A Transfer Learning Approach for Securing Resource-Constrained IoT Devices

Selim Yılmaz*, Emre Aydogan†, and Sevil Sen‡
WISE Lab., Department of Computer Engineering,
Hacettepe University, Ankara, Turkey
*Department of Computer Engineering,
Muğla Sıtkı Koçman University, Muğla, Turkey
†Department of Computer Engineering,
Akdeniz University, Antalya, Turkey
Email: *selimyilmaz@mu.edu.tr, †emreaydogan@akdeniz.edu.tr, ‡ssen@cs.hacettepe.edu.tr

***Abstract*—In recent years, Internet of Things (IoT) security has attracted significant interest by researchers due to new characteristics of IoT such as heterogeneity of devices, resource constraints, and new types of attacks targeting IoT. Intrusion detection, which is an indispensable part of a security system, is also included in these studies. In order to explore the complex characteristics of IoT, machine learning methods, which rely on long training time to generate intrusion detection models, are proposed in the literature. Furthermore, these systems need to learn a new/fresh model from scratch when the environment changes. This study explores the use of transfer learning in order to generate intrusion detection algorithms for such dynamically changing IoT. Transfer learning is an approach that stores knowledge learned from a problem domain/task and applies that knowledge to another problem domain/task. Here, it is employed in the following two settings: transferring knowledge for generating suitable intrusion algorithms for new devices, transferring knowledge for detecting new types of attacks. In this study, Routing Protocol for Low-Power and Lossy Network (RPL), a routing protocol for resource-constrained wireless networks, is used as an exemplar protocol and specific attacks against RPL are targeted. The experimental results show that the transfer learning approach gives better performance than the traditional approach. Moreover, the proposed approach significantly reduces learning time, which is an important factor for putting devices/networks in operation in a timely manner. Even though transfer learning has been considered a potential candidate for improving IoT security, to the best of our knowledge, this is the first application of transfer learning under these two settings in RPL-based IoT networks.**

***Index Terms*—IoT, security, transfer learning, intrusion detection, genetic programming, RPL**

I. Introduction

IoT is one of the most popular research topics in communication due to significantly increasing numbers of heterogeneous devices connecting to each other and to the Internet. According to Statista [1], the total installed base of IoT devices is expected to be around 75 billion globally by 2025, which shows a five-fold increase in ten years. Cisco projects that Machine-to-Machine (M2M) connections will constitute half of the global connected devices and connections by 2023 [2]. While nearly half of these connections result from home applications, the number of connections resulting from connected work and connected city applications has been showing an increasing trend in recent years [2].

The Low Power and Lossy Networks (LLN) are a special type of IoT, which have different application areas from smart homes to industry. In these networks, devices have generally resource constraints such as energy, memory, and processing power. Moreover, such resource-constrained devices are connected over lossy links. These special characteristics of LLNs have resulted in the emergence of new communication protocols. One of the standardized protocols for these resource-constrained networks is Routing Protocol for Low Power and Lossy Networks (RPL) [3]. RPL builds Destination Oriented Directed Acyclic Graphs (DODAG) in order to represent network topology. However this topology could be susceptible to attacks. Although RPL has some security mechanisms for external attackers, it is still open to insider attacks such as version number and rank attacks. Therefore, developing suitable intrusion detection systems for such resource-constrained networks is vital, which is the main aim of this study.

This study investigates the use of transfer learning (TL) for automatically generating suitable intrusion detection algorithms for a variety of devices in RPL-based IoT networks. Transfer learning simply helps move the knowledge learned in a task/domain to a new task/domain. It helps reduce the learning time needed in the new task/domain. Moreover, it is expected to produce higher initial and final performances for the learned model in the new task/domain compared to learning without transfer. This study investigates the use of TL in IoT security in two different ways: transferring knowledge for generating suitable intrusion algorithms for new devices, transferring knowledge for detecting new types of attacks. In the literature, TL is proposed as a promising approach for securing IoT systems, since they consist of different components such as devices, wireless sensor networks (WSNs), and cloud computing [4]. However, to the best of our knowledge, there is no study that covers both settings in the research area yet.

One of the main characteristics of LLNs is that it interconnects a number of heterogeneous, resource-constrained devices. Therefore, the main hypothesis in *transferring knowledge for new devices* is that an intrusion detection algorithm

developed for a particular type of device might not be suitable for other types of devices. Moreover, developing suitable intrusion detection algorithms for each type of device is a costly approach. Hence, by using TL, the knowledge of an intrusion detection algorithm could be transferred for generating new algorithms for devices with different constraints. For example, an effective algorithm on a device could be transferred to a more resource-constrained device. With this approach, it is expected to find a good balance between accuracy and resource consumption in a shorter time than learning an intrusion detection algorithm from scratch for a new device.

In *transferring knowledge for new types of attacks*, the knowledge is transferred from one domain to a new domain. With the increasing popularity of IoT networks, we expect new attacks to emerge. So, the developed intrusion detection algorithms might not be effective enough on such new attacks. Therefore, in this study, the knowledge obtained during the automatic generation of an intrusion detection algorithm by using machine learning techniques is proposed to be transferred for detecting new types of attacks. With this approach, it is expected to obtain effective intrusion detection algorithms for new attacks in a shorter time than traditional learning.

Here, genetic programming (GP) is employed for evolving intrusion detection algorithms due to their capability of exploring search space efficiently for complex environments such as LLNs. Moreover, it allows us to manually analyze evolved detection programs to some degree. Last but not least, it eases performing validation and testing automatically by producing detection programs written in C, hence they could be run directly on simulated devices running on Contiki operating system. GP is a population-based optimization algorithm. Hence, it outputs a group of candidate solutions for the problem at hand, and the best one is usually selected for testing. This characteristic of GP allows us to transfer a group of individuals in the last population in evolution to a new task/domain.

The contribution of this current study could be summarized as follows:

- The use of genetic programming is explored in detecting specific attacks against RPL-based IoT networks. The results show that GP could evolve effective algorithms for detecting rank, DODAG Information Solicitation (DIS) flooding, version, and worst parent attacks in a given time.
- Transfer learning is explored for detecting new types of attacks on three scenarios: single-to-single, single-to-multi, multi-to-multi. In single-to-multi scenario, single corresponds to an environment with a single attack, where learning takes place, and multi corresponds to a network with multiple attacks, where the learned detection algorithm is transferred to. In all scenarios, the positive effect of transfer learning is clearly shown. TL reduces the learning time and produces more effective detection algorithms.
- Finally, the application of transfer learning is investigated on generating separate algorithms for the different types of devices with different constraints. The results show that the proposed approach produces better results and converges faster than the traditional approach, which is important for putting devices/networks in operation in a timely manner.
- The use of transfer learning on IoT security is firstly explored for the following two settings: *transferring knowledge for new types of attacks* and *transferring knowledge for new devices*.

The paper is organized as follows. The overview of RPL and internal attacks against RPL is given in Section II. The background information about genetic programming and transfer learning is also given in this section. Section III discusses related studies in the field of intrusion detection on RPL and in the area of transfer learning in IoT security. The proposed approach is given in detail in Section IV. Experimental settings and results are evaluated and discussed in Section V. Section VI discusses the limitations of the proposed approach and the possible future studies. Finally, Section VII concludes the findings of this study.

II. Background

A. RPL and Target Attacks

RPL is one of the most popular routing protocols for LLNs [3]. It is a distance-vector and source routing protocol based on building DODAG in order to represent the network topology. RPL is mainly proposed for supporting multipoint-to-point communication (MP2P), however, it also supports point-to-multipoint (P2MP) and point-to-point (P2P) communication. Each DODAG has a single root node. Hence, in a typical scenario, sensor nodes periodically send their information to the root node. The route from these sensor nodes to the root node is determined based on objective functions such as expected transmission count (ETX), hop count, and energy.

RPL has four types of routing control messages. The root node initially broadcasts DODAG Information Object (DIO) messages in order to create routes in an upward direction. By using DIO messages, a node determines a set of candidate parents, selects one of them, and determines its rank. Rank represents the position of a node with respect to the root node. The objective function specifies how a node computes its rank value for the selection of its parent. Destination Advertisement Object (DAO) messages are used for reversal route construction. DAO Acknowledgement (DAO-ACK) is used to acknowledge the receipt of a DAO message. DIS messages are sent when a new node wants to join the DODAG and asks for DIO messages from its neighbors.

Although RPL has some countermeasures against external attackers, it is still vulnerable to attacks from inside. Attacks against RPL are covered in three classes in the literature [5]: attacks against resources, attacks on topology, and attacks on traffic. In this study, the following four attacks are targeted. The detection of these attacks is investigated by using transfer learning. Even though transfer learning could be applied for detecting variations of existing attacks, here, the main aim is to detect new types of attacks by using transfer learning due to being a more complex problem.

- *Decreased Rank (DR):* In this attack, attacker nodes illegitimately advertise a lower rank value to other nodes in the network. This results in many legitimate nodes in the network connecting to the DODAG graph over attacker nodes. Hence, a great portion of the network traffic might pass through attackers, which can be the initial step for future harmful attacks.
- *DIS Flood (DF):* This attack aims at exhausting nodes' resources. The attacker nodes generate a large number of DIS messages to nodes in their neighborhood to consume their resources (e.g., energy) and to cause congestion in the network. In this attack scenario, the attacker sends 20 DIS packets consecutively.
- *Increased Version (IV):* While version number in DIO messages is increased by only the root node and propagated throughout the network for global repair in RPL, it is illegitimately increased and broadcast by attackers, which results in an unnecessary effort for rebuilding the graph. In this attack scenario, the attacker increases the value of the version field by one every time he sends a DIO packet.
- *Worst Parent (WP):* In order to forward incoming packets, a node chooses a parent node with respect to the objective function. However, it is not the case in this attack scenario. Attacker node contrarily prefers the worst parent to send or forward packets, which degrades the performance of the network (e.g., end-to-end delay, delivery ratio).

