DEFIDNET: A Framework For Optimal Allocation of Cyberdefenses in Intrusion Detection Networks

Sergio Pastrana[a,*], Juan E. Tapiador[a], Agustin Orfila[a], Pedro Peris-Lopez[a]

[a]*Computer Security (COSEC) Lab*
Department of Computer Science, Universidad Carlos III de Madrid
Avda. Universidad 30, 28911, Leganés, Madrid, Spain

Abstract

Intrusion Detection Networks (IDN) are distributed cyberdefense systems composed of different nodes performing local detection and filtering functions, as well as sharing information with other nodes in the IDN. The security and resilience of such cyberdefense systems are paramount, since an attacker will try to evade them or render them unusable before attacking the end systems. In this paper, we introduce a system model for IDN nodes in terms of their logical components, functions, and communication channels. This allows us to model different IDN node roles (e.g., detectors, filters, aggregators, correlators, etc.) and architectures (e.g., hierarchical, centralized, fully distributed, etc.) We then introduce a threat model that considers adversarial actions executed against particular IDN nodes, and also the propagation of such actions throughout connected nodes. Based on such models, we finally introduce a countermeasure allocation model based on a multi-objective optimization algorithm to obtain optimal allocation strategies that minimize both risk and cost. Our experimental results obtained through simulation with different IDN architectures illustrate the benefit of our framework to design and reconfigure cyberdefense systems optimally.

Keywords: Cooperative cyberdefense, evasion attacks, resilient cyberdefenses, adversarial settings.

1. Introduction

Intrusion Detection Systems (IDS) constitute a primary component for securing computing infrastructures. An IDS monitors activity and seeks to identify evidence of ongoing attacks, intrusion attempts, or violations of the security policies [1]. IDSs have evolved since the first model proposed in the late 1980s [2], and the current threat landscape makes the classical approach[1] for intrusion detection no longer valid. Moreover, intrusion detection must also deal with emerging paradigms in computing and communications. For example, performing detection in wireless nodes such as smartphones [3] or wearable sensing devices [4], requires lightweight procedures that do not consume much resources like energy or memory.

Detection paradigms and architectures have also evolved to cope with the requirements of complex network infrastructures. Rather than standalone components strategically placed to protect a complete network or system, the current trend is to rely on a distributed network of detection nodes. Intrusion Detection Networks (IDN) [5] are composed of different IDS nodes distributed among a network to perform local detection functions and sharing information with other nodes in the IDN. One of the major advantages of IDNs is that, since the detection function is distributed across different network locations, so is the workload. Classical intrusion detection components such as Snort [6] must be implemented in a single

*Corresponding author. Tel: +34 91 624 6260, Fax: +34 91 624 9429
Email addresses: `spastran@inf.uc3m.es` (Sergio Pastrana), `jestevez@inf.uc3m.es` (Juan E. Tapiador), `adiaz@inf.uc3m.es` (Agustin Orfila), `pperis@inf.uc3m.es` (Pedro Peris-Lopez)

[1]By classical approach we mean one where a standalone IDS protects a small local area network with no collaboration with any other IDSs.

device. Therefore, this host is in charge of gathering the data (monitor the network), pre-process it, running detection algorithms and generating responses accordingly. This approach is inappropriate both for resource-constrained scenarios and for large networks. The problem becomes even harder in the worst-case scenario, where an adversary forces the detection algorithm to behave the worst in terms of performance, thus forcing the IDS to spend a large amount of resources. This may facilitate evasion, for example if the IDS discards packets. One way of achieving this is by constructing carefully crafted payloads that make the signature matching algorithm to repeatedly backtrack during inspection may render packet processing rates million of times slower than in the average case [7, 8].

IDNs attempt to solve this problem by distributing the tasks among different nodes. Depending on their role in the network, some nodes gather local data and send it to another node, probably with more resources, who correlates the data and performs actual detection. This separation of duties makes IDNs a suitable solution for distributed systems, including mobile ad hoc networks (MANETs), where there are no central nodes and every host must collaborate to ensure a proper network behavior [9]. IDNs are also used in networks geographically separated to allow different entities to collaborate and mitigate large scale attacks [10]. Current attacks are capable of infecting simultaneously various networks or incorporating evasion techniques to pass undetected [11]. Moreover, many zero-day attacks target simultaneously a huge number of systems worldwide [12], leaving little time to patch other networks. Thus, to prevent threats from propagating through different domains, collaboration between different IDNs is essential.

1.1. Motivation and Contributions

Due to the connectivity of the nodes in an IDN, threats affecting one node may propagate and affect the entire network. To completely mitigate such threats, it is required to protect every single node which is at risk. In many scenarios this is not possible due to resource constraints (in terms of time, human and financial costs, etc.) and the problem turns into optimally investing in security measures to minimize the residual risk. Similarly, given an acceptable level of the residual risk, the budget spent can be minimized.

In this sense, the placement of countermeasures in IDNs plays an important role. Deciding what has to be protected—i.e., where to allocate countermeasures—is a complex task. This decision depends on many factors, such as the adversarial model, the impact of the attacks, or the cost of implementing the countermeasures. These factors vary considerably even within the same network. For example, the cost in terms of time and energy of implementing cryptography mechanisms for wireless communications is different depending on the operating system and the software used [13]. Similarly, the impact of a denial of service attack on a Security Operations Center (SOC) is typically higher than that of an anti-virus solution for a personal computer.

Due to the wide variety and complexity of IDNs proposed in the literature, the risk-rating of these networks is typically done in an ad-hoc manner, what makes it expensive and error prone. In this paper, we develop a generic risk-rating framework for IDNs called DEFIDNET. This framework uses a generic system model tailored to IDNs that allows to define generic cyberdefense components that compose the network as well as the adversarial model. Then, it incorporates procedures to asses the risk of the IDN. Concretely, considering that some nodes may be targeted by an adversary, the framework evaluates the propagation of intrusive actions through the network considering the influences between nodes. It then estimates the impacts of such actions regarding different attack strategies. Overall, this provides a global picture of the IDN risk considering the impacts and likelihood of attacks. Finally, the framework provides a search-based optimization module that provides the set of optimal countermeasures in terms of cost and mitigated risk. To illustrate the benefits of DEFIDNET, we provide experimental results using different network architectures and a case study using a complex cooperative network.

In summary, the main contributions of this work are:

1. A system model that defines the elements of an IDN. It is composed of a model for IDN nodes in terms of their logical components, functions, and channels. Then, depending on which of these components are implemented in a node, several roles are presented. Finally, the architecture of the IDN is defined regarding the connections and influences between nodes.

2. A threat model that indicates sequences of intrusive actions that an adversary can execute against an IDN. The model considers not only actions against a single node, but also the propagation of such action throughout connected nodes and provides a global picture of the risk to which the IDN is exposed.
3. An allocation model based on a multi-objective optimization algorithm to obtain optimal allocation of countermeasures with respect to both risk and cost.
4. Experimental results obtained through simulation for different IDN architectures and a detailed case study illustrating the benefits of DEFIDNET.

Our experiments have been conducted with a prototype implementation of a tool based on the proposed framework that is freely available for download[2].

1.2. Organization

The paper is organized as follows. In Section 2 we provide some background and related work. In Section 3 we describe the system model to define IDNs, and in Section 4 we describe the framework in detail, including the threat, risk-rating and allocation modules. We provide experimental results in Section 5 and in Section 6 we present a case study with DEFIDNET. Finally, Section 7 presents the conclusions.

2. Background and Related Work

In this section, we first review existing architectures proposed for IDNs. We next present adversarial models for IDNs and a recent taxonomy of adversarial attacks against Intrusion Detection Systems (IDS) that we adopt in our work. Finally, we review the concept of resource allocation and how it is adapted to our setting.

2.1. Intrusion Detection Networks

IDNs are used in many scenarios, from collaborative domains where different entities share information to detect global attacks [5], to wireless local networks such as Mobile Ad-hoc Network (MANET) [14]. In both cases, the IDN is composed of multiple nodes distributed over the network where each node communicates with one or many other nodes [15]. Depending on how nodes are connected and which are their responsibilities or roles within the network, the architecture of an IDN can be either centralized, hierarchical, or distributed (see Table 1). However, real world IDNs typically combine various of these architectures, resulting in complex architectures which are difficult to classify.

There are many works in the research literature facing the problems of trust in IDNs [16], correlation methods, detection algorithms, etc. According to a recent survey by Patel et al. [15], most of the collaborative approaches are proposed for wireless networks such as MANETs, and few works address the problem of collaboration among entities from different administrative domains. However, this is nowadays a hot research topic which is gaining interest by the research community [17, 15, 18].

2.2. Adversarial model in IDNs

IDNs are complex defense mechanisms that counteract distributed, sophisticated attacks against distributed organizations or entities. Accordingly, they are as well an attractive target for attackers. Besides the requirement of efficacy and efficiency, it is important to design resilient and robust systems to maintain an acceptable level of security even in the presence of adversaries. However, few works have dealt with the problem of adversarial capabilities in IDNs.

Bye et al. [10] presented a general description of IDNs in base of five building blocks: Communication Scheme (how nodes communicates between them), Group Formation (how to aggregate nodes in the network), Organizational Structure (hierarchical vs fully-distributed architecture), Information Sharing (format of

[2]See `http://www.seg.inf.uc3m.es/~spastran/defidnet/DEFIDNET_GUI.zip`

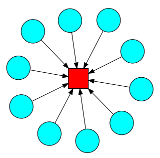
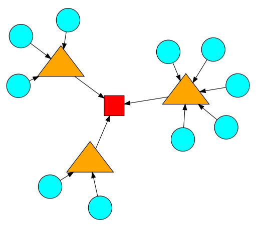
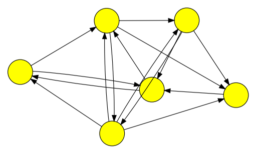
Architecture	Characteristics	Graphical view
Centralized	• Several nodes are connected to a central node. • External nodes collect data, and the central node aggregates and correlates it. • The collectors send information to the central node. • The central node makes the final decision and emit responses, if needed.	
Hierarchical	• The nodes in the IDN are clustered in different levels, being each cluster a centralized architecture. • Nodes in the lowest levels collect data. • Intermediate levels aggregate data from lower levels. • The highest level contains a single node that correlates data from lower levels and makes the final decision.	
Distributed	• Each node in the IDN is connected to one or various nodes. • There is not a single node making decisions, i.e., the detection is distributed. • There are different approaches to share information: P2P, publish-subscribe, etc.	

Table 1: Description of the architectures of IDNs and proposed works.

messages, considering privacy, anonymity and interoperability factors) and System Security (security of the IDNs itself, in terms of Trust Management, Access Control and Availability). Only the last module considers robustness. Authors define an adversarial model where adversaries are able to gather information to make query/response analysis and gain information about the functions and locations of the different nodes involved in the network. Then, authors propose a design for resilient IDNs that deals with attacks against the location privacy. However, authors do not consider other possible attacks and their model is not focused on complete adversarial capabilities.

Fung [19] defined a set of "insider attacks", i.e. attacks where the adversary has somehow gained access to the cooperative network. She introduces the *sybil* attack, where the adversary uses fake identities to make false reports and gain influence in the network; the *newcomer* attack, where the adversary changes its identification information once it is detected; the *betrayal* attack, where a benign node turns into malicious and sends false information attacking other nodes; and finally the *collusion* attack, where different malicious nodes collaborate to attack the network. Then, Fung reviews the main proposals for IDNs, analyzing the adversarial possibilities in terms of the proposed attacks for each proposal.

Xenakis et al. [14] provide a survey of IDN for MANETs, showing weaknesses of these architectures. Each of the weaknesses proposed can be viewed as adversarial capabilities to attack the ID network. However, the main purpose of the authors in that work was to point out flaws in the design of architectures, rather than considering the actual adversarial model.

A recent survey by Corona et al. [20] identifies six general categories of attack against classical IDS. The attacks are classified regarding the goal of the adversary, which results in different consequences:

1. **Evasion**, where an attack is carefully modified to avoid the IDS detecting it. This is the most common attack studied in the literature. For example, blending and mimicry techniques are examples of evasion.
2. **Overstimulation**, where the IDS is fed with a large number of attack patterns to overwhelm analysts and security operators. For example, *Mucus* is an IDS stimulation tool by Mutz et al. [21] that

generates packets that purposely match the signatures of Snort [6] to generate a large number of detection alerts.

