	-	-
	NoF	10	10	10	10	10	10	-	-
5	**ACCR**	0.800	0.800	0.850	0.855	0.855	0.870	0.870	0.870
	NoF	1	1	4	5	5	12	12	12
6	**ACCR**	0.870	0.870	0.870	0.870	0.870	-	-	-
	NoF	13	13	13	13	13	-	-	-
7	**ACCR**	0.820	0.820	0.820	0.820	0.820	0.820	-	-
	NoF	3	3	3	3	3	3	-	-
8	**ACCR**	0.870	0.870	0.870	0.870	0.870	0.875	0.875	0.875
	NoF	12	12	12	12	12	14	14	14
9	**ACCR**	0.850	0.850	0.850	0.850	0.850	-	-	-
	NoF	4	4	4	4	4	-	-	-
10	**ACCR**	0.800	0.800	0.850	0.855	0.855	0.870	0.870	0.875
	NoF	1	1	4	5	5	12	12	14

5.3. General Discussions

In this study, for evaluating our proposed IDS's performance, we have considered the set of Pareto front that represents the objectives of the Pareto dominant individuals found for each simulated scenario. Because we have run the GP algorithm 10 times for each scenario, we have aggregated 10 Pareto front sets and extracted the extreme points from these sets to reveal how well the GP algorithm could achieve maximum optimal performance according to the accuracy (ACCR) and the number of features (NoF). These extreme points are shown in Table 5.19 separately for each of the two objectives. The results for networks when attackers are mobile are given in Table 5.20. For example, '0.945 (8)' stated for the ACCR implies that the GP could reach 0.945 accuracy with a model that has eight distinct features. Similarly, '1 (0.850)' stated for NoF implies that the GP model has only one distinct feature and could reach 0.850 accuracy.

Table 5.19 The best performances with respect to accuracy (ACCR) and number of features (NoF) obtained in the Pareto front set.

Scenario	Objective	WP	HF	IV	DR
S1	**ACCR**	0.945 (8)	0.950 (13)	0.955 (12)	0.880 (11)
	NoF	1 (0.850)	1 (0.865)	1 (0.870)	1 (0.810)
S2	**ACCR**	0.940 (7)	0.945 (12)	0.950 (12)	0.870 (10)
	NoF	1 (0.840)	1 (0.855)	1 (0.870)	1 (0.800)
S3	**ACCR**	0.950 (11)	0.945 (10)	0.935 (8)	0.870 (10)
	NoF	1 (0.845)	1 (0.860)	1 (0.860)	1 (0.820)
S4	**ACCR**	0.940 (8)	0.940 (9)	0.925 (6)	0.865 (8)
	NoF	1 (0.840)	1 (0.855)	1 (0.850)	1 (0.820)
S5	**ACCR**	0.935 (8)	0.930 (9)	0.940 (10)	0.880 (11)
	NoF	1 (0.835)	1 (0.845)	1 (0.830)	1 (0.835)
S6	**ACCR**	0.935 (12)	0.925 (11)	0.945 (12)	0.875 (10)
	NoF	1 (0.830)	1 (0.840)	1 (0.825)	1 (0.810)
S7	**ACCR**	0.930 (9)	0.925 (12)	0.935 (10)	0.900 (12)
	NoF	1 (0.835)	1 (0.830)	1 (0.835)	1 (0.800)
S8	**ACCR**	0.925 (8)	0.920 (7)	0.910 (9)	0.890 (12)
	NoF	1 (0.825)	1 (0.830)	1 (0.810)	1 (0.805)

Table 5.20 The best performances with respect to accuracy (ACCR) and number of features (NoF) obtained in the Pareto front set when the attackers are mobile.