B. Genetic Programming

GP [6] is an evolutionary computation technique inspired by biological evolution. In its simplest form, it is based on the Darwinian *survival of the fittest* theory where individuals compete with each other for survival and reproduction in an environment that can only host a limited number of individuals [7]. It is a population-based search algorithm in order to evolve better individuals that correspond to candidate solutions for a targeted problem at each generation. It applies genetic operators such as crossover, mutation, and selection on the individuals in order to provide better solutions in the new population and to find the optimum (or close to the optimum) solution for the problem at hand. Since GP is capable of representing different types of complex problems, we see a wide variety of successful GP applications in the literature.

LLNs are complex environments due to their special characteristics such as having low power nodes and lossy links. Moreover, different trade-offs should be considered while designing a security solution for this complex environment such as accuracy, being lightweight and so stability. Humans are not particularly adept at selecting good choices when complex trade-offs have to be made. Mobility makes this environment more difficult to perceive. While RPL was not designed with mobility in mind, real-life applications could include mobile nodes. Evolutionary computation (EC) based approaches could be suitable for such complex and/or dynamic environments. Among various artificial intelligence techniques that have been proposed for intrusion detection, EC is considered one of the most promising approaches. It makes fewer assumptions about the solution space as other heuristic computation techniques. Intrusion detection programs derived using GP are open to manual analysis to some degree. Moreover, we can directly derive detection programs written in C, which eases to run them directly on Contiki. Last but least, since it is a population-based approach, it allows us to transfer a group of individuals to a new domain/task at the end of one run. These characteristics are among the main motivations behind using GP in this research. Furthermore, by using a multi-objective evolutionary algorithm, evolving detection programs that are both effective and also efficient (i.e., energy-aware) is explored.

The general steps of GP algorithm are given below. The algorithm starts with generating the first population. The individuals, which represent candidate solutions for the problem at hand, are usually generated randomly in the first population. Then, these individuals are transformed into a new, hopefully for the better, population of individuals by using genetic operators. The better the fitness value of an individual is, the more likely it is to be selected for the application of genetic operators. Fitness function represents how good or how close to the optimal the candidate solution is. In practice, since the optimal solution cannot be achieved in a timely manner, the algorithm is generally run up to the maximum number of generations or up to the attainment of a solution with sufficient quality.

Algorithm 1: General steps of GP

```
1 Initialize population;
2 repeat
3     Evaluate the fitness of each individual;
4     Rank the population according to fitness values;
5     Apply genetic operators (crossover, mutation, etc.)
        and reproduce new population;
6 until a termination criterion is satisfied;
7 return best-of-run individual
```

In GP, individuals are represented as trees. At each iteration, the individuals are evaluated by using the fitness function and selected for genetic operators. The selection mechanism provides a great opportunity for fitter individuals to survive by picking out individuals from the current population. One or two individuals, depending on the type of operator, are selected as the parent. Two main genetic operators applied upon parent individuals are crossover and mutation. In the crossover operator, two offspring are created by replacing the sub-trees of parent individuals. In the mutation operator, a mutation point in a parent tree is randomly selected and the sub-tree already rooted there is substituted by a new, randomly generated sub-tree. Mutation introduces diversity into the population. As the final step of each generation, individuals who will survive in the next generation are selected based on their fitness values. Please see the tutorial in [8] for further information on genetic programming.

C. Transfer Learning

The volume of training data is one of the most important factors that affect the learning capability of a machine learning algorithm. In that sense, while supervised learning is generally preferred over unsupervised learning, labeling a high volume of data is tiring, time-consuming, and even prone to errors in the case of manual labeling. A semi-supervised algorithm can address this issue as it needs a large amount of unlabeled data instead. However, since gathering unlabeled data can be also unrealistic for some problems, traditional machine learning may not be effective to solve them. Transfer learning is a way to handle this problem by transferring the information from a source domain to a target domain [9] that has limited data.

A domain $\mathcal{D}$ is defined by a feature space $\mathcal{X}$ and a marginal distribution $P(X)$ (i.e., $\mathcal{D} = \{\mathcal{X}, P(X)\}$). Here, X denotes an object (instance) set (i.e., $X = \{x \mid x_i \in \mathcal{X}, i = 1, 2, \ldots, m\}$). A task, however, $\mathcal{T}$ is composed of a label space $\mathcal{Y}$ and an objective predictive function f (i.e., $\mathcal{T} = \{\mathcal{Y}, f\}$). Given a source domain ($\mathcal{D}_S$), a source task ($\mathcal{T}_S$), a target domain ($\mathcal{D}_T$), and a target task ($\mathcal{T}_T$); transfer learning is defined as learning an efficient objective predictive function (f) for $\mathcal{D}_T$ by using information from $\mathcal{D}_S$ and $\mathcal{T}_S$, where $\mathcal{D}_S \neq \mathcal{D}_T$ or $\mathcal{T}_S \neq \mathcal{T}_T$. Traditional machine learning tasks deal with the same task and domain in training and testing data (i.e., $\mathcal{D}_S = \mathcal{D}_T$ and $\mathcal{T}_S = \mathcal{T}_T$). The model is obtained by using training data and then is applied to testing data in order to evaluate the model. However, transfer learning is quite different from the traditional ML in a way that task and domain in training and testing data can be different.

Based on the definitions above, $\mathcal{D}_S \neq \mathcal{D}_T$ implies that either feature space or marginal distribution between source and target must be different from each other (i.e., $\mathcal{X}_S \neq \mathcal{X}_T \vee P(X_S) \neq P(X_T)$). In homogeneous transfer learning, the feature space of source and target domains must be the same (i.e., $\mathcal{X}_S = \mathcal{X}_T$); otherwise, (i.e., $\mathcal{X}_S \neq \mathcal{X}_T$), it is called heterogeneous transfer learning [10]. Another type of TL is the one where conditional probability distributions are different (i.e., $P(Y_S|X_S) \neq P(Y_T|X_T)$). The output label space can also be different, $\mathcal{Y}_S \neq \mathcal{Y}_T$. If one of these two conditions is satisfied, then we can say that source and target tasks are different (i.e., $\mathcal{T}_S \neq \mathcal{T}_T$.)

III. Related Work

There is a good amount of works in the literature in order to prevent and detect external and internal attacks against RPL. Besides intrusion detection systems, there are security protocols which are grouped into cryptography-, trust-, and threshold-based solutions in the literature for enhancing the RPL security [11]. However, here, we mainly focus on studies proposed for detecting and mitigating RPL specific attacks.

SVELTE [12] is the first intrusion detection system (IDS) for IoT, which employs the combination of signature and anomaly-based systems. Since then, researchers mainly focus on mitigating or detecting particular types of attacks against RPL. While, rank and version number attacks are among the most analyzed attacks against RPL, RPL-specific flooding attacks such as DIS flooding [13], newly developed attacks such as DIO suppression [14] need more attention [11].

Version number attacks have been analyzed in [15][16]. Both analyses show that the attack increases the network overhead considerably. Moreover, it is shown that the more the attacker is away from the root node, the more damage it could give to the network [15]. Therefore, in a technique for mitigating this attack type [17], the version number update messages coming from nodes other than the root node and nodes in its neighborhood are dropped. In order to validate other update messages, the majority of nodes with better rank values are expected to have the same version number. In [18], a separate network consisting of the monitoring nodes is constructed by taking advantage of the multiple instance feature of RPL. Hence, each monitoring node shares its information in order to detect version number attacks. The effects of the rank attack are analyzed in [19]. It is shown that implementing attacks in an area with a high forwarding load such as where nodes have a limited choice of parents has a higher impact on the network. An analytical model of Sybil attackers is proposed to increase the evasiveness of Sybil attack by using artificial bee colony algorithm [20]. Then, a lightweight threshold-based Sybil attack detection system is proposed for the bounded region, scattered, and mobile attacker scenarios by introducing new fields into DIO messages such as control message counter and timestamps.

While most of the proposed IDS for RPL is anomaly-based, there are few specification-based IDS [21][22] in the literature. In the last few years, few machine learning-based IDS have been proposed for RPL. Compression Header Analyzer Intrusion Detection System (CHA-IDS) [23] employs features collected from IPv6 over Low-Power Wireless Personal Area Network (6LoWPAN) compression header, then applies six classification algorithms after the feature selection phase. Even though the proposed approach performs better than SVELTE [12] and [24], it has high power and memory consumption. Moreover, it is proposed for only WSN-based attacks. In [25], a neural network-based approach is applied for detecting hello flood, version, and rank attacks, and it is shown to be very effective in detecting hello flood attacks. In a recent study [26], intrusion detection algorithms are automatically generated by using evolutionary computation. Both central and distributed intrusion detection architectures are explored. Recently, a neural network-based intrusion detection system is proposed for RPL [27]. The effects of link-layer features on detecting RPL attacks were firstly explored and shown to reduce false positives.