3. **Poisoning**, where misleading patterns are injected in the data used to train or construct the detection function. This attack is applicable to IDS that use retraining, i.e. it modifies the detection function in detection time [22]. An example of such attacks are the Allergy Attacks [23, 24], which targets automatic signature generators such as Polygraph [25]. These attacks insert noisy data into the generation process to generate signatures in the IDS that filter out normal requests.
4. **Denial-of-Service (DoS)**, where the detection function is disabled or severely damaged. Algorithmic complexity attacks [7] are examples of such attacks. These attacks force the IDS to perform the worst case scenario, for example by generating packets that make the signature matching to generate the highest number of matches [8].
5. **Response Hijacking**, where carefully constructed patterns produce incorrect alerts so as to induce a desired response. This attack directly targets the response module of a system. For example, in a MANET, several colluding malicious nodes may send false reports indicating bad behavior of a benign node [17]. An IDN node then may block or ban such benign node from the network.
6. **Reverse Engineering**, where an adversary gathers information about the internals of the IDS by stimulating it with chosen input patterns and observing the response. The common approach is to perform query-response analysis, for example to discover signatures used by IDS [26].

Corona et al. classify the adversarial attacks in terms of the goal pursued by the adversary, and present specific proposals in the literature for each of these objectives. However, they do not provide an analysis of the actual adversarial possibilities in specific scenarios. Even though applying some of these attacks alone may not affect severely the IDS, a combination of various of them can be harmful. For example, an adversary who continuously overstimulates a sensor may cause a subsequent denial of service, and thus finally succeeding with an evasion attack. Moreover, this is to the best of our knowledge the only existing taxonomy of hostile actions presented against IDSs. Consequently, we will use it to categorize attacks against IDNs.

In Section 4.2, we present an adversarial model for IDN sustained on four basic intrusive actions to network communications: Interception, Blocking, Modification and Fabrication. Given the capabilities of an adversary to perform such actions, a threat module calculates the likelihood of the attacks to happen in each node of the IDN from the taxonomy of Corona et al.

2.3. Resource allocation

Resource allocation is the process of distributing a set of items among a set of entities with optimal efficiency, regarding some objective. In the case of IDNs, the items to be distributed are the set of countermeasures in the network and the objective is two-fold: minimize the cost and maximize the mitigated risk. There are two specific problems depending on the particular objective and constraints:

1. Given a tolerable risk, the problem is "Selecting the cheapest set of countermeasures that mitigates the risk below a tolerable level of risk"
2. Given an available budget of resources, the problem is "Selecting the more effective countermeasures (i.e. those that reduces the risk at most) that spend fewer resources than the given budget".

Recently Sang-Tran et al. [27] proposed an approach to select cost-effective countermeasures using risk models. This work assumes that "risks have been identified, estimated and evaluated and that the overall risk analysis process is ready to proceed with the risk treatment phase". The analysis conducted evaluates all the possible alternatives to allocate countermeasures and evaluates each of them. This approach is not easily tailored to IDN because two main reasons. First, risk assessment in IDNs is not straightforward, due to the propagation of the attack through the nodes. Second, evaluating all the possible alternatives for allocating countermeasures may be quite expensive and increases exponentially with the size of the network. Moreover, it highly depends on the adversarial model, which is not included in the analysis by Sang-Tran et al. In our work we provide a method for risk-rating in IDNs which uses an abstraction and thus is independent of specific IDNs. Moreover, we use an evolutionary-based search [28] of optimal alternatives, which improves the efficiency because it does not require to explore all the possible alternatives.

3. System model

Intrusion Detection Networks (IDNs) include a huge variety of approaches but we can consider two main types. On the one hand, different entities located in different places and sharing information through the Internet form a Collaborative Intrusion Detection Network (CIDN) [29, 30, 5]. On the other hand, different wireless sensors monitoring a Mobile Ad-Hoc Network (MANET) [31, 32] also constitute an IDN. The main difference between these two approaches is that while CIDNs are likely composed of well-equipped systems (i.e., having substantial memory and processing resources), MANETs are composed of resource-constrained nodes requiring lightweight detection algorithms. Other differences have to do with communication capabilities: while CIDNs communicate through the Internet and use wired channels, MANETs use a wireless medium. Thus, contrarily to a CIDN, nodes in a MANET are geographically closer to each other. Despite their differences in processing and communication capabilities, both approaches detect distributed intrusions by interconnecting detection nodes, and therefore share common functional modules and information channels.

In this section, we define a conceptual model for the nodes of an IDN. The main purpose is to establish common building blocks and thus define common threats, in order to define a generic adversarial model (which is provided in next section). Accordingly, we first present a general, functional overview of these nodes in terms of their logical components, functions, and channels. Then, we define the roles of these nodes in the IDN architecture, according to their responsibilities. Finally, we present the architectures proposed in the literature for IDNs in terms of the system model and the roles presented.

Every node participating in an IDN contains some basic components, depicted in Figure 1. We identify four channels of information and four functional modules. Depending on the role of the node in the IDN, these components may be activated or not, as we discuss later in section 3.3. Finally, nodes are connected to other nodes to share information. Depending on how the nodes are connected and their roles, the IDN may have different architectures.

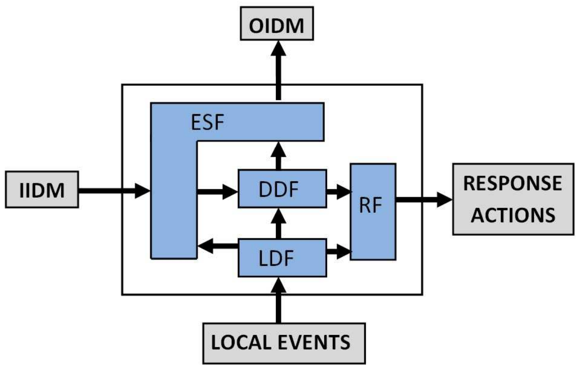


Figure 1: Functional view of an IDN node.

3.1. Channels

We define a communication channel as any input or output interface used by a node to receive or send information. We consider these channels logically, and do not focus on the physical implementation of the channel. Nodes manage three different types of messages:

1. **Intrusion Detection Messages (IDMsg)**. It comprises any exchanged message containing information related to the detection of attacks. The format of these messages vary for each IDN. For example, the nodes may implement the IDMEF format [16], which is an RFC standard [33] that defines a format for IDS alerts.
2. **Local Events (LE)**. This is the data monitored by sensors locally. It includes both host data (system logs, audit trails, etc.) and network traffic (TCP headers, HTTP payloads, connections, etc.).

3. **Response Actions (RA)**. A response action is triggered whenever a intrusion is detected. This includes activities such as logging alerts in a file, blocking IP addresses or turning devices off, to name a few.

We define four possible channels for nodes, two input channels and two output channels: the Input Intrusion Detection Messages (**IIDM**) to receive IDMsgs from other IDN nodes; the Local Events (**LE**) channel, to gather data locally; the Output Intrusion Detection Messages (**OIDM**), to send IDMsgs to other nodes in the IDN; and the Response Actions (**RA**) channel, from which active responses are triggered.

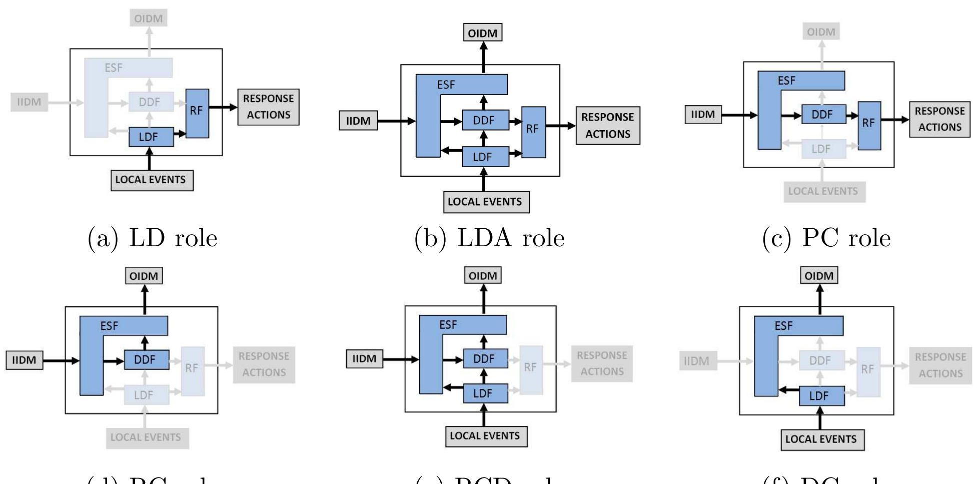
3.2. Functions

Distributed IDN nodes communicate with other nodes to share IDMsgs which may be relevant to detect or block distributed attacks. Every node may run up to four functions in the detection process. The inputs and outputs of these functions are provided by the channels defined above:

1. **Event Sharing Function (ESF)**. This function manages the communication between nodes in the network. Its responsibility in the node is twofold. First, it processes and formats incoming IDMsgs received through the IIDM channel, and provides these data to the Distributed Detection Function (DDF). Second, it processes and formats the output of the DDF and the Local Detection Function (LDF) to send these messages to other nodes through the OIDM channel.
2. **Distributed Detection Function (DDF)**. It aggregates and correlates data from both the LDF and the ESF. Then, the correlated data is sent to the Response Function (RF) if some response is needed, or to the ESF in the case that this data is being shared with other nodes.
3. **Response Function (RF)**. The RF is activated whenever a response action is needed. Among the activities involved in this function are updating blacklists with suspicious IPs, sending remote commands to shut down compromised systems, logging alerts, etc.
4. **Local Detection Function (LDF)**. The LDF uses local data, which may include both host and local network data, to perform local detection. As in the case of the DDF, it may use any of the classical detection techniques employed by IDS, like signature matching, anomaly detection, specification based detection, etc. The output of this function is provided as an input to the other three functions: to the RF if some response action is needed; to the DDF to correlate and aggregate local detection with other IDMsgs; and to the ESF to share the information with other nodes in the network.

3.3. Node Roles

We define six roles for nodes, which are explained below. Based on these roles, each node activates some or all of the channels and functions, as shown in Figure 2:



(a) LD role (b) LDA role (c) PC role

(d) RC role (e) RCD role (f) DC role

Figure 2: Logical schemes of roles for IDN nodes

1. **Local Detection (LD)**. The node detects intrusions based on local events only. It gathers local data using the LDF, which includes both network traffic from its LAN and host-based data. Because alerts are neither shared nor received from other nodes, the RF must be enabled to generate the corresponding response action whenever a local intrusion is observed. Figure 2-a) shows the schematic view of this role. Classical IDS working independently, like Snort [6] or Bro [34], have this role. In IDNs, there are no nodes with this role, because they can neither send nor receive data from other nodes. However, we include them in our study for the sake of completeness.
2. **Local Detection and Alert Sharing (LDA)**. This is the most complex role as it uses all the channels and runs all the functions, as shown in Figure 2-b). Nodes perform detection using as input both the local events and the IDMsgs received from other nodes. First, the LDF processes the local data, which are then aggregated and correlated in the DDF with external IDMsgs, after being processed by the ESF. Whenever an intrusion is detected, the RF generates the corresponding response action. Moreover, nodes with this role must share IDMsgs (using the OIDM channel and the ESF) with other nodes in the IDN.
3. **Pure Correlation (PC)**. The node correlates IDMsgs received from other nodes using its IIDM channel and the ESF. It then makes a decision using the DDF and, if needed, generates a response through the RF. Nodes with this role do not use local data and thus the LE channel is inactive. Well-known SIEM systems [35] have this role. Figure 2-c) shows the schematic view of this role.
4. **Remote Correlation (RC)**. The nodes receives multiple IDMsgs from other nodes through the IIDM channel and preprocesses them in the ESF. Then, it aggregates and correlates them using the DDF, and share the aggregated data and results from the correlation to other nodes through the OIDM channel. It does not generate any response. Figure 2-d) shows the schematic view of this role.
5. **Remote Correlation and Detection (RCD)**. The node performs local detection based on data gathered locally through the LE channel and the IDMsgs received from other nodes through the IIDM channel. Then, it shares the aggregated IDMsgs with other nodes using the OIDM channel, but it does not generates responses (the role is similar to the LDA, but without emitting responses). Figure 2-e) shows the schematic view of this role.
6. **Data collection (DC)**. The node is in charge of gathering local events through the LE channel. Then, it processes it with basic filters or routines in the LDF and provides other nodes with the corresponding IDMsgs, using the ESF and the OIDM channel. Figure 2-f) shows the schematic view of this role. For example, in a wireless sensor network, different sensors gather data and send it to a central sink for further processing. These sensors may have the DC role, while the sink node would have the PC role.