Scenario	Objective	WP	HF	IV	DR
S1	**ACCR**	0.940 (9)	0.945 (14)	0.945 (13)	0.875 (14)
	NoF	1 (0.840)	1 (0.855)	1 (0.860)	1 (0.800)
S2	**ACCR**	0.940 (8)	0.940 (13)	0.945 (14)	0.865 (12)
	NoF	1 (0.825)	1 (0.845)	1 (0.860)	1 (0.790)
S3	**ACCR**	0.950 (12)	0.935 (12)	0.920 (10)	0.865 (11)
	NoF	1 (0.825)	1 (0.855)	1 (0.855)	1 (0.810)
S4	**ACCR**	0.945 (9)	0.940 (12)	0.910 (8)	0.860 (9)
	NoF	1 (0.825)	1 (0.845)	1 (0.845)	1 (0.810)
S5	**ACCR**	0.930 (8)	0.925 (9)	0.935 (12)	0.875 (13)
	NoF	1 (0.825)	1 (0.835)	1 (0.825)	1 (0.830)
S6	**ACCR**	0.935 (13)	0.930 (12)	0.930 (14)	0.865 (12)
	NoF	1 (0.825)	1 (0.835)	1 (0.820)	1 (0.805)
S7	**ACCR**	0.930 (10)	0.920 (12)	0.925 (10)	0.910 (12)
	NoF	1 (0.835)	1 (0.830)	1 (0.835)	1 (0.800)
S8	**ACCR**	0.925 (12)	0.915 (9)	0.905 (12)	0.900 (14)
	NoF	1 (0.810)	1 (0.815)	1 (0.820)	1 (0.815)

The results allow us to evaluate our proposed IDS's performance from different points of view. The first is that DR is a harder-to-detect routing attack as compared to other attacks

when considering the average of the accuracies from eight scenarios (it is 0.88 overall when the attackers are not mobile, and 0.84 overall when the attackers are mobile). The difference in the average accuracy performances obtained from the other attacks is not significant, and GP succeeds to evolve a satisfactory ID program for these attacks (it is 0.94 overall when the attackers are not mobile, and it is 0.93 when the attackers are mobile). As for the evaluation according to the average of NoF, it is seen that the ID program requires the more features for the DR attack (when the attackers are not mobile it is 10.5 in overall, and when the attackers are mobile it is 10.625 in overall). This clearly suggests that GP is unable to evolve an ID program that gives higher detection accuracy even after a rigorous evaluation of massive feature data. Furthermore, when attackers are not mobile, the difference in accuracy performances obtained from extreme points with respect to ACCR and NoF ranges from 4.5% to 12.0%. They are the WP and DR attacks that give the highest and least difference in performance, respectively. Similarly, when the attackers are mobile these values are ranges from 4.5% to 12.5%. Here, the result shows that limiting the number of features in the ID program has an adverse impact on the detection capability to some degree, as expected, and this varies according to the targeted attack. Note that these give the largest difference in the performances of GP, and even a few increases in the number of features in ID program yield much better accuracy. Moreover, mobility increases the number of features required for effective detection for all attack types. On the other hand, whether the attacker nodes are mobile or not does not make a significant difference in the accuracy results. Because, even if the attacker nodes are mobile, they remain in a certain region in terms of their area of influence. In the future, a study can be carried out by expanding the range of motion of attacker nodes.

When it comes to the change in IDS performance when only attackers are repositioned with the same configurations of the monitoring nodes (that is, the comparison of S1 with S2, S3 with S4, and so forth) a slight performance degradation is often observed, and the change here is no more than 2.5% (when the attackers are not mobile) and 2% (when the attackers are mobile). This is not surprising because the locations of attackers are positioned randomly from the entire network, and there are cases studied in these scenarios where the ID program

evolved and tested when the attacker nodes are, respectively, in the vicinity and away from the monitoring or root node. Regarding the change in IDS performance according to different configurations of monitoring nodes by keeping attacker's positions the same and different (that is, comparison of S1 with S3 and S2 with S4), it is seen that the performance of the evolved ID program slightly improves (up to 2.5%, when the attackers are not mobile, and 3% when the attackers are mobile) when monitoring nodes are positioned within the first three hops.

In order to reveal how monitoring nodes are helpful in improving the attack detection capability of the ID program, we replicated the simulations by adopting a standalone architecture where the root node is in charge of intrusion detection alone. The simulation here is run with two attacker configurations that are denoted 'same' and 'different' which again represent the cases where attackers are positioned at the same and different locations when ID program is evaluated in the testing environment. Respectively, the results for static and mobile attackers are shown in Table 5.21-Table 5.22. Note that S1 through S4 in Table 5.19 and Table 5.20 should only be considered to ensure a fair comparison between the performance of ***collaborative*** and ***standalone*** architectures.

Table 5.21 The extreme points with respect to ACCR and NoF in the Pareto front sets obtained by the standalone architecture.