Modeling the normal behavior of IoT devices has also been exploited to develop intrusion detection systems targeting general attacks. In [28], the operations of devices are monitored, and a model that determines the 'normal' behavior of a device is extracted by using system statistics like central processing unit (CPU) usage, memory consumption of the monitored devices. They employ methods based on neural networks, linear regression, and recurrent neural networks. In another anomaly-based security model [29], changes in the energy pattern of IoT devices are monitored to detect cyber and physical attacks in the network. Here, a convolutional neural network, one of the popular deep learning (DL) algorithms, is employed to identify deviations from the normal behavior of

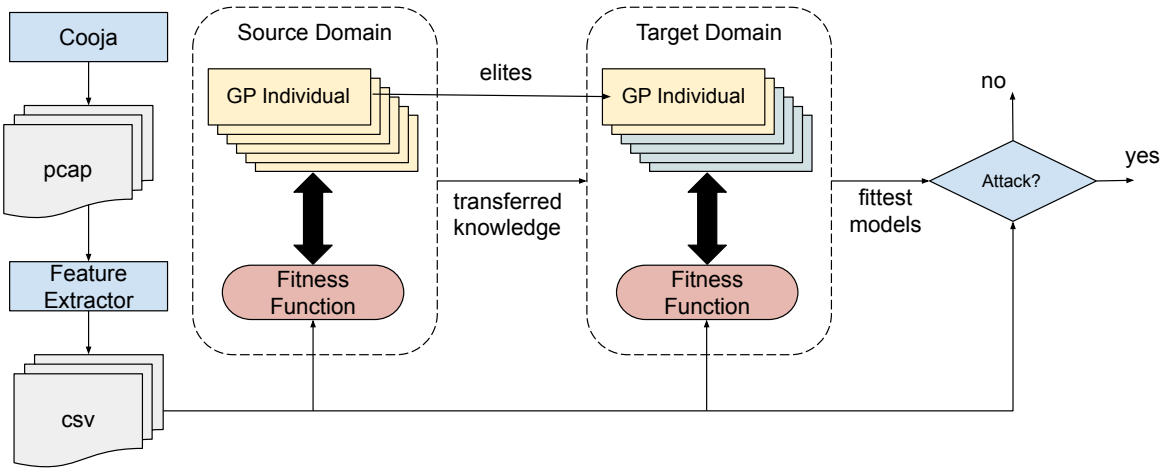


Fig. 1: The conceptual scheme of the proposed approach.

the system.

In the literature, transfer learning has been applied in various problems such as text classification, sentiment classification, image classification, WiFi localization, anomaly detection, spam filtering up to date [9]. However, there are only a few studies that explore the application of transfer learning on IoT security. Very recently, a DL-based attack detection system is proposed [30]. Due to the difficulty of having labeled data, they propose a transfer learning approach in which the latent representation of an autoencoder trained on a labeled dataset is transferred to another autoencoder trained on an unlabeled dataset. It aims to detect general attacks such as scanning, Transmission Control Protocol (TCP) flooding, and User Datagram Protocol (UDP) flooding, hence no specific IoT protocol is targeted. Nine datasets from N-BaIoT are employed [31]. 115 statistical features are extracted from the packet stream. Area Under Curve (AUC) score is used for performance comparison. Furthermore, the effectiveness of transferring information and processing time is analyzed. The results show that the proposed TL-based approach performs better than the baseline DL technique. Although the target datasets include attacks that do not exist in the source datasets, the effect of the proposed approach on only the new attacks is not shown explicitly in the results.

In another recent study [32], transfer learning is employed in order to encode high-dimensional features applied for multi-class classification for building a binary classifier. A detection model is trained in order to detect four attacks (i.e., denial-of-service (DoS), distributed DoS (DDoS), reconnaissance, and information theft) in IoT networks. In the study, no specific IoT protocol is targeted, but features related to internet protocol (IP), TCP, UDP, hypertext transfer protocol (HTTP) are extracted. It is shown that high accuracy is obtained for binary classification by using TL. In N-BaIoT [31], a network-based botnet detection for IoT, transferring knowledge of the detection model to other identical devices is left as future work.

In this study, the aim is not only to detect attacks against RPL, but also to develop suitable solutions for this environment: developing effective detection algorithms in a timely manner for new attacks and developing efficient algorithms for new devices. To the best of our knowledge, there is no such study in the literature. While anomaly-based systems [12][18] could be effective against new attacks, their performance on unknown attacks has not been evaluated. Furthermore, although the energy consumption of the proposed algorithms is given in some studies [12][23], their applicability to different devices is not taken into account as done in this study. Last but not least, although there are a few recent studies [30][32] that employ transfer learning for IoT security in the literature, these studies do not target RPL attacks. They do not use transfer learning both for new attacks and for new devices as in the current study. Moreover, the deep TL approach [30] only focuses on attacks belonging to the same class (i.e. DDoS).

IV. Evolving Intrusion Detection Algorithms

In this study, the use of transfer learning for intrusion detection in resource-constrained IoT networks is investigated. The conceptual scheme of the proposed approach is depicted in Fig. 1. Firstly, intrusion detection algorithms are evolved by using GP for the source domain. The features used in individuals are extracted from networks simulated by Cooja, a Java-based network simulator of sensor nodes running the Contiki operating system [33]. Please note that the first population in the source domain is initialized randomly. However, the first population in the target domain is constructed by transferring knowledge from the source domain. The fittest individuals learned in the source domain are transferred into the target domain. In this regard, we adopted *FullTree* approach proposed in [34], which selects a given percentage of best individuals of the last generation for transferring to the target domain. With transfer learning, it is expected to evolve better individuals in a shorter time than the traditional approach. Moreover, it generally performs better initial and final performance compared to learning without transfer.

In the source task/domain, the initial population is randomly constructed. The individuals in the population are candidate algorithms for detecting RPL-specific attacks. Each individual is represented by a GP tree. An exemplar GP tree is shown in

Fig. 2. Terminal (or leaf) nodes represent the features collected from RPL control and data packets. The feature set proposed in our previous study [35], which covers most of the features related to RPL control messages and data packets, together with randomly generated numbers has been employed as leaf nodes. The features are listed in Table I. While data-related features include information about data packets received by the root node in a time interval, topology-related features include information about routing control messages received by the root node, which could give useful insights for detecting RPL-specific attacks.

TABLE I: The feature set

No	Feature	Description
1	COUNT_DATA	Number of data packets
2	COUNT_DIO	Number of DIO packets
3	COUNT_DIS	Number of DIS packets
4	COUNT_DAO	Number of DAO packets
5	COUNT_DAOACK	Number of DAO-ACK packets
6	MAX_VERSION	Max. of versions
7	MIN_VERSION	Min. of versions
8	AVG_VERSION	Avg. of versions
9	MAX_RANK	Max. of ranks
10	MIN_RANK	Min. of ranks
11	AVG_RANK	Avg. of ranks
12	MAX_INTERVAL_MIN	Max. of DIO interval min.
13	MIN_INTERVAL_MIN	Min. of DIO interval min.
14	AVG_INTERVAL_MIN	Avg. of DIO interval min.
15	MAX_INTERVAL_DOUB	Max. of DIO interval doublings
16	MIN_INTERVAL_DOUB	Min. of DIO interval doublings
17	AVG_INTERVAL_DOUB	Avg. of DIO interval doublings
18	MAX_RPL_INS	Max. of rpl instances
19	MIN_RPL_INS	Min. of rpl instances
20	AVG_RPL_INS	Avg. of rpl instances
21	MAX_TIME_BTW_DATA	Max. time between data packets
22	MIN_TIME_BTW_DATA	Min. time between data packets
23	AVG_TIME_BTW_DATA	Avg. time between data packets
24	MAX_TIME_BTW_DIO	Max. time between DIO packets
25	MIN_TIME_BTW_DIO	Min. time between DIO packets
26	AVG_TIME_BTW_DIO	Avg. time between DIO packets
27	MAX_TIME_BTW_DIS	Max. time between DIS packets
28	MIN_TIME_BTW_DIS	Min. time between DIS packets
29	AVG_TIME_BTW_DIS	Avg. time between DIS packets
30	MAX_TIME_BTW_DAO	Max. time between DAO packets
31	MIN_TIME_BTW_DAO	Min. time between DAO packets
32	AVG_TIME_BTW_DAO	Avg. time between DAO packets
33	MAX_TIME_BTW_DAOACK	Max. time between DAO-ACK packets
34	MIN_TIME_BTW_DAOACK	Min. time between DAO-ACK packets
35	AVG_TIME_BTW_DAOACK	Avg. time between DAO-ACK packets

Mathematical operators such as addition, multiplication, and subtraction, together with the logical operators such as 'and', 'or' are used as non-terminal (intermediate) nodes to form a GP tree, as illustrated in Fig. 2. The root node, however, is constrained to be a comparison or logical operator so that the evolved tree returns a boolean expression. Each individual produces an if statement for detecting the attack and raising an alarm.

The intrusion detection algorithm corresponding to the tree in Fig. 2 is given below:

```
1 if ((COUNT_DIS + MAX_RANK) < COUNT_DIO) &&
    (COUNT_DATA == MAX_RPL_INS) then
2 |   alert(intrusion)
3 end
```

The parameter settings including the operators and functions are given in Table II. The number of generations is set as the termination criterion of evolution in the experiments. In order to evaluate evolved algorithms, accuracy is employed as the fitness function. As for the implementation of GP algorithm, a Java-based evolutionary computation toolkit (ECJ) [36] is used in the experiments. The other settings of GP not listed in the table are the default parameters of ECJ.