3.4. Node Connections

The system model considers nodes regarding their corresponding functions and channels. Nodes use the ESF, either for sending data (through the OIDM) to other nodes or to receive data (through the IIDM) from other nodes. The nodes are thus connected between them. This connection is unidirectional, i.e. if node A_i sends data to node A_j, then the OIDM channel of A_i is directly connected to the IIDM channel of A_j and the information flows in one direction, from A_i to A_j. In general, we refer to the set of n nodes connected to one node A_i as $C_i = \{A_1, \dots, A_j, \dots, A_n\}$, with $j \neq i$.

When considering the information received externally, nodes may either believe this information or they may question its correctness. Accordingly, many approaches in the literature of IDNs propose the concept of trust in the information received by one node [32, 36].

Nodes may assign different levels of importance to the information received from external nodes through the IIDM channel. We normalize such importance to values between 0 and 1, and consequently, each connection is weighted by an influence factor. For each node, the sum of the influences received from the connected nodes must add up to 1. Accordingly, for each node A_i of the IDN, let I_{ij} be the influence from node A_j to node A_i, with $j \neq i$. Thus:

$$\sum_{\forall A_j \in C_i} I_{ij} = 1 \tag{1}$$

If no external nodes are trustworthy, then the node may just deactivate the IIDM channel and thus it may have the DC or LD role. In the case where both the IIDM and LE channels are used, the information received from other nodes is used internally in the DDF together with data gathered locally using aggregation and correlation procedures.

3.5. IDN Architectures

In this section we review the architectures defined for Intrusion Detection Networks in the literature and analyze them in terms of the proposed system model. A rather complete survey of such architectures can be found in [5].

3.5.1. Centralized

In a centralized architecture, several nodes gather local information and perform local detection. Detection results are sent to a central node. This central node receives multiples IDMsgs, aggregates and correlates them, and performs distributed detection. Figure 3 shows the scheme of a centralized architecture with six detection nodes and a central correlation node. Each detection node has the DC role, as they just gather local data, process these data (this may include any routine to obtain a local view from different local sources of data) and send the data with the IDMsg format (using the OIDM channel) to the central node, which has the PC role. This central node, upon the reception of new IDMsgs (using the IIDM channel) performs correlation and aggregation, and activates a response if needed.

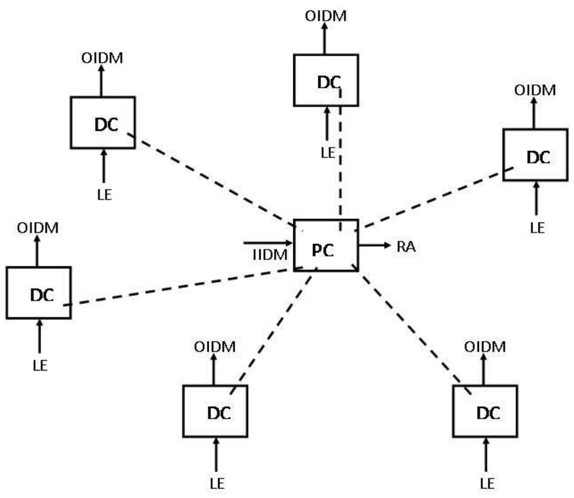


Figure 3: Centralized architecture of IDN.

3.5.2. Hierarchical

In an hierarchical architecture, the IDN is divided into levels. Nodes in each layer gather data received from lower levels or gathered locally, and send aggregated data to the upper level for correlation. In the top level, a global node performs the final correlation and emits responses if needed. Figure 4 shows a hierarchical architecture with three levels. The lowest level is comprised of nodes with the DC role that locally gather and process local events. They send these data in the IDMsgs format to some node in the upper level of the hierarchy, who aggregates and correlates the received IDMsgs. This node, may have the RC role, if it only uses external IDMsgs in the correlation, or the role RCD, if it also includes local data. The correlated data is sent to the upper level, if any, or it is used to emit responses if there are no more levels. We consider that at the top level of a hierarchical architecture there is a single node, which is only responsible of correlating the data received from other nodes (i.e., with role PC) and generating responses if needed. This could be the case of, for example, a Security Information and Event Management (SIEM) system or a Security Operation Center (SOC). Thus, the highest level in the hierarchy contains a single node with the PC role. The response module of PC nodes should command response actions to the lower level nodes through IDMsgs. Thus, the detection events are propagated in a bottom-up design, while the the responses are emitted in a top-down design.

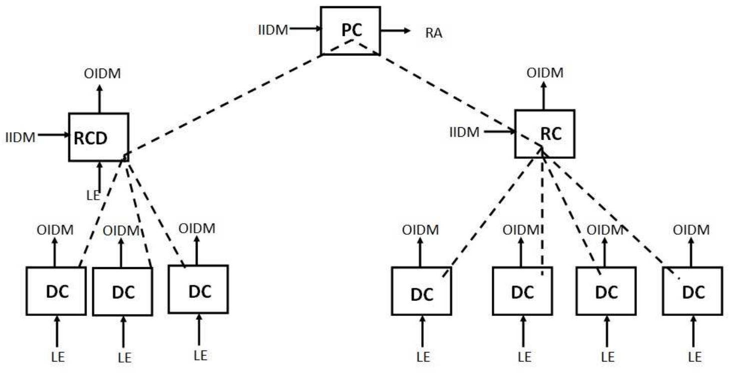


Figure 4: Hierarchical architecture of IDN.

3.5.3. Distributed

Figure 5 shows a distributed architecture with five nodes. All nodes have an LDA role. The detection process in the nodes combines local and distributed detection, as it uses both data gathered locally and IDMsgs received from other nodes. Thus, unlike other architectures, in the distributed architecture there are no critical, central nodes with higher responsibilities and all nodes are equally responsible of the network defense. However, because of the large number of connections, attacks to a single node are rapidly propagated to the entire IDN.

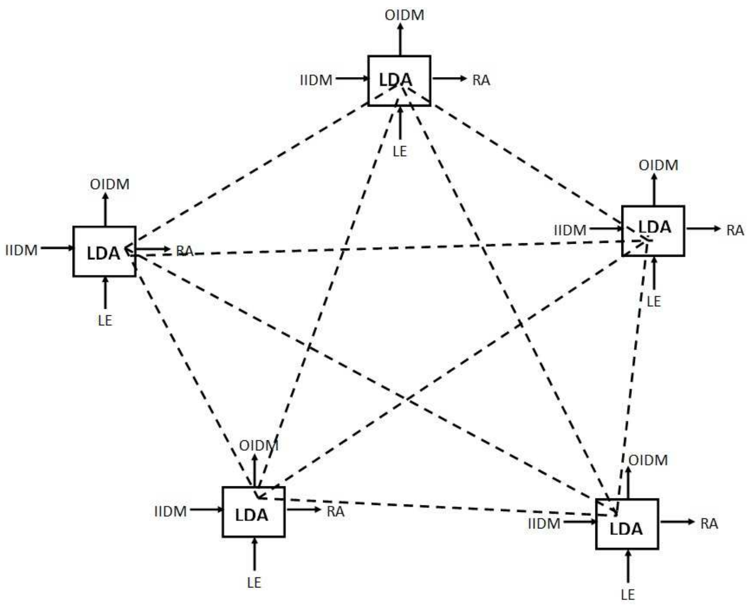


Figure 5: Fully-distributed architecture of IDN.

4. Framework Description

In this section, we describe the different modules of DEFIDNET, whose aim is to establish the risk to which the IDN is exposed to and to provide optimal allocation of countermeasures. These modules are applied on the IDN through the abstraction provided by the system model presented in Section 3. The modules, namely, the threat module, the risk-rating module, and the allocation module, are depicted in Figure 6. Below we describe each in detail.

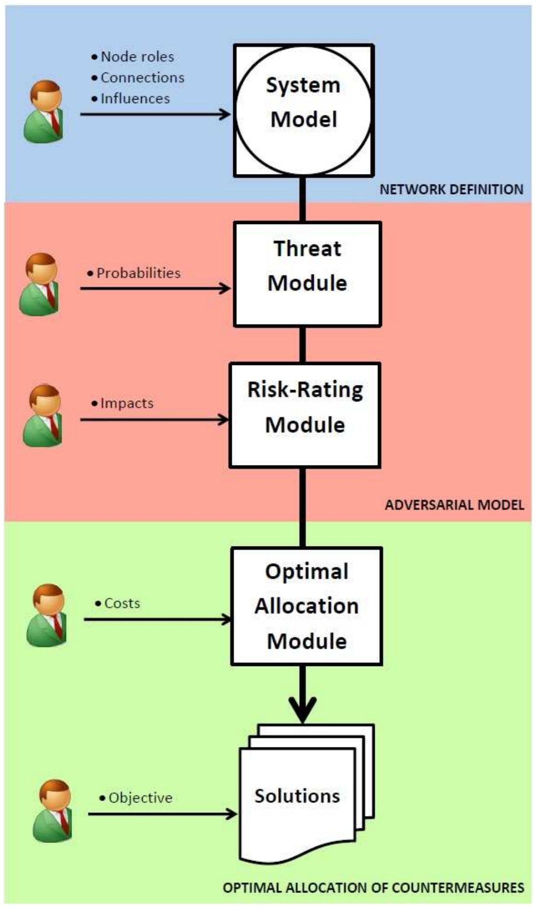


Figure 6: Modules of the framework DEFIDNET. It uses as input an IDN abstraction and applies the different modules to the network. The output is a set of optimal solutions (the choice of a specific allocation depends on the particular objective)

.

4.1. Threat Module

This module is used to define the threats to which IDN nodes are exposed and the propagation of the intrusive actions throughout the network. It receives an specification of the IDN based on the system model introduced in Section 3. For every IDN node, this specification defines four channels: input of local events (LE), input of IDMsg (IIDM), output of IDMsg (OIDM), and output of response actions (RA).

Nodes in an IDN send and receive data using these communication channels. The communication consists of the exchange of packets of information using some network protocol and the specific format of the IDMsg. Accordingly, we consider that adversaries in IDNs may perform any of the following intrusive actions:

1. **Blocking**. This attack targets the integrity and availability of the data. The adversary interrupts the communication or makes it unavailable. Packet Dropping attacks [32] in MANETs are an example of blocking attacks, where a malicious node drops packets that are supposed to be forwarded.

2. **Modification**. This attack targets the integrity of the data. The adversary intercepts data, modify its content and forwards it to the actual destination. For example, the adversary may modify the content of an attack to evade the signature matching process from IDSs [37].
3. **Interception**. This is a passive attack that compromises the confidentiality of the information shared within the IDN, which may allow an adversary to, for example, locate nodes with the highest responsibilities in the IDN in order to perform a more severe attack later. The adversary eavesdrops the contents of the messages transmitted in the channels. For example, a malicious node in a MANET which promiscuously monitors its neighbors, performs interception. This attack is hard to detect, but can be counteracted by cryptographic techniques.
4. **Fabrication**. Fabrication attacks compromise the authenticity of the data. The adversary generates fake data and sends it to the victim. For example, using spoofed addresses, the attacker may fabricate packets that match the signatures of an IDS in order to overstimulate it [21].

We consider an adversary who targets a certain number of nodes at a time, performing one or more intrusive actions. The threat module is applied in two steps. First, it receives the probabilities of each intrusive action being performed for every channel and every IDN node. These probabilities are manually entered. Then, the threat-module automatically propagates these initial probabilities throughout the network. Below we further explain these two steps of the module.