Position	Objective	WP	HF	IV	DR
Same	**ACCR**	0.910 (9)	0.920 (12)	0.935 (11)	0.865 (10)
	NoF	1 (0.835)	1 (0.840)	1 (0.830)	1 (0.790)
Different	**ACCR**	0.905 (8)	0.915 (11)	0.925 (12)	0.860 (9)
	NoF	1 (0.825)	1 (0.830)	1 (0.825)	1 (0.780)

Table 5.22 The extreme points with respect to ACCR and NoF in the Pareto front sets obtained by the standalone architecture when the attackers are mobile.

Position	Objective	WP	HF	IV	DR
Same	**ACCR**	0.905 (10)	0.915 (12)	0.920 (13)	0.855 (15)
	NoF	1 (0.820)	1 (0.815)	1 (0.805)	1 (0.770)
Different	**ACCR**	0.900 (10)	0.910 (12)	0.910 (13)	0.850 (10)
	NoF	1 (0.815)	1 (0.820)	1 (0.815)	1 (0.765)

The results suggest that the performance of the ID program increases with the collaborative architecture, and when attackers are not mobile, the difference reaches 4%. On the other hand, when attackers are mobile, this difference increases to 4.5%. However, no significant differences are observed in terms of NoF.

In order to evaluate how early the proposed IDS system can detect intrusions, we inspect the behavior of the IDS on individual flows that are split for 60 s. Note that, as stressed earlier, the network traffic is split into the individual flows that capture the feature from the sub-traffic within a predefined time interval, and the extracted features are evaluated by IDS just after the flow. Experimental results show that the proposed IDS can detect 50%, 46.875%, and 3.125% of attacks at the end of the first, second, and third window interval, respectively.

Finally, the importance of the features is also assessed to reveal which features greatly influence the detection ability of the evolved IDS. For this, the Weka [78] toolkit is used. It has been observed that PacketCountDIS (97.5%), MaxIntervalMin (96.42%), PacketCountData (94.5%), MinTimeBtwDAO (92.5%), PacketCountDIO (91.25%), PacketCountDIO (91.25%) are important for **WP attack** whereas PacketCountDIS (97.25%), PacketCountDIO (97.25%), MinTimeBtwDAO (96%), AvgIntervalDoubling (93.5%), AvgTimeBtwData (91%), MaxIntervalMin (88%) are important for **HF attack**. As for **IV attack**, PacketCountDIO (96%), PacketCountData (95.5%), AvgTimeBtwData (94.5%), MinTimeBtwDIO (93%), MaxIntervalMin (90%) are found important features. Finally, PacketCountDIS (97.5%), PacketCountDIO (96.25%), PacketCountData (95.25%), MaxIntervalMin (93%), AvgTimeBtwData (92%), MinTimeBtwDIO (91%) are important for **DR attack**.

6. CONCLUSION

The rapid development of IoT, which gradually increases its impact and frequency of use in all areas of life, unfortunately has brought security concerns with it. IoT devices share and transfer data among themselves due to the nature of the network they are in. In many cases, this data may be proprietary or confidential information. A specific purpose of an IoT device may require it to connect to the Internet or stay connected to the network. However, these devices also have some resource (memory, energy etc.) limitations. These limitations are the weakest link for IoT devices. Currently, these systems exist in a wide range of applications and usage areas due to the requirements of our age. From smart hospital projects to space technologies; it is in a long range from wearable technologies to smart phones. This wide usage area and different system requirements cause traditional security mechanisms and candidate solutions for IoT security to not meet the security needs most of the time, and often create a so-called security system. In addition, in resource-constrained networks, it is very important to calculate the communication cost and keep it at a minimum. However, current security solutions are not suitable for such complex networks problems.

RPL, one of the routing protocol, is designed for resource-constrained IoT networks. The lack of security mechanisms in the RPL structure is highly susceptible to internal and external attacks. The network structure of RPL-based networks has made this network too complex for various intrusion detection to be performed by traditional methods. There are many solutions and approaches for these complex optimization problems. Pareto-based approaches are one of the most popular approaches in this area, for producing satisfactory solutions and solving multi-objective problems. Since more than one candidate offers a solution, Pareto-based approach gives the researcher a chance to evaluate by adding real-life concrete or abstract constraints.