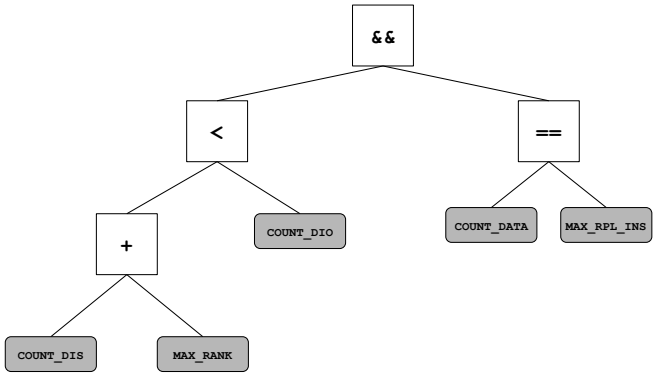


Fig. 2: An example GP-tree.

TABLE II: GP parameters and their values.

Parameters	Value
Non-terminals	+, -, *, /, sin, cos, log, ln, sqrt, abs, exp, ceil, floor, max, min, pow, mod, $<$, $\leq$, $>$, $\geq$, $==$, $!=$, and, or
Terminals	features in [35] and rnd(0,1)
Generations^a	1000 (source), 500 (target)
Generations^b	100 (source), 25 (target)
Population Size^a	100 (source), 50 (target)
Population Size^b	20 (source), 20 (target)
Elite Fraction	10%^{a}, 20%^{b}
Crossover probability	0.9
Mutation probability	0.1
Selection strategy	Tournament with the size of 7
Max. depth of the tree	17

^{a}: transferring knowledge for new types of attacks
^{b}: transferring knowledge for new devices

The running time of evolving an intrusion detection algorithm is proportional to the running time of genetic programming, which is mainly determined by the individual/generation size and the fitness evaluation cost. Hence, the time complexity of the proposed approach is $O(G \times I \times F)$; where G, I, and F denote the number of generations, the number of individuals evaluated in each generation, and the fitness evaluation respectively. In the fitness evaluation of each individual, the average of running the individual on five network simulations, in each which consist of 300 records collected in 60s window intervals, is calculated. Please note that each individual is extracted by tree traversal in GP. If the number of nodes in the individual tree is n, the time complexity of the tree in-order traversal by using traditional algorithms is $O(n)$. Here, the number of nodes in the tree is limited by the maximum tree depth as given in Table II.

Depending on the placement strategy (i.e., centralized, distributed, and hybrid), the evolved IDS algorithms can be deployed to any node in a network. In this study, a central intrusion detection system is proposed, where the evolved algorithm is deployed on the root node. So, the features are

collected from packets sent from/received by the root node. As stated earlier, the applicability of the transfer learning-based approach is investigated under two different settings:

Transferring knowledge for new types of attacks: Here, the effect of transfer learning on detecting new RPL attacks is explored. For this purpose, RPL attacks other than those used in learning in the source domain are employed at the target domain. Three different scenarios are explored in this setting. In the first scenario, called *single-to-single*, a different single attack takes part in both source and target domains. In the second scenario, called *single-to-multi*, while the source domain evolves a detection algorithm for a single attack, it is transferred for detecting multiple attacks in the target domain. Finally, in the third scenario, called *multi-to-multi*, both domains include different combinations of multiple attacks.

In this setting, the same feature space is used in both source and target problems, i.e. $\mathcal{X}_S = \mathcal{X}_T$. However the marginal distribution is different from each other as the source and target attacks are different, i.e. $P(X_S) \neq P(X_T)$, which changes data distribution. The same output label is used for both tasks, i.e. $\mathcal{Y}_S = \mathcal{Y}_T$. Lastly, the predictive function used in both source and target problems is the same. Based on these definitions, source and destination tasks are the same, source and target domains are different from the transfer learning point of view.

Transferring knowledge for new devices: In this setting, the detection algorithms are transferred to a new environment where a new type of devices takes part. Here, both detection ability and energy consumption of evolved detection algorithms are taken into account on transferring. Here, the main motivation is the need to deploy intrusion detection algorithms on more resource-constrained devices. Therefore, while the generated algorithms in the source domain are evolved by using only the accuracy of the algorithms, in the target domain both accuracy and energy consumption of evolved algorithms are taken into consideration in the fitness function. Therefore, the problem at hand becomes a multi-objective problem with the goals of higher accuracy and lower energy consumption.

In the literature, a great deal of effort has been put in order to extend evolutionary-based algorithms for solving multi-objective problems. Now, evolutionary-based multi-objective algorithms (EMOAs) are known to be very successful in finding well-converged and –diversified solutions. In contrast to single-objective problems where the best individual is found for that objective; for the multi-objective problem, the solution is usually not unique, but a set of optimal solutions called Pareto-dominant solutions. To define formally, a solution (say $\boldsymbol{u}$) is a Pareto-dominant solution in comparison to other solution (say $\boldsymbol{v}$) (denoted as $\boldsymbol{u} \prec \boldsymbol{v}$) if its objective set, $f(\boldsymbol{u})$, is partially less than that of $\boldsymbol{v}$, $f(\boldsymbol{v})$ for a minimization problem, i.e., $\forall i : f_i(\boldsymbol{u}) \leq f_i(\boldsymbol{v}) \wedge \exists i : f_i(\boldsymbol{u}) < f_i(\boldsymbol{v}) \mid i \in \{1, \ldots, k\}$; where k is the number of objectives. Therefore, Pareto-dominant solutions are better than non-dominant solutions in every objective but not comparable to each other. Here, Non-dominated Sorting Genetic Algorithm II (NSGA-II) [37], one of the most popular Pareto-based EMOAs, is employed. In this algorithm, in contrast to the fitness proportionate selection in single-objective GP, Pareto ranking and crowding distance measurements of candidate solutions obtained from their objective values, are taken into account for selecting an individual to survive in next generations.

In this setting, the same type of RPL attacks takes part in both domains; however, energy consumed by an intrusion detection agent becomes more critical. In order to simulate a more constrained device, the frequency of sending data packets sent from sensor nodes is increased, which also increases the network traffic to be handled by the root node. While the time interval is 15 s in the source domain, it is set as 5 s in the target domain. Since the root node consumes more energy to handle the increased traffic, the intrusion detection algorithm is expected to consume less energy in order to tolerate additional energy cost caused by the data packets without sacrificing its detection accuracy, if possible. So, even though the same input feature space and the same attacks are used in both source and target domains (i.e., $\mathcal{X}_S = \mathcal{X}_T$), the marginal distribution becomes different (i.e., $P(X_S) \neq P(X_T)$). Again, the same output space is employed (i.e., $\mathcal{Y}_S = \mathcal{Y}_T$). However, unlike the previous setting, energy consumption is added as another objective for the target problem, which makes the source and target task different. To sum up, from the transfer learning perspective, both domain and tasks are different in this setting.

V. Experimental Results

A. Simulation Environment

In this study, Cooja simulator [38] is used to simulate IoT networks. Cooja is the most used simulator in the literature. In addition, since it includes RPL implementation, it is preferred to be used in this study. It is capable of simulating wireless sensor networks consisting of different mote types. Here, Zolertia Z1 platform is adopted as a mote type in all simulations due to its bigger read-only memory (ROM) capacity than other platforms. Targeted attacks described in Section II-A are implemented by using the *RPL attacks framework* [39]. In order to measure the power consumption of evolved algorithms, Powertrace [40] tool is integrated into the Cooja simulator.

In the experiments, 15 different network topologies are employed for each scenario, where five topologies are used for training, and the remaining 10 topologies are used for testing. Each topology is run twice for generating benign and malicious traffic. Each topology is run for five hours and has 300 samples since features are collected every 60 s as given in the subsequent section. In simulations involving attacker nodes, malicious nodes attack the network during the whole simulation time; hence, the dataset is balanced. In order to see the multi-hop characteristics of RPL, at least 25 nodes are suggested to be used in RPL-based networks [41]. Therefore, 30 nodes are deployed in each simulation. There is a trade-off between the number of nodes and the simulation time. That's why bigger networks were not preferred in order to be able to run many simulations and to get statistically significant results. While 5 nodes ($\approx$ 15%) are set as an attacker performing malicious activities, the rest are set as sensor nodes, except the root node where the evolved detection algorithms are placed on. Sensor nodes send periodic data packets (every 15 s) as in a typical sensor network application.

B. Performance Evaluation

In the evolution of intrusion detection algorithms by GP, we adopted the window-based approach where the network traffic is collected in a time interval and the detection algorithm is triggered at the end of each interval. Otherwise, it would be too costly to run a detection algorithm after every incoming packet. Therefore, firstly, the optimal length of this time window is explored. To do that, GP is run five times for each attack type with five different network configurations in various time intervals (5, 15, 30, 60, and 90 s). The effects of various time intervals are compared by using detection accuracy. According to the overall findings, the detection performance proportionally increases with the time interval until 60 s, after which it slightly degrades. Therefore, the length of the time window is set as 60 s in the following experiments.