4.1.1. Attack Probabilities in Node Channels

The threat module receives as input the system model and the probabilities of intrusive actions for each channel of every node. If the nodes uses the same physical channel for input and/or output data, then the probabilities would be the same. For example, if a computer uses the same network interface to monitor local traffic and to receive messages from other computers, then blocking the data in that network interface may block the LE and IIDM channels of the node. Additionally, in many adversarial scenarios, the probabilities of the intrusive actions are similar. For example, if the adversary is able to perform modification, it is likely that she may perform fabrication too. Security managers generally have this information as a result of standard risk analysis conducted over the network. Indeed, if some countermeasures have already been placed in a node, then the probabilities in this node would be lower than in a non-secured node. For example, the manager can estimate that the information flowing outside the limits of its organization may be more vulnerable to blocking attacks. In any case, if the manager is not sure about these probabilities, it can establish the highest value (i.e., one) to analyze the worst-case scenario. In any case, we do not consider in our framework how these probabilities are established, because it may strongly depend on the adversarial model and the network specifications. They are just defined manually, as shown in Figure 6.

This step of the threat module provides information about which nodes have been targeted initially. Thus, the threat module indicates where countermeasures should be allocated to protect nodes against the intrusive actions. However, it does not specify where it is better to place these countermeasures if the resources are limited, because it depends on the damage of each action and the cost of implementing the associated countermeasures. Moreover, as explained in the next step, the probabilities of attacks are propagated throughout the nodes, and thus the damage may vary depending on the architecture of the IDN. These issues are dealt with by the the allocation and the risk-rating modules, respectively, which are described later in this section.

4.1.2. Propagation of Probabilities Throughout the IDN

Nodes in an IDN share information using IDMsg. Accordingly, the intrusive actions that affect one node are propagated throughout the connected nodes. The degree of propagation depends on the influences of the nodes connected. Consequently, a single intrusive action in a channel of a node may not increase considerably the risk of the network. However, due to propagation, an adversary who wisely select its victims or combines several actions can provoke a serious damage in the network.

For example, suppose that a node A with the role DC is connected to a node B with the role PC. A provides data to B, who correlates this data and emits responses, if needed. Suppose that the adversary modifies the data in the LE channel of A. This intrusive action affects the OIDM channel of A as well, because the LE and the OIDM channels are internally connected through the event sharing function (ESF). Moreover, the OIDM channel of A is connected to the IIDM channel of B, and thus the modification would affect B as well. The degree of damage caused in B depends on the influence of A in B. The lower the influence is, the lower the damage propagated from A to B is.

The initial probabilities are manually entered for each node, and define the nodes that are vulnerable to external attacks. Then, the probabilities are propagated throughout the network as follows. First, the nodes propagate the probabilities internally. Depending on the role and the functions enabled in each node, the IIDM and LE channels may be connected to the RA and/or OIDM. Thus, the probabilities of every action affecting the input channels are propagated to the output channels in each node. Second, the probabilities of the OIDM channels are propagated to the IIDM channels of the connected nodes, weighted by the influence of the connection. Equation 2 indicates the propagation between connected nodes. In this expression, C_i represents the set of nodes connected to A_i, I_{ij} is the influence from the node A_j to node A_i, with $j \neq i$, and we use the terms t and $t+1$ to distinguish between the previous and the updated probabilities, respectively. If one node previously had a probability in the IIDM channel ($P_{IIDM}(A)^{(t)}$) greater than zero, then the new probability ($P_{IIDM}(A)^{(t)}$) is the maximum between the previous probability and the sum of the probabilities propagated from connected nodes. With this propagation scheme, we are considering the worst case situation from the defensive perspective, as the maximum probability of being attacked is considered in the risk rating process:

$$P_{IIDM}(A_i)^{t+1} = \max\{P_{IIDM}(A_i)^t, \sum_{\forall A_j \in C_i} P_{OIDM}(A_j)^t * I_{ij}\} \tag{2}$$

Figure 7 shows an example of the propagation effect in a network with three nodes. The boxes in each node correspond to the probabilities of the four intrusive actions (B, M, I, and F) in each channel of the node. The nodes with role DC (*Slave1* and *Slave2*) are connected to a node with role PC (*Global*). In the initial state, only *Slave1* and *Slave2* are targeted, but after the propagation effect, the probabilities of the node *Global* are updated following Equation 2, and thus it becomes at risk.

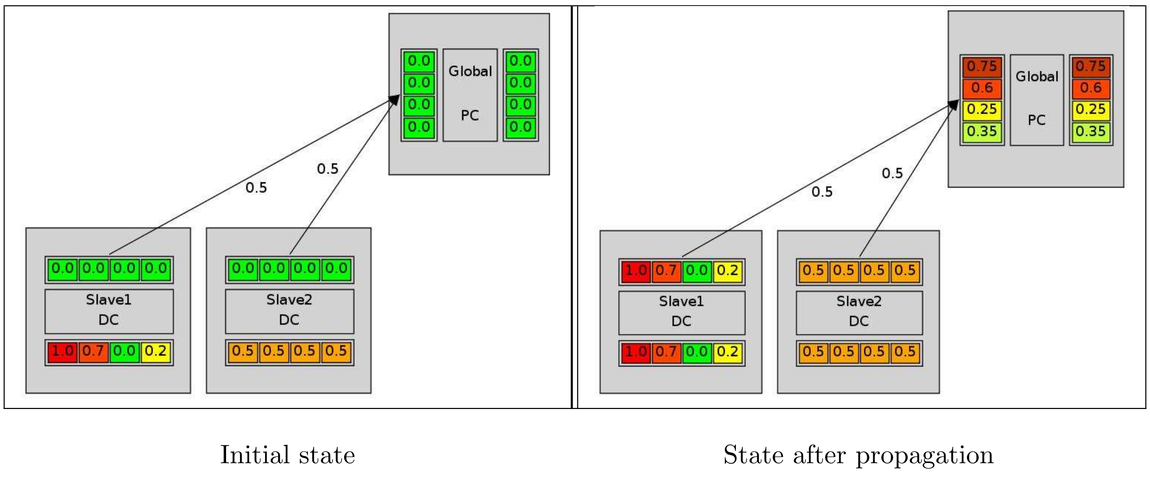


Figure 7: Effect of propagation of the probabilities of attack in an IDN with three nodes. The channels are represented as in Figure 1, i.e. the LE is on the bottom, the IIDM is on the left side, the OIDM is on top, and the RA is on the right side. Each box corresponds to the probabilities of the four intrusive actions (B, M, I, and F) in each channel of the node. The influence is represented as a weight associated to each with an arrow.

4.2. Risk-Rating Module

In Information and Communication Technology (ICT) systems, risk is traditionally calculated as the product of the damage caused by attacks to the system times the likelihood (i.e. probability) of these attacks happening [38]. As explained in the previous section, the threat module outputs the probability of each intrusive action in each channel of every node. The risk-rating module first receives the impact of the attacks, and then calculates the risk using the probabilities of these attacks happening. Next, we explain how the impacts and likelihood are defined, and how the risk is calculated from these metrics.

4.2.1. Attack Impacts

The adversarial model of IDNs is rather different to other ICT networks, where the main objective of attackers is to compromise the information confidentiality, integrity, or availability of data. In IDNs, though, attacks may have different goals. We adopt the taxonomy proposed by Corona et al. in [20], which classifies attacks to Intrusion Detection Systems. This taxonomy is further explained in section 2.

The impacts of attacks in each node depends on its consequences and it may be different depending on the adversarial model, the architectures, etc. For example, in a centralized IDN, a DoS attack against the central PC node may have higher impact than a DoS against the DC nodes. Thus, in DEFIDNET the impacts are defined for each node and each attack (i.e. evasion, overstimulation, DoS, poisoning, reverse engineering, and response hijacking), by manually entering a damage value.

4.2.2. Attack Likelihoods

When targeting IDNs, adversaries may use different attack strategies. To assess the risk, each possible attack strategy should be considered. For example, a DoS could be performed by blocking the IDMsgs sent to the node, or by flooding the node with local events (considering a DoS is any action forced by an adversary that prevents the victim node from performing detection [20]). Table 2 summarizes the intrusive actions required depending on the attack goal. Following, we explain how each attack goal can be achieved by an adversary by performing the different actions on the channels.

- **Evasion**. The adversary causes the node to misbehave and stealthily attacks the IDN. It only affects detection functions, i.e. the LDF and DDF, as the objective is to force the actual detection to malfunction. This can be done by means of one of these three techniques:
  1. *Blocking.* In some approaches, the detection process starts when some suspicious packet or message is received, like anomalous routing data in MANETs [39]. The attacker may perform the attack blocking these packets in the node to avoid detection.
  2. *Modification.* The attacker carefully modifies the data to hide the intrusion evidence the IDN node is looking for. This way, the adversary can avoid signature matching [40], or it can mimic the statistical properties of a normal model [11].
  3. *Fabrication.* The IDN node may be waiting for specific IDMsgs or packets to see whether a node is correctly forwarding packets or not [41]. In such a case, the adversary can generate this packet specifically for the IDN node.

- **Overstimulation**. A set of well-crafted packets are sent to the node to make it trigger a huge amount of responses. The objective of overstimulation is to stimulate the system so that it generates a huge amount of alerts, thus overwhelming the security operator auditing such alerts. Thus, it can be applied to every function of the nodes. Overstimulation is always performed using *fabrication*, namely, the adversary should generate some specific packet that provokes the node reaction. For example, by fabricating packets that match the signatures of the targeted node [21], the adversary can overwhelm security analysts or overload the IDS resources.

- **Poisoning**. The adversary looks for nodes that update their detection function in real time with new data. The goal is to inject some noise forcing the detection function to learn wrong patterns. Similarly to the evasion goal, this objective is only applicable to the LDF and DDF. Since the objective of the adversary is to inject specific information in the node, it needs *modification* (of data sent by other nodes in the IDN) or *fabrication* (of new data) attacks. For example, two colluding nodes can report good behavior from each other. This way, their reputation in other IDN nodes would be increased.

- **Denial of Service**. This attack targets the availability of the nodes. The adversary aims at deactivating the detection and response capabilities of the nodes, forcing them to drop packets or avoiding packets to reach the communication channels. The difference between this and overstimulation is that a DoS aims at stopping the node from working properly, while overstimulation attempts to force the generation of such a huge amount of alerts that the security operator will be overwhelmed. A DoS may affect the LDF, DDF and ESF, which are the functions receiving external data. To force those functions to stop working, the adversary may either flood them to overload its resources, using *fabrication*, or it can *block* traffic to prevent the node from receiving the required data to work.
- **Response Hijacking**. The adversary sends selected intrusive data to the node, forcing it to generate a specific response. The responses in the nodes are generated in the RF, so this is the only function affected by these attacks. To provoke a specific response in the node, the adversary has to use one of the following techniques:
  1. *Blocking.* As explained above with the evasion, the IDN node may be waiting for specific IDMsgs or packets to confirm that a peer is not malicious. If the adversary blocks this critical data sent by a third peer, the node may erroneously believe that this third peer is malicious.
  2. *Modification.* The adversary may modify reports or IDMsgs to indicate that a third node is malicious.
  3. *Fabrication.* As with the modification, the adversary can generate false reports about a third node to force the detector to trigger an erroneous response.
- **Reverse Engineering**. The adversary gains information about the behavior of the node (architecture, detection function, set of measurements, etc.), which can lead to a more severe attack afterwards, such as evasion or denial of service. It is applicable to every function in the nodes. This could be done using the same techniques employed for an overstimulation attack, but in addition the node must intercept traffic to monitor both the inputs and outputs to the node and make the analysis. A paradigmatic reverse engineering attack in IDNs occurs when the adversary performs traffic analysis of the network in order to locate IDN nodes and responsibilities (for example, to know which nodes are performing correlation in a hierarchical architecture). Additionally, using *fabrication* or *modification* in the LE channel, and *interception* in the OIDM or RA channels, the adversary can perform query response analysis to infer secret information used internally by nodes [42].