In this thesis, we explore the use of the Pareto-based multi-objective approach to efficiently and effectively detect four different attack types (WP attack, HF attack, IV attack, and DR attack) specific to RPL.

To the best of the author's knowledge, this thesis is the first study which goals to simultaneously optimize detection accuracy (ACCR) of ID programs and their costs including communication cost of ID nodes. To do that, a massive number of simulations are generated, and the different ID programs are evolved from these simulations. The evaluations are made on the basis of Pareto sets obtained from the evolved programs. The proposed approach has been shown to ensure satisfactory performance in the detection accuracy of the attacks covered in this study.

REFERENCES

[1] Selim Yılmaz, Emre Aydogan, and Sevil Sen. A transfer learning approach for securing resource-constrained iot devices. *IEEE Transactions on Information Forensics and Security*, 16:4405–4418, **2021**.

[2] Ahmed Raoof, Ashraf Matrawy, and Chung-Horng Lung. Routing attacks and mitigation methods for rpl-based internet of things. *IEEE Communications Surveys & Tutorials*, 21(2):1582–1606, **2018**.

[3] Anthea Mayzaud, Remi Badonnel, and Isabelle Chrisment. A taxonomy of attacks in rpl-based internet of things. *International Journal of Network Security*, 18(3):459–473, **2016**.

[4] Erdem Canbalaban and Sevil Sen. A cross-layer intrusion detection system for rpl-based internet of things. In *International Conference on Ad-Hoc Networks and Wireless*, pages 214–227. Springer, **2020**.

[5] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, **2002**.

[6] Ahmet Arış and Sema F Oktuğ. Analysis of the rpl version number attack with multiple attackers. In *2020 International Conference on Cyber Situational Awareness, Data Analytics and Assessment (CyberSA)*, pages 1–8. IEEE, **2020**.

[7] Virendra Pal Singh, Sweta Jain, and Jyoti Singhai. Hello flood attack and its countermeasures in wireless sensor networks. *International Journal of Computer Science Issues (IJCSI)*, 7(3):23, **2010**.

[8] Internet of things (iot) connected devices installed base worldwide from 2015 to 2025 (in billions). `https://www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide/`. Accessed: 2020-04-01.

[9] Cisco. Cisco visual networking index: Forecast and trends, 2017–2022 white paper. https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/white-paper-c11-741490.html. Accessed: 2020-04-01.

[10] Alex Koohang, Carol Springer Sargent, Jeretta Horn Nord, and Joanna Paliszkiewicz. Internet of things (iot): From awareness to continued use. *International Journal of Information Management*, 62:102442, **2022**.

[11] Jonathan Hui and Pascal Thubert. Compression format for ipv6 datagrams over ieee 802.15. 4-based networks. Technical report, **2011**.

[12] IEEE Standards Association et al. Ieee std 802.15. 4-2011, ieee standard for local and metropolitan area networks—part 15.4: Low-rate wireless personal area networks (lr-wpans), **2011**.

[13] Roger Alexander, Anders Brandt, JP Vasseur, Jonathan Hui, Kris Pister, Pascal Thubert, P Levis, Rene Struik, Richard Kelsey, and Tim Winter. RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks. RFC 6550, **2012**. doi:10.17487/RFC6550.

[14] Akshaya Dhingra and Vikas Sindhu. A study of rpl attacks and defense mechanisms in the internet of things network. In *2022 International Conference on Computing, Communication, Security and Intelligent Systems (IC3SIS)*, pages 1–6. IEEE, **2022**.

[15] YP Raiwani. Internet of things: a new paradigm. *International Journal of Scientific and Research Publications*, 3(4):1–4, **2013**.

[16] Jayavardhana Gubbi, Rajkumar Buyya, Slaven Marusic, and Marimuthu Palaniswami. Internet of things (iot): A vision, architectural elements, and future directions. *Future generation computer systems*, 29(7):1645–1660, **2013**.

[17] Pradyumna Gokhale, Omkar Bhat, and Sagar Bhat. Introduction to iot. *International Advanced Research Journal in Science, Engineering and Technology*, 5(1):41–44, **2018**.

[18] Number of internet of things (iot) connected devices worldwide from 2019 to 2030(in billions). `https://www.statista.com/statistics/1183457/iot-connected-devices-worldwide/`. Accessed:2022-06-06.