*1) **Transferring knowledge for new types of attack**:* In this setting, a detection algorithm evolved for RPL attack(s) at the source domain is transferred to the target domain where another RPL attack(s) is implemented. Different attack combinations are explored under three different scenarios: single-to-single, single-to-multi, and multi-to-multi. Each model is trained by using five different networks and tested by using 10 networks having varying topologies. GP is run 10 times for every setting due to the stochastic nature of evolutionary computation-based algorithms, then the evolved best detection algorithm of each run is used for testing. Pre-trained models are obtained at the 1000th generation. Then, the individuals of the last population are transferred to the target domain and the evolution goes on in the new environment. Here, instead of the whole population, only elite individuals (10% of the population) are transferred to the target domain and the remaining part of the population is randomly generated to protect diversity in the population. A detection algorithm is also evolved by using traditional learning (without transfer) for comparison. In traditional learning, GP is initialized with a random population. Both learning process continues for 500 generations and the best algorithms (with/without transfer) are obtained at the 500th generation for a fair comparison. Fig. 3 shows the average results of 10 GP runs for each setting. The accuracy of each transfer for different attack cases is represented in the figure. Each attack case is given on the x-axis. For instance, DR→DF denotes that the pre-trained models are evolved in an environment where DR takes place and is transferred to the target domain where DF attack is implemented. The results show that the proposed transfer learning-based approach yields better performance than the traditional learning approach on 8 out of 12 attack cases in the single-to-single scenario, 18 out of 24 attack cases in the single-to-multi scenario, and 26 out of 30 attack cases in the multi-to-multi scenario. The overall accuracy results obtained by averaging all the attack cases (given as dashed lines in the figure) also affirm the improvements yielded by the proposed TL-based approach.

In addition to the overall performance comparison, Wilcoxon signed-rank test has been applied with a statistical significance level of 95% to reveal whether there is a statistically difference between two approaches (i.e., TL-based and traditional) and if so, the number of simulations where the proposed TL-based approach exhibits superior or inferior performance than the traditional approach are listed. The total number of cases for each scenario where TL-based approach has shown superior (+), equal (=), and inferior (-) performance is given in Table III. The results show that findings obtained by the rank test are highly correlated with the overall performance given in Fig. 3. It is statistically proven that the proposed approach shows much better performance than the traditional approach. The results also support that there is no meaningful difference observable in cases, in which the average performances are very similar to each other as depicted in Fig. 3.

TABLE III: Performance of the proposed TL-based approach based on the signed Wilcoxon rank test.

Scenario	Total count (+/=/-)
single-to-single	7/4/1
single-to-multi	15/5/4
multi-to-multi	22/8/0
Sum	44/17/5

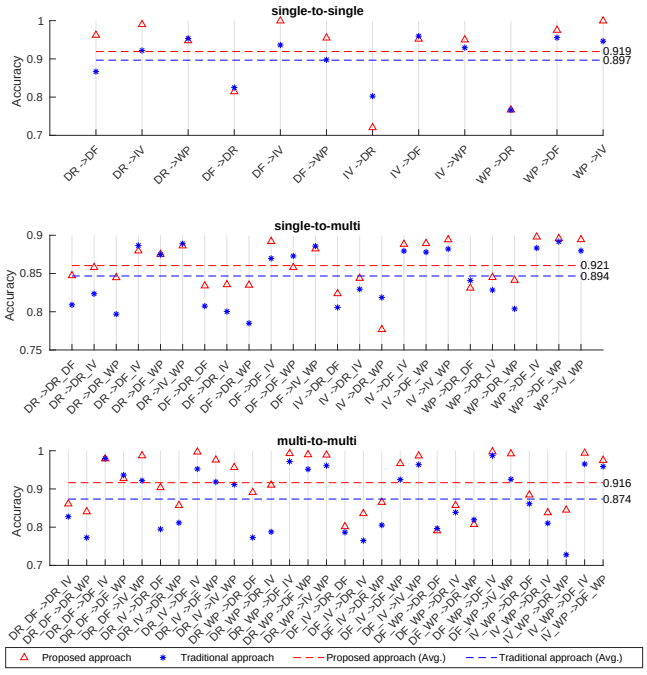


Fig. 3: Overall accuracy performance of the proposed transfer-based learning approach.

The attack cases where the proposed approach is less effective on average than the traditional learning are further analyzed through statistical box plots given in Fig. 4. In the single-to-single scenario (Fig. 4a), the proposed approach yields very similar detection accuracy on DR→WP case. It even outperforms the traditional approach on the remaining two cases (i.e., DF→DR and IV→DF) in which only two runs bring about a degradation to the overall performance. However, an apparent degradation is observable in IV→DR case. The overall testing results show that GP shows its best

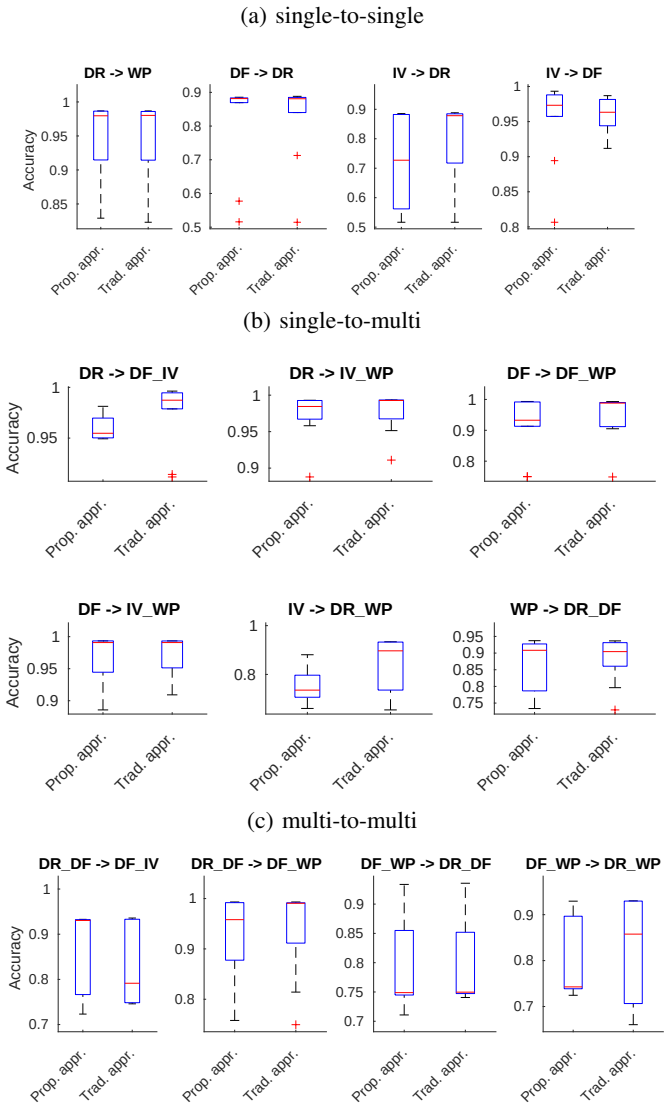


Fig. 4: Detailed comparative analysis.

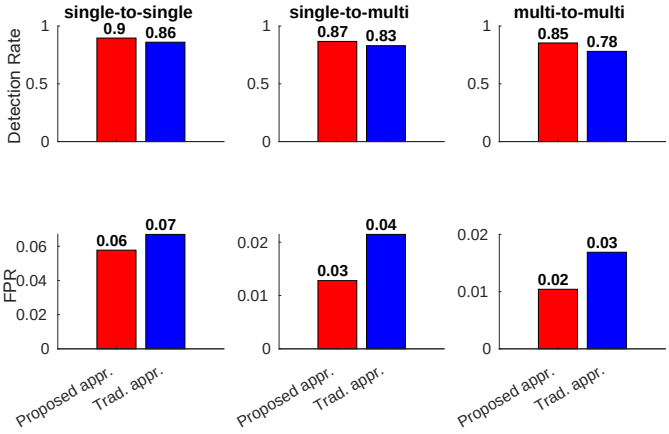


Fig. 5: Overall detection rate and FPR.

and worst performance on IV (93.4%) and DR (79.7%) attacks, respectively. It is observed that the feature set is less separable on DR than IV. This might make the situation more difficult in order to find optimal solutions when we transfer solutions from IV to DR. Therefore, a randomly initialized population could perform better in few simulations, which affects the average results as observed in the results. As shown in Fig. 7, the average accuracy at the initial generation obtained by using the traditional approach is higher than that achieved with the proposed approach. In the single-to-multi scenario (Fig. 4b), the proposed approach is highly competitive with the traditional approach on four cases (i.e., DR→DF_IV, DR→IV_WP, DF→DF_WP, and DF→IV_WP) but becomes inferior on the remaining cases (i.e., IV→DR_WP and WP→DR_DF). Finally, the proposed approach shows very similar performance for all cases in the multi-to-multi scenario (Fig. 4c).

The performance of the proposed approach is also analyzed in terms of the detection rate and false positive rate (FPR) metrics, which are important indicators in order to show how well the algorithm is able to distinguish malicious flows from benign flows. The overall detection and false positive rates are given for each scenario in Fig. 5. As it is shown in the figure, the proposed approach has a higher detection rate and lower false positive rate than the traditional approach for each scenario. Moreover, there is a trade-off between detection rate and false positive rate among the scenarios in general. A case-basis plot is given in Fig. 6 for detailed detection rate and FPR metric values of the proposed approach. It can be seen that the proposed approach assures a higher detection rate (or lower FPR) performance in almost all cases, but it slightly deteriorates when DR attack is implemented on either source or target domain overall. DR is observed to generally degrade the performance of the proposed approach.

In addition to the final performances, the pre-trained models are expected to have better initial performances. Here, the initial accuracy values are comparatively given in Fig. 7. The figure clearly shows that the proposed approach has better adaptability to a new domain overall. Concretely speaking, it becomes superior to the traditional approach on 7 out of 12 attack cases used in the single-to-single scenario, 22 out of 24 attack cases used in the single-to-multi scenario, and all attack cases used in the multi-to-multi scenario. A much higher performance difference is observed when multiple attacks take part in the source domain, since the pre-trained IDS model that is evolved from multiple attacks rather than a specific attack, is more generic and includes more knowledge.