Accordingly, using the probabilities of the nodes established in the threat module and the attack strategies showed in Table 2, the likelihood of each attack is calculated as the maximum probability of actions that conduct to it. For example, consider the *Global* node in Figure 7 after the propagation. The probabilities of intrusive actions in the IIDM channel (left side of the node) are 0.75 for blocking, 0.6 for modification, 0.25 for interception, and 0.35 for fabrication. As shown in Table 2, evasion in the node *Global* can be done by either blocking, modifying or fabricating in the channel IIDM. The most probable action is blocking (0.75).

Table 2: Taxonomy of attacks showing which intrusive actions may use the adversary on each channel to achieve different goals.

Adversarial attack	intrusive actions on channels			
	LE	IIDM	OIDM	RA
Evasion	$B \vee M \vee F$	$B \vee M \vee F$	–	–
Overstimulation	F	F	$M \vee F$	–
Poisoning	$M \vee F$	$M \vee F$	–	–
Denial of Service	$B \vee F$	$B \vee F$	B	–
Response Hijacking	$B \vee M \vee F$	$B \vee M \vee F$	–	$B \vee M \vee F$
Reverse Engineering	$(M \vee F) \wedge I$	$(M \vee F) \wedge I$	I	I

4.2.3. Risk Calculation

Once the impact of attacks are entered, and the likelihood of these attacks happening is calculated, the risk-rating module calculates the risk of one attack as the product of the likelihood of this attack multiplied by its impact on the node. Assuming that the impact of evasion in the *Global* node from Figure 7 is 100 and the likelihood of evasion is 0.75, then the risk of the *Global* node being evaded would be $0.75 * 100 = 75$.

The risk-rating module outputs the total risk of the IDN, and for each node, the risks for each attack and its aggregated risk (sum of all the attack risks). The total risk of the IDN is the sum of the risks of all the individual nodes. This information together with the information about which nodes have been targeted (given by the threat module), is given to the allocation module described in the next section.

4.3. Allocation Module

DEFIDNET is a framework to optimally allocate countermeasures in an IDN. In this section, we consider the problem of reducing the estimated risk using the lowest possible quantity of resources. The allocation module first receives the cost of the countermeasures, and then calculates optimal allocation of these countermeasures to reduce the risk. We next explain the two steps involved in this module.

4.3.1. Cost of Countermeasures

According to RFC 4949 [43], a countermeasure is "an action, device, procedure, or technique that meets or opposes (i.e., counters) a threat, a vulnerability, or an attack by eliminating or preventing it". Implementing such actions, techniques or procedures, or even buying such devices, entails some cost, which is different depending on the protected host and the environment of application. Our model assumes that a countermeasure effectively counteracts the attack on the channel of the node where it is implemented. Thus, we define the cost of a countermeasure as the quantity of resources required to protect a single channel of one node against a specific intrusive action. We consider this cost as a single value, and we do not consider neither what exactly it is (money, time, energy, etc.) nor how it is measured. For example, to protect against interception, encryption mechanisms can be used. These mechanisms may require the use of secret keys or a PKI. Depending on the network and the scenario, this may be more or less costly. Moreover, the cost of protecting against interception is not the same in different nodes and channels. For example, encrypting the communication in a MANET is usually more costly than encrypting a wired link. DEFIDNET uses as input the cost to protect each intrusive action on each channel of the nodes.

In the following, we consider a solution as a set of countermeasures to be applied to the IDN. On the one hand, when a countermeasure is applied to one channel to counter an intrusive action, the probability of this action happening in this channel becomes zero. However, since not all the channels are protected, after applying the countermeasures of a solution, some residual risk is left behind in the IDN. On the other hand, each countermeasure has an individual cost, and thus, applying a set of countermeasures has a total cost calculated as the sum of each individual cost.

4.3.2. Optimization of the Cost-Risk Tradeoff

For each solution, the more risk is mitigated, the higher the cost is. Ideally, optimal solutions should minimize both the risk and the cost. However, these are mutually conflicting objectives, and there is not a single optimal solution. Thus, a tradeoff between risk and cost must be considered. Accordingly, we use Multi-Objective Optimization (MOO) to obtain the set of optimal solutions that conform the Pareto set. In MOO with two objectives, a solution from the Pareto set is called non-dominated if there is not any other solution that improves one of the objectives without degrading the other objective. The set of non-dominated solutions is called the Pareto front.

There are several algorithms to obtain the Pareto front. In our experiments, we use an evolutionary MOO algorithm known as SPEA2 [44] (an optimization of the Strength Pareto Evolutionary Algorithm). SPEA2 is one of the most popular MOO evolutionary algorithms and has been successfully applied in the intrusion detection domain [45, 28]. Indeed, it is one of the two MOO algorithms implemented in the ECJ framework [46], which we use in our experiments. The other algorithm implemented in ECJ is NSGA2 (Non-dominated Sorting Genetic Algorithm) [47]. While both of them are sound algorithms, SPEA2 obtains further optimization in the central points of the Pareto front than NSGA2, which is more convenient to obtain solutions in the boundaries of the Pareto front. In our particular domain, solutions that are very costly or that reduce an small amount of risk are generally not recommended. Accordingly, the main purpose was to optimize the points where it is unclear the tradeoff between cost and risk, which are the central points of the Pareto front. For these reasons, we chose SPEA2 ofr our experiments.

Evolutionary algorithms perform heuristic search to explore a solution space. The algorithm manages a population of "individuals", which in our case are the different solutions to allocate countermeasures expressed as a binary mask (1 means that the concrete countermeasure is applied, 0 that it is not applied). Figure 8 shows an schematic view of an individual. Each position of the mask indicates which actions (i.e, blocking, modification, interception, and/or fabrication), for each channel, and for every IDN node, are counteracted by the represented solution. The evolution applies various genetic operators to the individuals to obtain each new generation. We next explain such operators:

- **Crossover**. Given two individuals, (i.e. binary masks), the crossover breeds a new individual by mixing certain genes from the individuals (i.e. chunks from the masks). Concretely, in our experimentation, the crossover randomly divides the masks of both individuals in 5 chunks, which are swapped between them uniformly.
- **Mutation**. Given one individual, the mutation operator flips a 10% of the bits from its mask. These bits are randomly chosen.
- **Selection**. This operators selects individuals from one generation to compose the next generation. We use the tournament selection, which creates several "tournaments" randomly choosing a number k of individuals from the population. Then, the best individual in each "tournament" is selected.

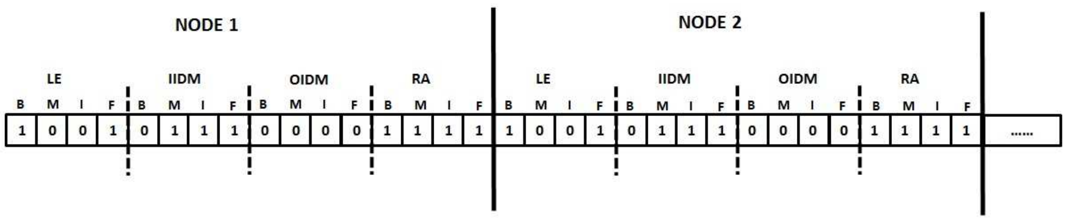


Figure 8: Representation of a SPEA2 solution to allocate countermeasures as a binary mask. Each position in the mask indicates whether a countermeasure to each attack action must be applied (1) or not (0) in the corresponding channel of every IDN node.

.

The population evolves during several generations, increasing the "fitness" of individuals. The fitness is a value associated with each individual, which indicates how good it is. Concretely, the individuals are evaluated as follows. First, the countermeasures indicated by the mask are applied (i.e. the corresponding probability is set to zero) and its corresponding cost is calculated. Then, the remaining probabilities of attacks in the network (i.e. those whose action have not been counteracted) are propagated, and the risk is re-calculated. Finally, the pair "cost-risk" is given to the algorithm to calculate the fitness. In the case of SPEA2, the fitness of each solution considers the number of points that are dominated by the Pareto points in the solution.

When it is required to reduce the risk completely or when there are unlimited resources, then all the nodes are protected completely (i.e, all the risk is mitigated). However, when the cost is limited or the IDN tolerates some risk, the Pareto front indicates which are the optimal solutions. These solutions indicate which are the countermeasures to be applied in order to solve one of the two following problems:

1. Given a tolerable risk, the problem is "selecting the cheapest set of countermeasures that mitigates the risk below a tolerable level of risk".
2. Given an available budget, the problem is "selecting the set of countermeasures that reduce the risk the most while spending fewer resources than the given budget".

If the budget is limited, the allocation solution must reduce the risk the most. If there is a tolerable risk, the allocation solution must be the cheapest that decreases the risk below the tolerated level. In some situations, though, neither the cost nor the risk are limited. In these cases, it is helpful to know whether it is worth to spend more resources to reduce the risk or not. When defending an IDN, one may think that the more resources are spent, the more risk is mitigated. However, this is not always the case. In the following section, we show experimental results that confirm this intuition.

5. Experimental Work and Discussion

As it has been discussed in the previous section, one of the advantages of DEFIDNET is that it allows to model IDNs through the definition of architectural parameters (size of the network, role of the nodes, connections, and influences) and per node parameters (probabilities of attack, impact of attack, and cost of countermeasures).

In this section, we provide analytic results using DEFIDNET. Concretely, we model different IDNs built over two architectures, i.e., hierarchical and centralized. The studied IDNs share the same adversarial model, where the data collector nodes are targeted by an adversary. To analyze the allocation of countermeasures under different circumstances, we build IDNs using different values for the cost of implementing countermeasures in the IDNs and different values for the influence of the connections between nodes. In this section, we first explain how the networks are modeled and the parameters established. Second, we provide an analysis of the tradeoff between cost and risk on each IDN, which helps to decide the placement of countermeasures and whether it is worth or not to spend resources, depending on the mitigated risk. Finally, we discuss the use of DEFIDNET in the modeling of IDNs and how it provides optimal alternatives to allocate the countermeasures depending on the resources available or the tolerable level of risk.

5.1. Network Modeling

We have conducted experiments using two network topologies: centralized and hierarchical. Each IDN has 101 nodes. We next describe the use of DEFIDNET applied to each of these networks.

5.2. IDN Definition

For each IDN, we first define the system model, which is composed of the number of nodes, the role of each node, and the connections between them. Depending on the role of each node, some of their functions and channels are activated, as explained in Section 3.

In a centralized network, a central node correlates the information received from several nodes and generates responses. We use a network with 100 nodes that collect data and send it to a central correlation node. In a hierarchical IDN, the network is divided into levels. We have established 3 levels in the network. In the top level there is one global node (with role PC). In the middle level, there are 10 nodes (with role RC) connected to the global node. Finally, in the bottom-level, 10 nodes are connected to each of RC nodes in the middle-level. The nodes in the bottom level have the role DC. Thus, the hierarchical network has 111 nodes in total: 1 with PC role, 10 with RC role, and 100 with DC role.

The influences between nodes determine how the intrusive actions that affect one node are propagated to the connected nodes. In our experimental work, we establish different values for these influences, according to different data distributions. In the following sections, we retake this issue and detail how the influences are established.

5.3. Adversarial Model

Once the system is modeled, we establish the attack probabilities on each node and their impacts. For all the IDNs, we consider the same adversarial model. Specifically, we consider a powerful adversary who attacks all the DC nodes that collect data locally, with a probability equal to 1. These DC nodes only send data to a correlation node and do not emit responses. Thus, the impact of the attacks in DC nodes is low, as opposite to the impact in the correlation nodes, which is one hundred times greater. Since the impact is greater in the correlation nodes, and because only the DC nodes are targeted, the risk of the entire IDN depends on how the risk propagation within the IDN affects other systems.

5.4. Allocation Module

The allocation module requires as input the cost of the countermeasures and the output of the threat module, i.e. the definition of which nodes are at risk and which countermeasures should be taken. Optimal allocation of countermeasures depends on two factors, the residual risk of the network after the countermeasures are implemented, and the cost of these countermeasures. On the one hand, the risk of the network is affected by the propagation of attacks, which in turn depends on the influences between nodes. On the other hand, the cost of the countermeasures varies from one node to another.