[19] Cleber M de Morais, Djamel Sadok, and Judith Kelner. An iot sensor and scenario survey for data researchers. *Journal of the Brazilian Computer Society*, 25(1):1–17, **2019**.

[20] Hao Chen, Xueqin Jia, and Heng Li. A brief introduction to iot gateway. In *IET international conference on communication technology and application (ICCTA 2011)*, pages 610–613. IET, **2011**.

[21] Ulrich Herberg and Thomas Clausen. A comparative performance study of the routing protocols load and rpl with bi-directional traffic in low-power and lossy networks (lln). In *Proceedings of the 8th ACM Symposium on Performance Evaluation of Wireless Ad Hoc, Sensor, and Ubiquitous Networks*, PE-WASUN '11, page 73–80. Association for Computing Machinery, New York, NY, USA, **2011**. ISBN 9781450309004. doi:10.1145/2069063.2069076.

[22] Ulrich Herberg and Thomas Clausen. A comparative performance study of the routing protocols load and rpl with bi-directional traffic in low-power and lossy networks (lln). In *Proceedings of the 8th ACM Symposium on Performance evaluation of wireless ad hoc, sensor, and ubiquitous networks*, pages 73–80. **2011**.

[23] Cansu Dogan, Selim Yilmaz, and Sevil Sen. Analysis of rpl objective functions with security perspective. In *SENSORNETS*, pages 71–80. **2022**.

[24] Ibtissem Zaatouri, Nouha Alyaoui, Awatef Benfradj Guiloufi, Francoise Sailhan, and Abdennaceur Kachouri. Design and performance analysis of objective functions for rpl routing protocol. *Wireless Personal Communications*, 124(3):2677–2697, **2022**.

[25] Tim Winter, Pascal Thubert, Anders Brandt, Jonathan Hui, Richard Kelsey, Philip Levis, Kris Pister, Rene Struik, Jean-Philippe Vasseur, and Roger Alexander. Rpl: Ipv6 routing protocol for low-power and lossy networks. Technical report, **2012**.

[26] Anthéa Mayzaud, Rémi Badonnel, and Isabelle Chrisment. A taxonomy of attacks in rpl-based internet of things. *Int. J. Netw. Secur.*, 18:459–473, **2016**.

[27] Sevil Şen, John A Clark, and Juan E Tapiador. Power-aware intrusion detection in mobile ad hoc networks. In *Ad Hoc Networks: First International Conference, ADHOCNETS 2009, Niagara Falls, Ontario, Canada, September 22-25, 2009. Revised Selected Papers 1*, pages 224–239. Springer, **2010**.

[28] Behnam Farzaneh, Mohammad Ali Montazeri, and Shahram Jamali. An anomaly-based ids for detecting attacks in rpl-based internet of things. In *2019 5th International Conference on Web Research (ICWR)*, pages 61–66. IEEE, **2019**.

[29] Jaspreet Kaur and Gagandeep Singh. A blockchain-based machine learning intrusion detection system for internet of things. In *Principles and Practice of Blockchains*, pages 119–134. Springer, **2023**.

[30] Pankaj Pal, Sachin Tripathi, and Chiranjeev Kumar. Bandwidth estimation in high mobility scenarios of manet using nsga-ii optimized fuzzy inference system. *Applied Soft Computing*, 123:108936, **2022**.

[31] Safia Lateef, Muhammad Rizwan, and Muhammad Abul Hassan. Security threats in flying ad hoc network (fanet). *Computational Intelligence for Unmanned Aerial Vehicles Communication Networks*, pages 73–96, **2022**.

[32] Mohsen Sheibani, Behrang Barekatain, and Erfan Arvan. A lightweight distributed detection algorithm for ddao attack on rpl routing protocol in internet of things. *Pervasive and Mobile Computing*, 80:101525, **2022**.

[33] Aryan Mohammadi Pasikhani, John A Clark, and Prosanta Gope. Adaptive hybrid heterogeneous ids for 6lowpan. *arXiv preprint arXiv:2205.09170*, **2022**.

[34] Xin-She Yang and Luniver Press. Nature-inspired metaheuristic algorithms second edition, **2010**.

[35] Selim Yılmaz. Electric fish optimization: A new heuristic algorithm based on electrolocation. **2020**.