The comparative convergence behavior captured at the target domain is also analyzed in Fig. 8. It is unsurprising to observe that the comparative convergence capability of the proposed approach is proportional to its initial performance by showing early convergence behavior. Hence, these results easily suggest that the time needed to evolve an intrusion detection algorithm can dramatically be decreased when the proposed TL-based approach is used, which is highly crucial in such a dynamically changing environment.

Finally, a comparison with the recently proposed cross-layer intrusion detection system (CL-IDS) [27] has been carried out in order to further evaluate the effectiveness of transferring knowledge for detecting new types of attacks. Since CL-IDS shares its attack dataset [42], it provides a platform for performing a comparison with other studies. CL-IDS is a

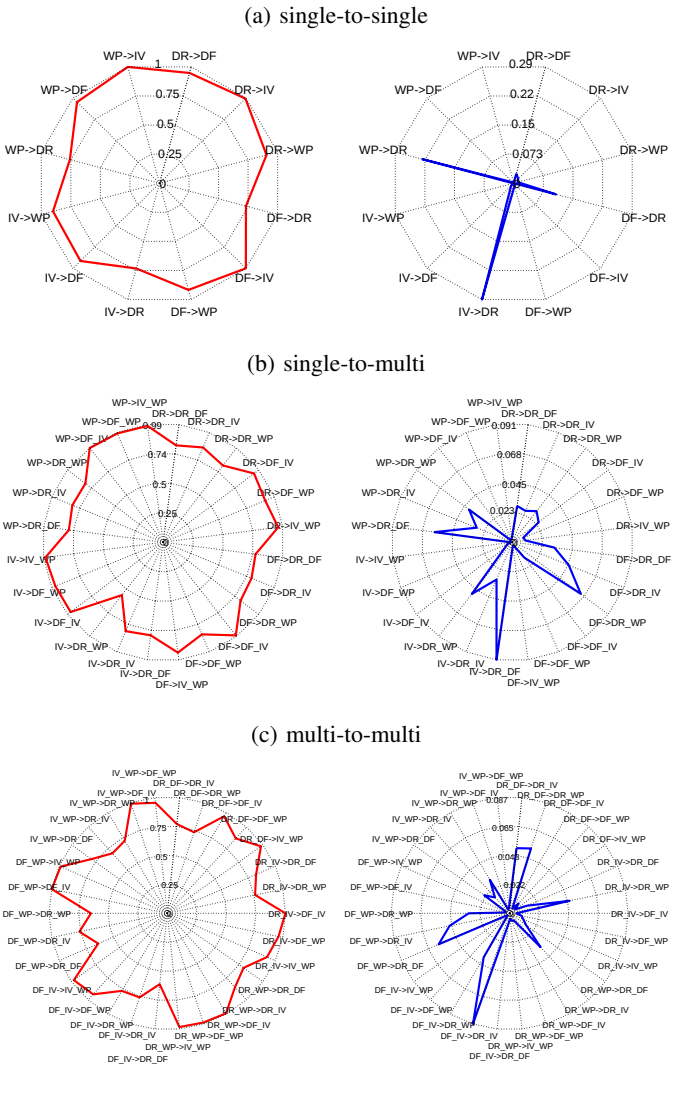


Fig. 6: Detailed detection rate (red lines) and false positive rate (blue lines) analysis.

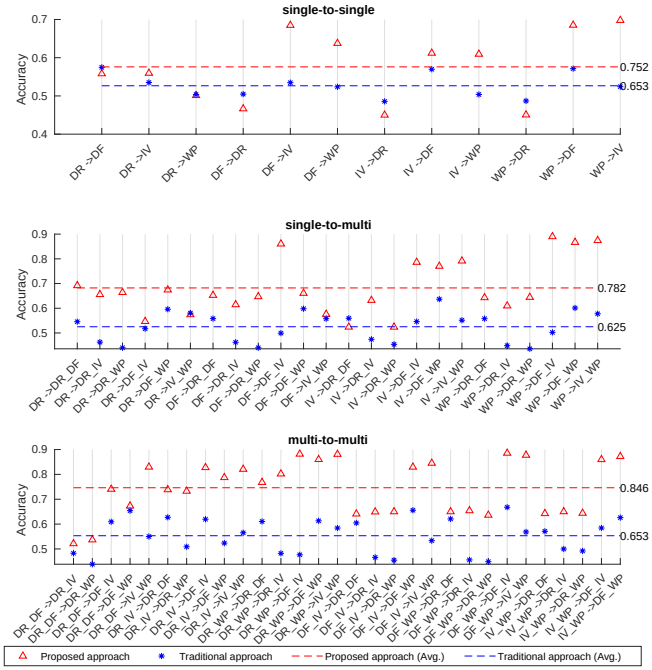


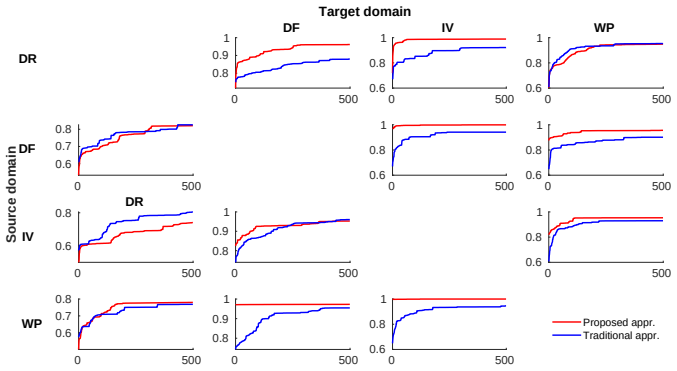
Fig. 7: Comparative accuracy values obtained at the initial generation.

neural network-based intrusion detection system that explores the effects of both routing and link-layer features on detecting RPL attacks. Here, DF, IV, and WP attacks are included in the evaluations as in [27]. To ensure a fair comparison, the same network traffic data used for the training and testing of the CL-IDS and the same length of the time window (5s) for collecting features is employed as given in [27]. The parameter setting shown in Table II is used for evolving intrusion detection systems. The proposed approach is run 10 times for each attack scenario. The performances are compared in terms of detection rate as given in [27] for each attack with varying attacker densities. The comparison results are given in Table IV. Please note that while CL-IDS uses all attacks in one training, because of the use of transfer-learning in our approach, separate models are generated separately for each attack type. For example, the test results of WP are the average testing results of the models trained for IV and DF, and transferred for detecting WP. Hence it is the average result of 20 runs for each attack.

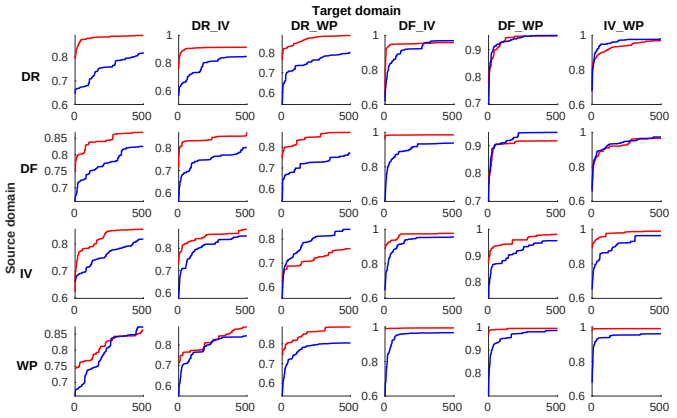
The comparison results show that the proposed transfer-learning approach is able to detect attacks with varying attacker densities at high detection rates. In addition, the detection performance increases proportionally to the attack density in general. While the proposed approach performs better than CL-IDS on IV and WP attacks, it is slightly inferior to CL-IDS on detecting DF attack. CL-IDS produces FPR between 0.16% and 0.61% depending on the features included in the training. While the former case has both the network layer and the link-layer features in training, the latter case includes the network layer feature set that contains the same features used in our approach. The false positive rate of our approach is given separately for each attack density in Table IV. As shown in the results, our approach outperforms CL-IDS in networks, where 10% and 20% of nodes are attackers. Even though the FPR in networks with a lower density of attackers is comparably high in our approach, this mainly results from the high false positive outputs of few runs. However, in most of the individual runs, high accuracy is achieved with 0% FPR.

To sum up, it is shown that the evolved intrusion detection algorithms could be transferred for detecting the new type of attacks by using transfer learning. It not only introduces more effective intrusion detection algorithms for new attacks but also achieves that in a timely manner. Both the initial and final performances of the transfer learning-based approach are better than the performances of the traditional approach. By transferring knowledge from one domain to a new domain, intrusion detection algorithms with high accuracy in a shorter time are obtained. With transferring more knowledge such as in multi-to-multi scenarios, better results are obtained. Therefore, the proposed approach is a good candidate for IoT

(a) single-to-single



(b) single-to-multi



(c) multi-to-multi

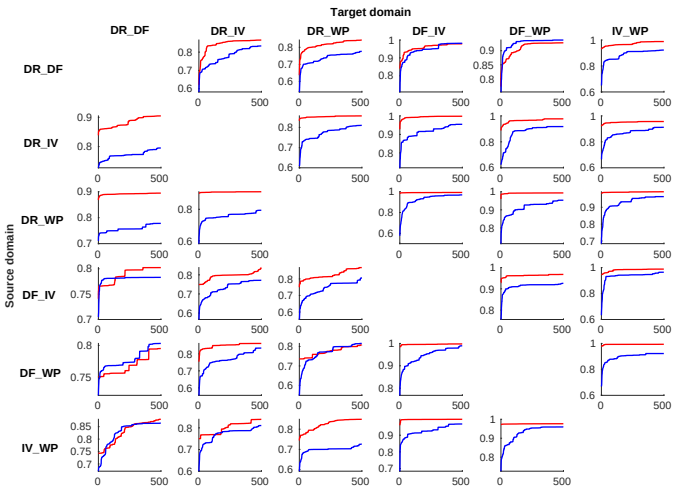


Fig. 8: Comparative convergence performance captured throughout the evolution.

networks that are becoming popular every day and become the target of new attacks.