We conduct experiments on IDNs built using different values for the influences and costs of the countermeasures. Concretely, these values are established following four different data distributions (in the following, we refer to the value of influence or cost given to each node as *quantity*):

1. *Fractional (F).* As shown in Figure 9-(a), with this distribution the highest value is given to a single node, and then the lower the quantity is, the more the number of nodes that receive it.
2. *Exponential (E).* This distribution is shown in Figure 9-(b). With this distribution, the lower the quantity is, the fewer the number of nodes receive this quantity. Thus, the highest quantity is given to many nodes.
3. *Gaussian (G).* The assignment follows a normal distribution. The majority of the nodes receive a medium quantity, while some few nodes receive low quantities and some few nodes receive high quantities. Figure 9-(c) shows this distribution.
4. *Uniform (U).* All the nodes receive the same quantity.

In all the experiments, the total cost of the network distributed among the nodes is 100. This means that, if we consider an IDN with 100 nodes and a uniform distribution, the cost to defend each node is one.

For each distribution of influence and each distribution of cost, we use DEFIDNET to analyze the tradeoff between the cost of the countermeasures and the mitigated risk. Thus, for each architecture we conduct experiments on 16 IDNs.

5.5. Analysis of the Obtained Results

We have run experiments using four distributions of influence, and for each of them, four distributions of cost, resulting in 16 different configurations for each network architecture. The analysis of the solutions on these IDNs provides insights on how an IDN built with different distributions of costs and influences may have different defense strategies. As DEFIDNET uses randomization in the allocation search (i.e., the multi-objective evolutionary search), we repeat each simulation 10 times.

The experiments have been conducted in a Intel Core 2 Duo with two core processors at 3.00 GHZ and 4GB of RAM. We set the number of generations of the evolutionary search to 150 and the population size to 2500 individuals. The average simulation time is 28 minutes and 14 seconds per run for the centralized architecture, and 33 minutes and 21 seconds per run for the hierarchical architecture. These times depend on the machine used to run the simulations and the parameters of the evolutionary algorithm.

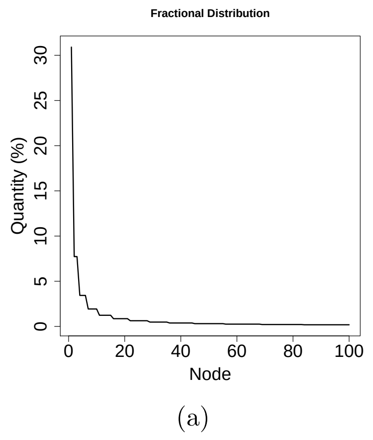
In general, different runs on the same simulated IDN are similar and only small fluctuations are noticeable. Figure 10) shows the Pareto fronts obtained with two different configurations: hierarchical and centralized architectures. It can be observed that the 10 runs of the algorithm obtains similar Pareto fronts. In the next sections, we consider the best Pareto front obtained in each configuration (i.e., the Pareto front whose points are closer, in average, to the origin) and analyze what placement strategy it suggests.

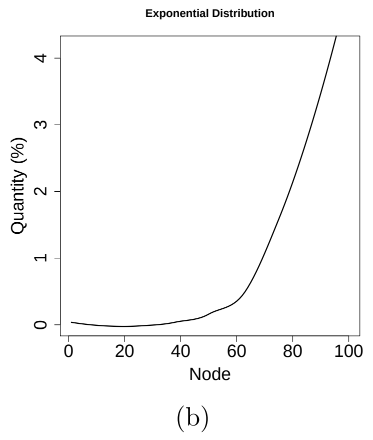
We provide the results both numerically, showing the actual percentages of risk mitigated and associated cost by the solutions, and graphically, plotting the Pareto fronts and analyzing their trend lines.

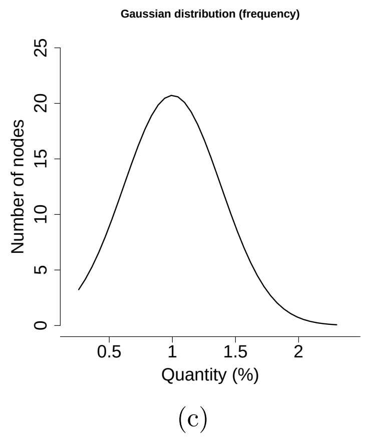
5.5.1. Numerical Analysis

Table 3 shows the quartiles of the amount of mitigated risk. Each quartile indicates the percentage of the cost required to mitigate the 25% (Q1), 50% (Q2), 75% (Q3), and 100% (Q4) of the risk. The total cost is calculated by applying all the countermeasures in the network, i.e. spending all the resources.

The symbols F, E, G and U corresponds to fractional, exponential, gaussian and uniform distributions respectively. We have highlighted the lowest values for each quartile, which indicates which the cheapest







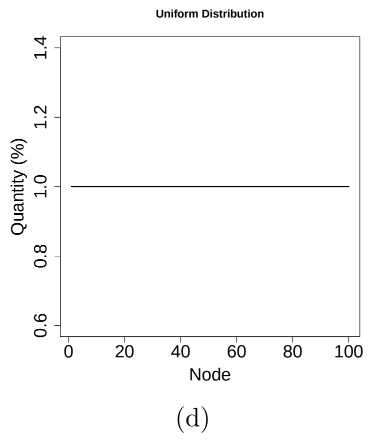
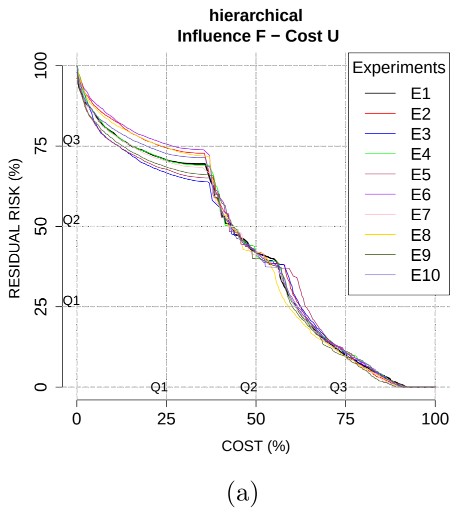


Figure 9: Data distributions used. The plots show the percentage of cost or influence (quantity) assigned to each node in a network of 100 nodes. In the Gaussian distribution, we show the frequency plot instead of the real distribution.



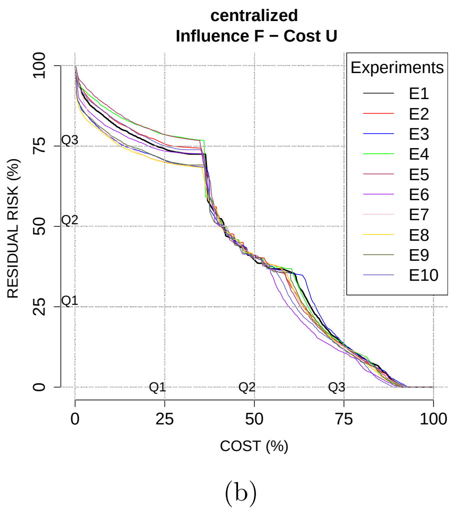


Figure 10: Pareto fronts obtained in 10 simulations, in configurations obtained establishing the influences with the fractional distribution and the costs with the Gaussian distribution, for both the hierarchical (a) and centralized (b) architectures.

configurations are. These values indicate the situations where some distributions of cost and influence are better than others. For example, in both architectures, IDNs with fractional distribution of influence and exponential distribution of costs (referred as IF-CE) are the cheapest to reduce the risk a 25%. Indeed, this situation happens because under these distributions of influences and costs, some nodes present low cost and big influence values. Thus, establishing countermeasures on these nodes is cheap while the risk is considerably reduced. Though this result may seem obvious, when a more complex IDNs is faced, it is not straightforward to decide the best strategy to apply countermeasures to defend the nodes.

Table 3: Percentage of the cost required to mitigate 25% (Q1), 50% (Q2), 75% (Q3), and 100% (Q4) of the risk in centralized and hierarchical IDNs. The first two columns indicate the distribution of the influence and cost, respectively.

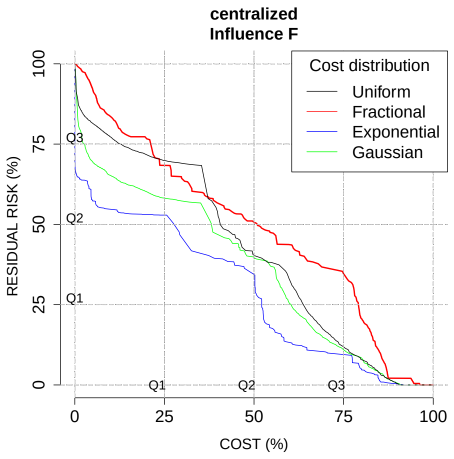
Distributions		Centralized architecture				Hierarchical architecture			
Influence	Cost	Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4
F	F	21.61	50.94	79.15	96.3	14.06	35.62	75.54	87.49
	E	**3x10^{4}**	29.12	52.15	92.11	**2.49**	29.82	54.24	90.06
	G	2.83	38.51	60.45	91.11	14.69	42.66	60.62	90.11
	U	12.73	40.61	63.84	90.71	11.8	43.2	62.6	91.0
E	F	1.85	24.48	44.58	93.49	3.89	24.6	47.72	95.45
	E	23.24	50.56	82.33	90.21	11.4635	39.87	71.51	90.47
	G	11.43	43.7	60.43	91.56	12.4	45.74	64.09	90.7
	U	13.74	46.26	61.41	89.7	18.2	46.6	61.0	90.6
G	F	9.28	28.69	63.32	88.1	7.66	25.33	55.29	89.5
	E	9.1	37.84	63.57	91.54	5.78	36.33	61.59	95.24
	G	21.03	50.36	68.62	90.67	19.16	48.66	69.26	89.45
	U	22.83	51.52	69.09	91.11	18.4	49.6	67.2	88.4
U	F	8.43	29.67	54.04	85.92	9.13	26.9	51.12	93.45
	E	9.23	40.16	68.77	90.85	10.01	36.27	62.27	90.34
	G	22.05	50.18	72.84	92.93	22.09	49.22	70.84	91.35
	U	26.26	52.12	69.09	87.88	22.4	51.2	70.4	91.2

5.5.2. Visual Analysis

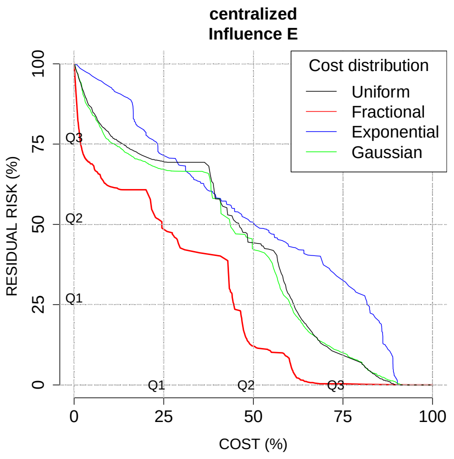
Figures 11 and 12 show the solutions of the Pareto fronts obtained for the centralized and hierarchical networks, respectively. For the sake of illustration, we have grouped the Pareto fronts in 4 plots, each one corresponding to an influence distribution. Each plot shows the percentage of risk (y-axis) mitigated and the percentage of cost (x-axis) associated with the solutions. A 100% of risk means that no countermeasures are applied (i.e. 0% of the cost spent), while a 100% of the cost means that all the required countermeasures to protect the network are applied. As previously stated, the associated Pareto front for each configuration corresponds to the best one obtained from a set of 10 simulations.