[36] Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and TAMT Meyarivan. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE transactions on evolutionary computation*, 6(2):182–197, **2002**.

[37] Girish Sharma, Jyoti Grover, and Abhishek Verma. Performance evaluation of mobile rpl-based iot networks under version number attack. *Computer Communications*, **2022**.

[38] Cooja simulator. `https://www:https://anrg.usc.edu/contiki/index.php/Cooja_Simulator`. Accessed: 2022-12-9.

[39] Anhtuan Le, Jonathan Loo, Aboubaker Lasebae, Alexey Vinel, Yue Chen, and Michael Chai. The impact of rank attack on network topology of routing protocol for low-power and lossy networks. *IEEE Sensors Journal*, 13(10):3685–3692, **2013**.

[40] Anhtuan Le, Jonathan Loo, Yuan Luo, and Aboubaker Lasebae. The impacts of internal threats towards routing protocol for low power and lossy network performance. In *2013 IEEE symposium on computers and communications (ISCC)*, pages 000789–000794. IEEE, **2013**.

[41] Shahid Raza, Linus Wallgren, and Thiemo Voigt. Svelte: Real-time intrusion detection in the internet of things. *Ad hoc networks*, 11(8):2661–2674, **2013**.

[42] Deepali Bankatsingh Gothawal and SV Nagaraj. Anomaly-based intrusion detection system in rpl by applying stochastic and evolutionary game models over iot environment. *Wireless Personal Communications*, 110(3):1323–1344, **2020**.

[43] Emre Aydogan, Selim Yilmaz, Sevil Sen, Ismail Butun, Stefan Forsström, and Mikael Gidlund. A central intrusion detection system for rpl-based industrial internet of things. In *2019 15th IEEE International Workshop on Factory Communication Systems (WFCS)*, pages 1–5. IEEE, **2019**.

[44] F Zahra, NZ Jhanjhi, Sarfraz Nawaz Brohi, Navid Ali Khan, Mehedi Masud, and Mohammed A AlZain. Rank and wormhole attack detection model for rpl-based internet of things using machine learning. *Sensors*, 22(18):6765, **2022**.

[45] Wireshark. `https://sys.cs.uos.de/bonnmotion/doc/README.pdf`. Accessed: 2022-12-12.

[46] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems*, 30, **2017**.

[47] Nicolas M Müller, Pascal Debus, Daniel Kowatsch, and Konstantin Böttinger. Distributed anomaly detection of single mote attacks in rpl networks. In *ICETE (2)*, pages 378–385. **2019**.

[48] Anthéa Mayzaud, Rémi Badonnel, and Isabelle Chrisment. A distributed monitoring strategy for detecting version number attacks in rpl-based networks. *IEEE transactions on network and service management*, 14(2):472–486, **2017**.

[49] Anthéa Mayzaud, Anuj Sehgal, Rémi Badonnel, Isabelle Chrisment, and Jürgen 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*, pages 366–374. IEEE, **2016**.

[50] Syeda Mariam Muzammal, Raja Kumar Murugesan, Noor Zaman Jhanjhi, and Low Tang Jung. Smtrust: proposing trust-based secure routing protocol for rpl attacks for iot applications. In *2020 International Conference on Computational Intelligence (ICCI)*, pages 305–310. IEEE, **2020**.

[51] David Airehrour, Jairo A Gutierrez, and Sayan Kumar Ray. Sectrust-rpl: A secure trust-aware rpl routing protocol for internet of things. *Future Generation Computer Systems*, 93:860–876, **2019**.

[52] Nabil Djedjig, Djamel Tandjaoui, Faiza Medjek, and Imed Romdhani. Trust-aware and cooperative routing protocol for iot security. *Journal of Information Security and Applications*, 52:102467, **2020**.

[53] Abhishek Verma and Virender 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)*, pages 1–6. IEEE, **2019**.

[54] John T Hancock and Taghi M Khoshgoftaar. Survey on categorical data for neural networks. *Journal of Big Data*, 7(1):1–41, **2020**.

[55] Furkan Yusuf Yavuz, ÜNAL Devrim, and GÜL Ensar. Deep learning for detection of routing attacks in the internet of things. *International Journal of Computational Intelligence Systems*, 12(1):39, **2018**.