*2) **Transferring knowledge for new devices**:* In this setting, two conflicting objectives are optimized simultaneously in order to evolve a detection algorithm that is energy-aware and having high accuracy. In order to measure the energy overhead of evolved algorithms, the energy consumption of the root node, where the central IDS is placed upon, is measured with and without IDS.

TABLE IV: Comparison of our approach with CL-IDS in [27].

Attack	Attack Density	CL-IDS [27] Det. Rate (%)	Proposed approach Det. Rate (%)	Proposed approach FPR (%)
DF	2%	**99.83**	99.75	3.40
	6%	**100.00**	97.32	2.53
	10%	**100.00**	99.70	**0.00**
	20%	**100.00**	99.41	**0.00**
IV	2%	86.66	**98.56**	4.24
	6%	92.99	**99.17**	**0.00**
	10%	98.58	**99.49**	**0.00**
	20%	94.83	**99.71**	1.61
WP	2%	N.A.	86.71	5.29
	6%	N.A.	95.58	**0.00**
	10%	96.91	**99.98**	**0.00**
	20%	99.42	**100.00**	**0.00**

N.A.: not available

It is worth stressing that elite individuals (algorithms) in this setting are regarded as the Pareto dominant algorithms where their objectives are better (i.e., gives high accuracy, low energy increase) than the non-dominant algorithms. Therefore, in this experiment, the Pareto-dominant algorithms found at the final population at the target domain are transferred to the testing environment. Contrary to the first setting, one network topology is used for training, while four different topologies are used in testing. The reason for that is to reduce the excessive computation time caused by a separate real-time simulation in which each candidate IDS is evaluated to obtain its energy consumption.

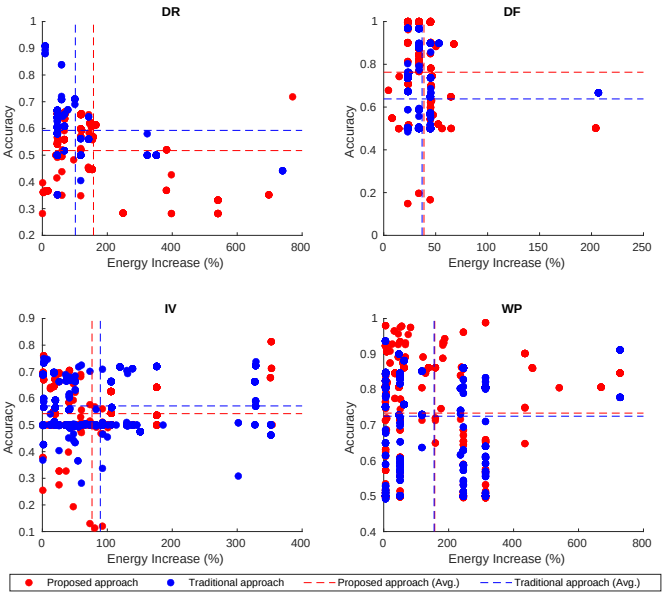


Fig. 9: Accuracy and energy increase performance of Pareto dominant algorithms.

The accuracy and energy consumption performances of the Pareto dominant algorithms are separately given in Fig. 9 for every attack case. The markers in this figure indicate the performance of these algorithms on four different networks used in testing. The overall performances of the Pareto dominant

algorithms are also comparatively given as dashed lines in this figure.

The results show that the proposed transfer learning-based framework gives better performance on all the attack cases except DR. As also shown in the previous results, DR generally degrades the performance of the evolved detection algorithms. When it is analyzed in detail, it is found out that the current feature set is not able to distinguish DR from benign traffic. For detecting DF, the traditional approach produces slightly better algorithms in terms of energy usage but they sacrifice detection accuracy; whereas, the proposed approach ensures much better detection accuracy. As for IV attack, the Pareto dominant algorithms evolved by the proposed approach are overwhelmingly better than the traditional approach according to energy usage, while they produce very similar accuracy performance on overall. Finally, both approaches give very similar results overall on WP attack.

As it is aimed to evolve IDS algorithms that have high detection accuracy but low energy usage, those located at the upper-left region in the objective space (Fig. 9) are highly attractive solutions. Therefore, a close view of the upper-left region is given in Fig. 10. As it is stated above, an IDS algorithm evolved by the traditional approach should be preferred for DR. However, for detecting DF efficiently, the proposed approach is superior to the traditional approach. Although both approaches evolve algorithms with similar energy usage for detecting IV, the TL-based approach obtains algorithms with higher accuracy. Lastly, for WP attack, while the proposed approach is much better in terms of accuracy (around 98%) but slightly worse in terms of energy increase (less than 1% increase) than the traditional approach.

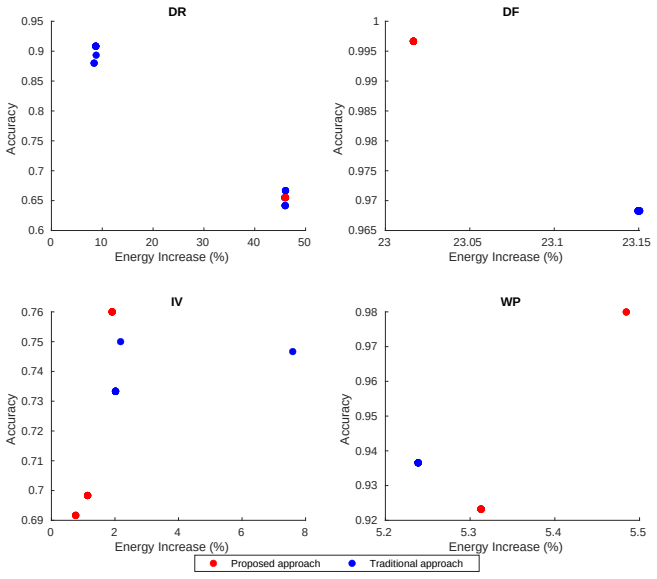


Fig. 10: Comparative best testing performance of Pareto dominant algorithms.

In order to analyze convergence and diversity of multi-objective solutions obtained by EMOA, the hyper-volume (HV) metric is usually employed. HV reveals the diversity and convergence performance of the Pareto-dominant individuals together by evaluating the space enclosed by the worst reference point and the Pareto dominant individuals. The higher HV value means a larger space in which the Pareto-dominant individuals occupy. Hence, higher HV values correspond to better convergence and diversity obtained by the Pareto-dominant individuals. Here, jMetal framework [43], a Java-based framework for multi-objective optimization with metaheuristics, is used for implementation and calculation of HV metric. It is worth stating that jMetal concatenates all the Pareto dominant individuals found, then generates reference points from them when there are no known reference points of the problem (also called as true Pareto front), which is the case in this setting. The comparative HV values are given in Table V. The results show that the proposed TL-based approach ensures a more convergent and diverse Pareto-dominant detection algorithm overall than those evolved by the traditional learning-based approach.

TABLE V: Comparative HV metric values.

	HV values	
	Prop. Appr.	**Trad. Appr.**
DR	3.84e-02	8.80e-02
DF	6.00e-02	1.20e-03
IV	1.24e-01	1.26e-02
WP	5.39e-01	5.34e-01

A comparative analysis is also made on convergence behavior of the proposed and traditional approaches here. Unlike the first setting where the accuracy of the best detection algorithm at each generation is shown, HV metric is evaluated to show the convergence behavior throughout the evolution. The comparative convergence curves with respect to this metric are given in Fig. 11 separately for every attack type addressed. It is worth restating that the higher the HV value is, the better an algorithm is able to converge to the true Pareto front of the problem. It can be seen from the figure that the convergence characteristic evaluated by HV verifies the findings shown in Fig. 9 and 10. HV curves reveal that the proposed transfer learning-based method converges better and earlier to the true Pareto front of all attack types except that of DR.

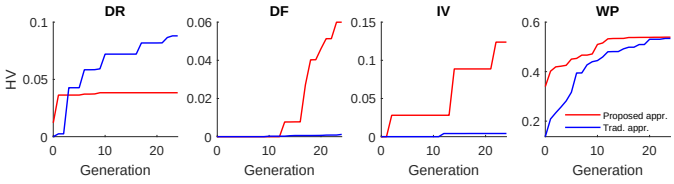


Fig. 11: Convergence characteristic on HV metric.

To sum up, while multi-objective evolutionary computation allows us to discover trade-offs between accuracy and energy consumption, with the use of transfer learning better trade-offs for a new environment could be obtained in a shorter time than the traditional multi-objective EC approach. For each attack type other than DR, better accuracy or better energy consumption is achieved in Pareto fronts of the transfer learning-based approach. Hence, the proposed approach could be used for developing suitable intrusion detection

algorithms for new devices introduced to the system with different constraints. The proposed approach complies with the heterogeneous nature of IoT networks.

VI. Limitations and Future Works

In this study, a central intrusion detection system is proposed for showing the effects of transfer learning on intrusion detection. Because of that, it could be a single point of failure. However, in real life, such central systems are always supported with backups. Moreover, other nodes could participate in intrusion detection and take place of the root node in case of a failure. For instance, one of the root's neighbors, which maintains local connectivity with the root node, could be selected randomly for the sake of security in case the root node fails.