Assume that, in a given time, some countermeasures have been applied in the IDN, and thus the risk has been reduced partially. At this point, is it better to spend more resources to further reduce the risk, or is it better to do nothing, thus saving resources? The analysis of the trend line of the obtained Pareto front helps to answer these questions. We next provide two examples:

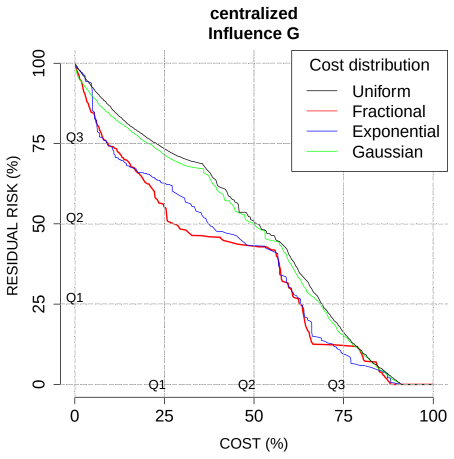
- *Example 1.* Consider the IDN with centralized architecture, with influences established following a gaussian distribution and costs established following a fractional distribution. The Pareto front is depicted in green in Figure 11-(c). The trend line indicates that spending a 25% of the cost (Q1) reduces almost the same risk than spending half of the cost (Q2). Thus, considering the network risk has been reduced up to 50% (using 25% of the budget), a substantial reduction of risk would require to spend around 30% more. In this case, if there are no restrictions about the risk, an intelligent decision would be to save resources and do not implement more countermeasures.



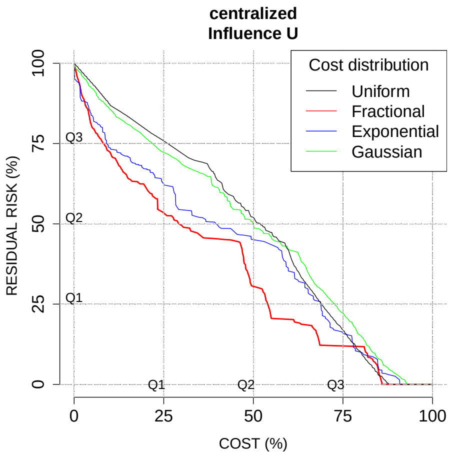
(a)



(b)



(c)



(d)

Figure 11: Cost-risk tradeoff in the centralized IDN. Each plot shows the Pareto front obtained for each cost distribution and one influence distribution.

- *Example 2.* Consider now the IDN with hierarchical architecture, with influences established following a fractional distribution and costs established following a gaussian distribution. The Pareto front is depicted in red in Figure 12-(a). In this case, with 40% of the cost, the risk is only reduced to a 70%. However, analyzing the trend line, it can be observed that with only some more cost (around 15% more) the risk is drastically reduced to less than 50%. Thus, contrarily to *Example 1*, it is worth spending some more resources because it significantly reduces risk.

Similarly to the numerical analysis, the trend line of the Pareto fronts can be used to analyze how to defend IDNs with different configurations. In such a case, the decision of which configuration is better is done by analyzing the lines: the line "arriving" first to the desired quartile is easier to defend, as it requires less cost to mitigate the same risk. In the centralized network, for example, with fractional influence (IF), the blue line (CE) is the first to reach the third quartile (Q3) of the risk. Accordingly, the centralized IDN with the IF-CE configuration is the cheapest to defend if a 75% of risk in the organization can be assumed. This result matches with the numerical analysis done in the previous section.

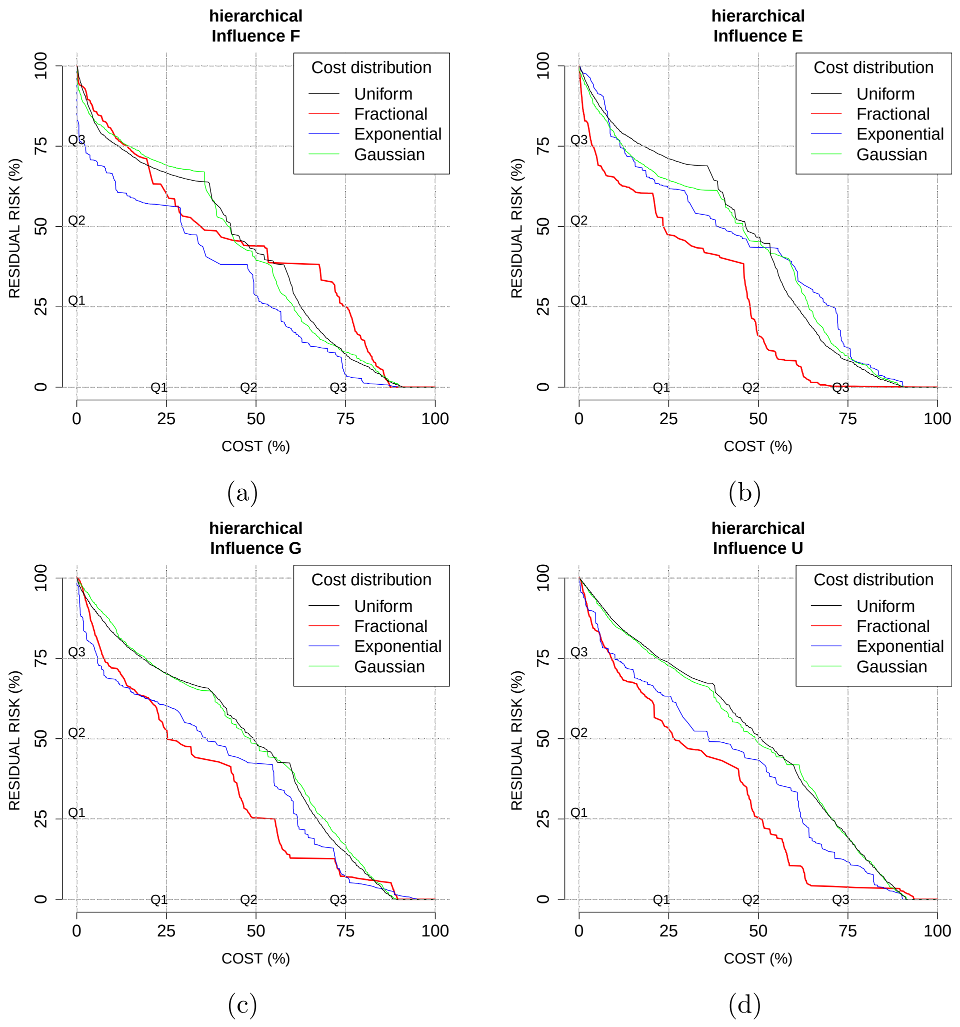


Figure 12: Cost-risk tradeoff in the hierarchical architecture. Each plot shows the Pareto front obtained for each cost distribution and one influence distribution.

5.6. Discussion

In this section we have discussed how DEFIDNET helps to decide when it is better to spend more resources to reduce the risk, and where. In many networks, establishing countermeasures in a indiscriminate manner may reduce the risk proportionally to the cost, i.e. spending more resources involves reducing more the risk. Graphically, the trend line of the Pareto front of this approach would have a constant decrement, like the green and black lines of Figures 11-(d) and 12-(d). However, the analysis performed in this section shows that in some IDNs, the previous effect is not true, and increasing the cost does not always translates into a risk reduction. In this case, the trend line of the Pareto front would have substantial leaps, like the lines analyzed in this section.

In order to save resources, it is useful to know when it is convenient to allocate new countermeasures, and where should they be placed. The decision depends on several parameters, like the architecture of the network, the influences between nodes, the cost of setting countermeasures in the nodes, etc. In the experiments presenteda bove, we have analyzed different configurations for rather small and simple IDNs using both centralized and hierarchical architectures. We defined such configurations by assigning influences and costs using different data distributions. However, when dealing with bigger networks, the value of

DEFIDNET is even greater. In the next section, we provide experimental results using a larger and more complex IDN than those analysed in this section.

6. Case Study

In this section, we use DEFIDNET to analyze a Cooperative Intrusion Detection Network (CIDN). In a CIDN, several entities (companies, organizations, governs, etc.) share the information that they have obtained from their own corporate network. Each of these corporate subnetworks are proprietary of one entity, and thus they may have different architectures, sizes, and also the adversarial model varies for each of them.

In this section, we first define the IDN and all its parameters. Second, we show the tradeoff between cost and risk of different solutions in terms of the Pareto front. Finally, we select one specific solution according to a budget and discuss the countermeasures that this solution suggests.

6.1. Architecture Design and Adversarial Model

Figure 13 shows the IDN used for the case study, once the probabilities are entered and propagated. The colors in the figure represent the risk level of each node in descending order from red (highest), orange, yellow, blue, and green (lowest). The network simulates a scenario where five entities share information of intrusion attempts in their corporate subnetworks. Each corporate subnetwork has a global node in charge of the communication with global nodes from other corporations. The global nodes correlate data received from their corporate subnetwork and from the other global nodes. If needed, global nodes emit responses. Because the corporations are independent and they are vested with the same authority, the global nodes are interconnected in a distributed fashion, and each of them is connected with all the remaining global nodes. Next we describe each corporate subnetwork:

1. **Subnetwork A**. Within this network, 100 nodes with role DC gather data locally and send it to the global node. The influence of these nodes to the global one is established with a fractional distribution, and the costs of each DC node is established with an exponential distribution. This subnetwork is highly targeted: 70% of the DC nodes have full probability of being attacked (i.e., the probability for each intrusive action is set to 1).
2. **Subnetwork B**. This network has a centralized architecture but, unlike the previous network, it has only 20 nodes collecting data. Moreover, these nodes do not communicate directly with the global node of the entity, but with a proxy who actually communicates with this global node. The distribution of influences and costs of the subnetwork are uniform, and only 10% of the nodes are targeted.
3. **Subnetwork C**. The global node of this entity is the root of a hierarchical subtnetwork. The hierarchy has two middle-level nodes (with role RC), each of them correlating data gathered from five DC nodes. Thus, there are 13 nodes in total. This network is completely targeted, and thus all the nodes have their probability of being attacked equal to one. The distribution of the influences is uniform, and the distribution of the costs is fractional.
4. **Subnetwork D**. This subnetwork uses an architecture that is an ensemble of a distributed architecture and a centralized architecture, composed by a total of 15 nodes organized as we explain next. First, five nodes are interconnected within a distributed ring (i.e., a node is connected to just another node and receives data from a different one). At the same time, each of these nodes correlates data received from three nodes that gather data locally, and thus is the central node of the corresponding centralized subnetworks. One of the nodes in the ring, acting as proxy, is connected to the global node of this entity. Both the influences and costs are uniformly distributed, and only three nodes in the lower-level of the hierarchy are targeted.
5. **Subnetwork E**. The detection within this network is distributed and all the 50 nodes have the same responsibility. Each node has the role LDA and is connected to 10% of the remaining nodes of the subnetwork. One of the nodes is the proxy connected to the global node. The distributions of the costs and influences are uniform. Only a small subset of these nodes are targeted (5%). However, it can

be observed in Figure 13 that, due to the propagation of the probabilities, almost every node within this network is put at risk. The impact of compromising each of these nodes, though, is one-hundred times lower than the impact of compromising the global node of this entity (following we explain the impacts assigned to the nodes of the network).

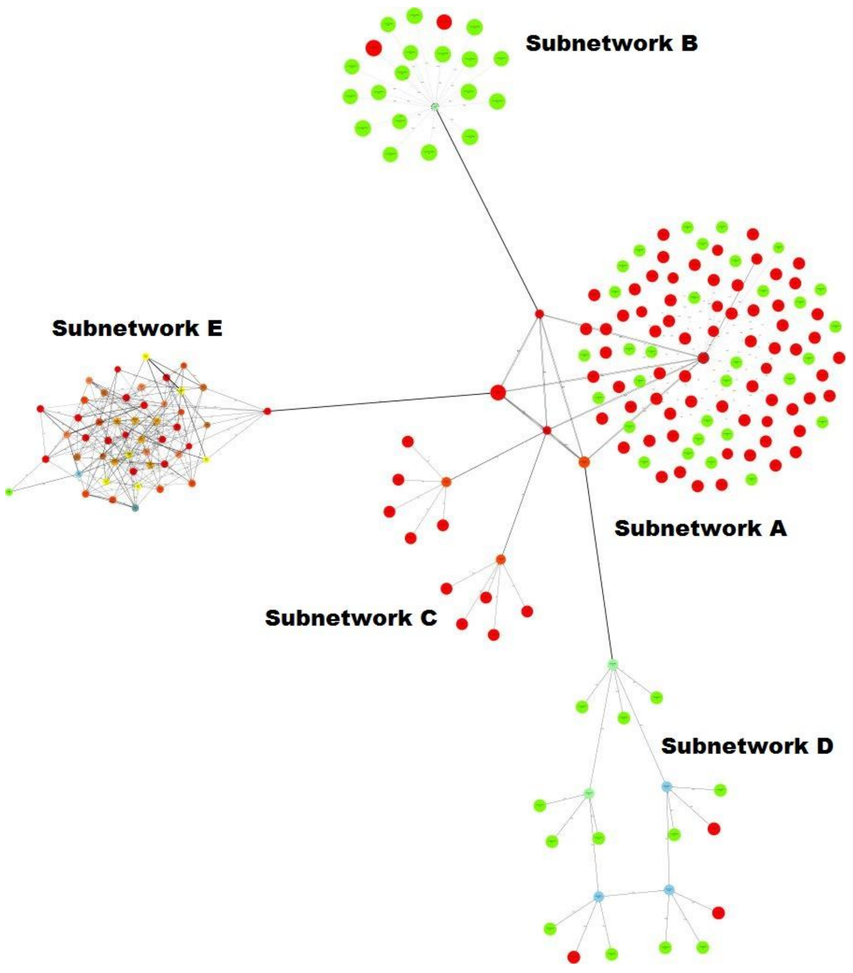


Figure 13: CIDN for the case study in its initial state.