[56] Semih Cakir, Sinan Toklu, and Nesibe Yalcin. Rpl attack detection and prevention in the internet of things networks using a gru based deep learning. *IEEE Access*, 8:183678–183689, **2020**.

[57] Fangyu Li, Aditya Shinde, Yang Shi, Jin Ye, Xiang-Yang Li, and Wenzhan Song. System statistics learning-based iot security: Feasibility and suitability. *IEEE Internet of Things Journal*, 6(4):6396–6403, **2019**.

[58] Danish Attique, Wang Hao, and Wang Ping. Fog-assisted deep-learning-empowered intrusion detection system for rpl-based resource-constrained smart industries. *Sensors*, 22(23):9416, **2022**.

[59] Huixin Zhang and Yiding Zhao. Vehicle load monitoring method based on nbiot. In *2022 5th International Symposium on Autonomous Systems (ISAS)*, pages 1–5. IEEE, **2022**.

[60] Balachandra Muniyal and Manjula C Belavagi. Intrusion detection using rule based approach in rpl networks. **2022**.

[61] A Parmisano, Sebastian Garcia, and MJ Erquiaga. Stratosphere laboratory. a labeled dataset with malicious and benign iot network traffic, **2019**.

[62] Iman Almomani, Bassam Al-Kasasbeh, and Mousa Al-Akhras. Wsn-ds: A dataset for intrusion detection systems in wireless sensor networks. *Journal of Sensors*, 2016, **2016**.

[63] Ly Vu, Quang Uy Nguyen, Diep N Nguyen, Dinh Thai Hoang, and Eryk Dutkiewicz. Deep transfer learning for iot attack detection. *IEEE Access*, 8:107335–107344, **2020**.

[64] Sean Luke. The ecj owner's manual. *San Francisco, California, A user manual for the ECJ Evolutionary Computation Library*, pages 1–206, **2010**.

[65] Genetic algorithm. `https://https://en.wikipedia.org/wiki/Genetic_algorithm`. Accessed: 2022-12-12.

[66] Andrew L Nelson, Gregory J Barlow, and Lefteris Doitsidis. Fitness functions in evolutionary robotics: A survey and analysis. *Robotics and Autonomous Systems*, 57(4):345–370, **2009**.

[67] Contiki-Ng. contiki-ng/contiki-ng. `https://github.com/contiki-ng/contiki-ng/wiki`, **2004**. Accessed 13-July-2021.

[68] Contiki. contiki-os/contiki. https://github.com/contiki-os/contiki. Accessed: 2022-12-12.

[69] Adam Dunkels, Bjorn Gronvall, and Thiemo Voigt. Contiki-a lightweight and flexible operating system for tiny networked sensors. In *29th annual IEEE international conference on local computer networks*, pages 455–462. IEEE, **2004**.

[70] Mainpage:contiki - zolertia. https://zolertia.sourceforge.net/wiki/index.php/Mainpage:Contiki. Accessed: 2022-12-9.

[71] How to implement sky motes in coap on cooja simulation? https://slogix.in/source-code/contiki-cooja-samples-for-IoT/how-to-implement-sky-motes-in-coap-on-cooja-simulation/. Accessed: 2022-12-9.

[72] Avr motes in cooja. https://github.com/contiki-os/contiki/wiki/AVR-Motes-in-Cooja. Accessed: 2022-12-9.

[73] Dhondta and Bahmadh. Rpl-attacks: Rpl attacks framework. [Online] Avaliable : https://doi.org/10.5281/zenodo.55352, accessed:2022-08-19, **2016**.

[74] Mobility of nodes in cooja. https://anrg.usc.edu/contiki/index.php/Mobility_of_Nodes_in_Cooja. Accessed: 2022-12-11.

[75] Contiki projects community code. https://sourceforge.net/p/contikiprojects/code/HEAD/tree/sics.se/mobility/. Accessed: 2022-12-11.

[76] A mobility scenario generation and analysis tool documentation, version: February 18, 2016 . https://sys.cs.uos.de/bonnmotion/doc/README.pdf. Accessed: 2022-12-12.

[77] Accuracy vs auc in machine learning . https://https://www.baeldung.com/cs/ml-accuracy-vs-auc. Accessed: 2022-12-12.

[78] Weka 3: Machine learning software in java. https://www.cs.waikato.ac.nz/ml/weka/. Accessed: 2022-12-12.