The proposed system can be adapted to a distributed and/or hybrid architecture. The main motivation of using a centralized intrusion detection system in this work is its practicality rather than choosing it over a distributed and cooperative architecture. In [26], both central and distributed intrusion detection architectures are explored and shown that since the root node has more information, the proposed central intrusion detection system presents higher accuracy than IDSs running on nodes as stand-alone. On the other hand, by collaborating with other IDSs, higher accuracy could be obtained. However, in that case, the communication between IDS agents should be investigated, a trade-off between accuracy and communication cost should be discovered. As far as we know, this is not explored yet in the literature. Since the main motivation of this paper is to show the use of transferring knowledge on intrusion detection, it is applied on a central node for the sake of simplicity. However, due to taking into account multiple objectives simultaneously, the proposed approach could easily be enriched by including the communication cost as another objective besides accuracy and energy usage. This is left as future work.

GP could allow us to manually analyze the evolved programs to some degree. On the other hand, the code bloating problem of GP, which increases the size of individuals due to the long trees, could disrupt the readability of the evolved programs. This problem is not desired in many GP applications, hence the tree depth parameter is often decreased. However, it has a positive effect from the security point of view. By increasing the tree depth parameter deliberately here, the code bloating problem helps evolve longer and more complex programs that are more robust against adversarial attacks. On the other hand, this will increase the power consumption of the detection programs. As it is seen, the problem at hand has multiple objectives that conflict with each other. Multi-objective evolutionary computation is a good candidate for solving such complex problems having different trade-offs.

Although GP is employed in this study for various reasons stated above, DL is one of the most popular algorithms used for solving engineering problems. It has started to be used extensively in many domains, particularly in image processing and in computer vision. Unlike traditional machine learning techniques, which extract meaningful features from raw data and give them as input to the training process, DL techniques use raw data directly and extract features on their own. Furthermore, it is easily applicable for transferring knowledge to another domain/task. Nowadays, we have started to see the applications of DL in security as well. In the future, DL can be easily employed as a replacement for GP. This replacement can give us an advantage for saving on feature extraction time as we could give raw traffic data directly as input to DL.

VII. Conclusion

In this study, the use of the transfer learning approach is explored in order to detect RPL-specific attacks. To the best of the authors' knowledge, this is the first application of transfer learning in IoT security in the following two settings: *transferring knowledge for new types of attacks* and *transferring knowledge for new devices*.

In *transferring knowledge for new types of attacks*, the efficacy of transfer learning is analyzed for detecting different attacks, where three attack scenarios are included: single-to-single, single-to-multi, and multi-to-multi. In *transferring knowledge for new devices*, the use of transfer learning is explored for generating suitable intrusion detection algorithms for new types of devices with varying energy constraints. While only the accuracy of the generated intrusion detection algorithms is considered for the first setting, the energy consumption of these algorithms is also included in the second setting. The experimental results in both settings prove that the proposed transfer learning-based approach yields better performance than the traditional learning approach in most cases. In addition, it performs higher convergence speed and hence a shorter training time for the evolution of new detection algorithms in a new task/domain. This is especially important for IoT, where new attacks emerge every day and new types of devices can be added to the network. Although the proposed approach targets attacks against RPL, it could be easily adapted to other protocols and/or general attacks in IoT.

Acknowledgement

The authors would like to thank Erdem Canbalaban for sharing the dataset [27].

References

[1] "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.

[2] 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.

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

[4] M. A. Al-Garadi, A. Mohamed, A. Al-Ali, X. Du, and M. Guizani, "A survey of machine and deep learning methods for internet of things (iot) security," arXiv preprint arXiv:1807.11023, 2018.

[5] A. Mayzaud, R. Badonnel, and I. Chrisment, "A taxonomy of attacks in rpl-based internet of things," I. J. Network Security, vol. 18, pp. 459–473, 2016.

[6] J. R. Koza, "Genetic programming as a means for programming computers by natural selection," Statistics and Computing, vol. 4, no. 2, pp. 87–112, 1994. [Online]. Available: https://doi.org/10.1007/BF00175355

[7] A. E. Eiben, J. E. Smith et al., Introduction to evolutionary computing. Springer, 2003, vol. 53.

[8] R. Poli, W. B. Langdon, N. F. McPhee, and J. R. Koza, "Genetic programming: An introductory tutorial and a survey of techniques and applications," University of Essex, UK, Tech. Rep. CES-475, 2007.

[9] F. Zhuang, Z. Qi, K. Duan, D. Xi, Y. Zhu, H. Zhu, H. Xiong, and Q. He, "A comprehensive survey on transfer learning," 2019.

[10] K. Weiss, T. M. Khoshgoftaar, and D. Wang, "A survey of transfer learning," Journal of Big Data, vol. 3, no. 1, p. 9, 2016. [Online]. Available: https://doi.org/10.1186/s40537-016-0043-6

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

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

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

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

[15] A. Mayzaud, A. Sehgal, R. Badonnel, I. Chrisment, and J. Schönwälder, "A study of rpl dodag version attacks," in IFIP international conference on autonomous infrastructure, management and security. Springer, 2014, pp. 92–104.

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

[17] A. Arış, S. B. Ö. Yalçın, and S. F. Oktuğ, "New lightweight mitigation techniques for rpl version number attacks," Ad Hoc Networks, vol. 85, pp. 81–91, 2019.

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

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

[20] D. Karaboga and B. Basturk, "A powerful and efficient algorithm for numerical function optimization: artificial bee colony (abc) algorithm," Journal of Global Optimization, vol. 39, no. 3, pp. 459–471, Nov 2007. [Online]. Available: https://doi.org/10.1007/s10898-007-9149-x

[21] A. Le, J. Loo, K. K. Chai, and M. Aiash, "A specification-based ids for detecting attacks on rpl-based network topology," Information, vol. 7, no. 2, p. 25, 2016.

[22] U. Shafique, A. Khan, A. Rehman, F. Bashir, and M. Alam, "Detection of rank attack in routing protocol for low power and lossy networks," Annals of Telecommunications, vol. 73, no. 7-8, pp. 429–438, 2018.

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

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

[25] F. Y. Yavuz, Ü. Devrim, and G. Ensar, "Deep learning for detection of routing attacks in the internet of things," International Journal of Computational Intelligence Systems, vol. 12, no. 1, pp. 39–58, 2018.

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

[27] E. Canbalaban and S. Sen, "A cross-layer intrusion detection system for rpl-based internet of things," in In Proc. of ADHOC-NOW 2020, Lecture Notes in Computer Science, vol. 12338. Springer, 2020, pp. 214–227.

[28] F. Li, A. Shinde, Y. Shi, J. Ye, X. Li, and W. Song, "System statistics learning-based iot security: Feasibility and suitability," IEEE Internet of Things Journal, vol. 6, no. 4, pp. 6396–6403, 2019.

[29] F. Li, Y. Shi, A. Shinde, J. Ye, and W. Song, "Enhanced cyber-physical security in internet of things through energy auditing," IEEE Internet of Things Journal, vol. 6, no. 3, pp. 5224–5231, 2019.

[30] L. Vu, Q. U. Nguyen, D. N. Nguyen, D. T. Hoang, and E. Dutkiewicz, "Deep transfer learning for iot attack detection," IEEE Access, vol. 8, pp. 107 335–107 344, 2020.

[31] Y. Meidan, M. Bohadana, Y. Mathov, Y. Mirsky, A. Shabtai, D. Breitenbacher, and Y. Elovici, "N-baiot—network-based detection of iot botnet attacks using deep autoencoders," IEEE Pervasive Computing, vol. 17, no. 3, pp. 12–22, 2018.

[32] M. Ge, N. F. Syed, X. Fu, Z. Baig, and A. Robles-Kelly, "Toward a deep learning-driven intrusion detection approach for internet of things," arXiv preprint arXiv:2007.09342, 2020.

[33] Contiki-Ng, "contiki-ng/contiki-ng." [Online]. Available: https://github.com/contiki-ng/contiki-ng/wiki

[34] T. T. H. Dinh, T. H. Chu, and Q. U. Nguyen, "Transfer learning in genetic programming," in 2015 IEEE Congress on Evolutionary Computation (CEC). IEEE, 2015, pp. 1145–1151.

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

[36] ECJ, "A java-based evolutionary computation research system," https://www.cs.gmu.edu/ eclab/projects/ecj/, 2017.

[37] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: Nsga-ii," IEEE Transactions on Evolutionary Computation, vol. 6, no. 2, pp. 182–197, 2002.

[38] F. Osterlind, A. Dunkels, J. Eriksson, N. Finne, and T. Voigt, "Cross-level sensor network simulation with cooja," in Proceedings. 2006 31st IEEE Conference on Local Computer Networks, 2006, pp. 641–648.

[39] dhondta and bahmadh, "rpl-attacks: Rpl attacks framework," Jun. 2016. [Online]. Available: https://doi.org/10.5281/zenodo.55352

[40] A. Dunkels, J. Eriksson, N. Finne, and N. Tsiftes, "Powertrace: Network-level power profiling for low-power wireless networks," 2011.

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

[42] "Rpl attack dataset," https://wise.cs.hacettepe.edu.tr/projects/rplsec/, accessed: 2020-12-04.

[43] J. J. Durillo and A. J. Nebro, "jMetal: A java framework for multi-objective optimization," Advances in Engineering Software, vol. 42, no. 10, pp. 760–771, 2011.