Each global node, besides cooperating with the remainder global nodes, generates responses.[3] Accordingly, the impact of attacks on these nodes is higher than in the remaining nodes of the network. The impacts established for each node role are:

- Global nodes: 100
- Proxy nodes (those that are connected to a global node): 10
- Data Collector and peers in the partially-distributed: 1

A priori, it is not easy to determine where it is better to set countermeasures in this network. Should every node in the centralized-100 subnetwork being protected first?, or is it better to protect the hierarchical subnetwork because it is smaller? We next use DEFIDNET to analyze the cost-risk tradeoff by plotting the Pareto front obtained. Then, we analyze specific solutions suggested by DEFIDNET.

[3]Note that we do not consider in our study the specific responses emitted by nodes

6.2. *Cost-Risk Tradeoff*

Figure 14 shows the Pareto front of the solutions for the cooperative network. As explained above, each point in the Pareto front correspond to an optimal solution for each risk level with the least cost. In the figure, we have highlighted certain points which we analyze further in the next section. The trend line of the Pareto indicates that, spending few resources in the beginning, the risk rapidly decreases, and spending approximately 10% of the cost, the risk decreases to 75%. In that point, the gradient of the line becomes zero for an appreciable interval. That means that there is no improvement in the mitigated risk spending 15% and 22% of the cost. In the next subsection we analyze these two solutions and analyze why this situation occurs. Suddenly, the trend line substantially falls below 50% (between the orange triangle and the blue circle points), which means that, a solution that spends only 7% more of the cost than the previous one, reduces the risk 24%. Again, in the next section we analyze why this improvement occurs.

Following the analysis of the trend line, we observe that once the risk has been reduced below 50%, the trend line relatively enters in a decreasing interval for a while. However, it can be observed that it is required a greater percentage of the cost than the corresponding risk reduction. Finally, just before the line crosses the third quartile of the risk (i.e., just before reducing the risk below 25%), the trend line suffers a significant decrease, and the network becomes practically secured by spending 80% of the cost.

The analysis of this specific line shows that, if the budget for defending the network is low, then a good option is to reduce the risk at least 50%, because it can be achieved with little more than 25% of the budget. However, if the budget is high, it is better to spend the 80% because in such a case the risk is practically reduced to zero.

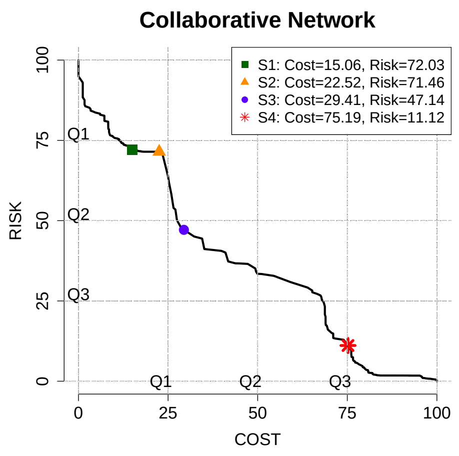


Figure 14: Pareto front showing the cost-risk tradeoff of the CIDN studied. The highlighted points are the solutions analyzed in Section 6.3.

6.3. *Analysis of Specific Solutions*

As explained above, each point in the Pareto front corresponds to one solution in the network, i.e. the set of countermeasures to be applied to optimally reduce some risk using some cost. In this section, we analyze four solutions suggested by DEFIDNET, which are highlighted in Figure 14.

The two solutions S1 and S2 mitigate little risk and spend few resources. The difference between them is that, while S2 mitigates only 1% more than S1, it costs 7% more. However, analyzing the line in the point S2, it can be seen that spending again another 7% of the cost provokes that the risk is reduced 25% (as seen in the solution S3 highlighted in Figure 14). What exactly is the difference between S1, S2 and S3?

In Table 4 we show the number of countermeasures that solutions S1, S2, and S3 allocate in each of the corporate subnetworks. The biggest different between solutions S1 and S2 is in the subnetwork centralized-100 and the subnetwork distributed. This difference suggests that setting countermeasures in these subnetworks does not reduce substantially the risk, unless all the risk in these subnetworks is mitigated. In the case of the centralized-100 subnetwork, the problem is that all the DC nodes are directly connected to the global node. In the case of the distributed subnetwork, the problem is that a single targeted node will propagate the risk throughout the entire network, and because one of the participants in this subnetwork is directly connected to the global node, the attack is propagated to the entire CIDN.

Regarding the difference between the solutions S2 and S3, it can be observed that in the hierarchical subnetwork, the defenses allocated by S3 are doubled, and even in the centralized-20 there is one less countermeasure. It can be observed that it spends more resources in protecting the hierarchical subnetwork and the overall risk is considerably reduced. The costs in the hierarchical network follow a fractional distribution, which means that some of the nodes have a high cost while the remainder has low cost. This means that, if there is enough budget to protect the highest cost nodes, then it is worth doing so, because the remaining nodes are cheap and then it becomes easier to protect the entire subnetwork.

Figure 15 shows the countermeasures allocated per action type. As it can be observed, in the solution S4, countermeasures to interception are almost the same than in solutions S3 and S2. This suggests that avoiding interception is not optimal unless there are enough resources. Actually, as shown in Table 2, the interception is only valuable for an adversary to perform a reverse engineering attack. Thus, protecting against interception only reduces the risk for reverse engineering. Another conclusion that can be extracted from Figure 15 is the importance of avoiding fabrication actions. Indeed, between S1 and S2, the countermeasures to protect against fabrication are the same, while there are considerable differences in the blocking, modification, and interception. As seen in Figure 14, solutions S1 and S2 mitigate almost the same risk, S2 being a 7% more expensive than S1. Solution S3, though, by spending a few more resources than S2 to protect nodes against fabrication, decreases the risk considerably.

Table 4: Analysis of the solutions highlighted in Figure 14.

Solution	Number of countermeasures in each subnetwork				
	Centralized-100	Centralized-20	Hierarchical	Ring	Distributed
S1	96	3	5	3	7
S2	131	3	7	4	11
S3	139	2	14	6	14
S4	167	5	33	11	33

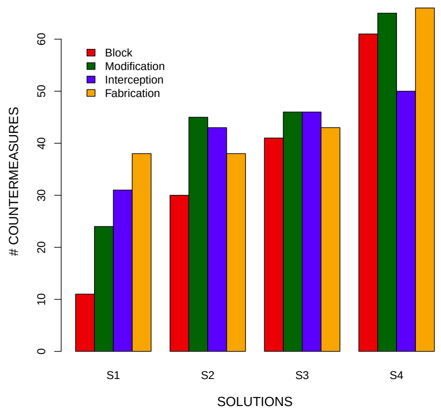


Figure 15: Number of countermeasures per action type.

Finally, Figure 16 shows the CIDN after the solution S4 is applied. It can be observed that, as it was suggested in the analysis of solutions S1, S2, and S3, when allocating countermeasures, the optimal approach is to leave the centralized-100 network at the end of the queue and protect the other subnetworks first.

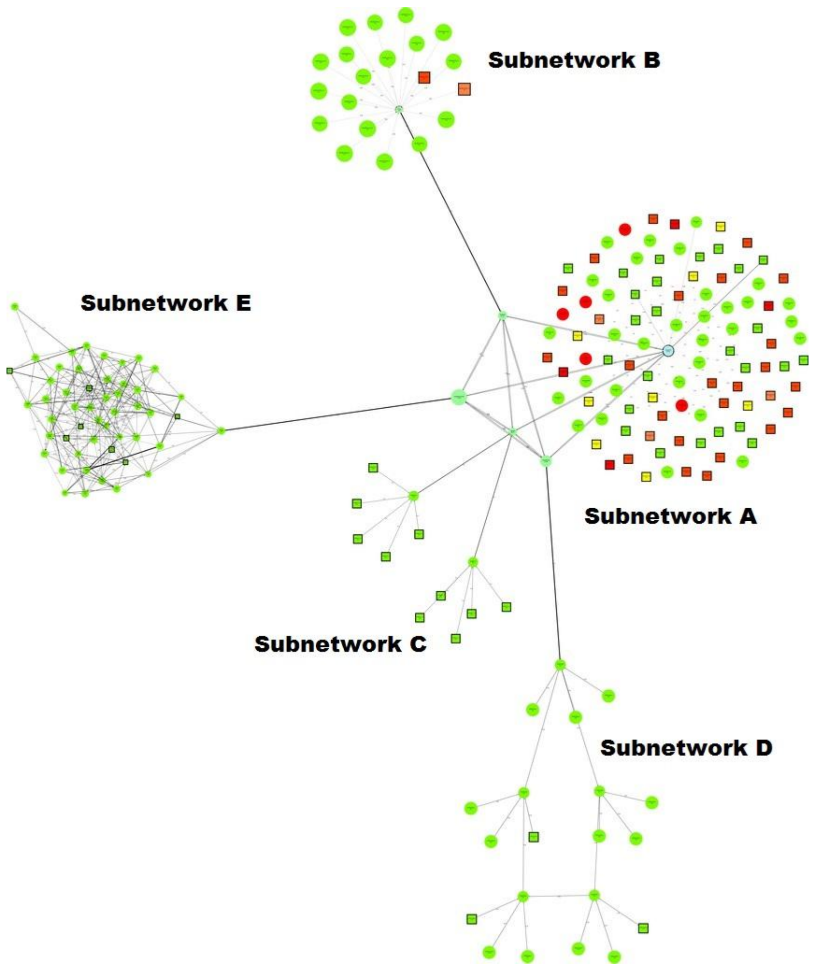


Figure 16: CIDN after applying the countermeasures of the solution S4.

7. Conclusions

Intrusion Detection Networks are used to detect complex, distributed attacks. They aggregate several nodes with different roles that are interconnected to share information. Accordingly, a compromised node may expose the entire IDN to a risk. Due to the adversarial scenarios in which these networks operate, the design of robust architectures is critical to maintain an acceptable level of security.

In this paper, we have presented DEFIDNET, a framework that assesses the risk of IDNs against specific attacks in the nodes. Node abstraction allows the definition of single probabilities of intrusive actions in the channels of each node, which is simpler than defining the probability of complex attacks in the entire network. Then, considering these probabilities and their propagation throughout the network, the likelihood of different attacks being happening is calculated. These attacks are defined regarding its consequences on the IDN, and they have an associated impact. Using the likelihood and the impact of attacks, the risk of the network is calculated.

In order to save resources, it is important to analyze the tradeoff between cost and risk of implementing countermeasures in the channels. To this end, we use a Multi-Objective Optimization algorithm to get optimal allocations of these countermeasures. Concretely, we use an evolutionary algorithm known as SPEA2. This algorithm provides solutions that are Pareto optimal, where a solution is the set of countermeasures to be applied in order to protect the channels of the IDN nodes.

In our experimental results we have seen that, depending on the cost and the connections in the network, deciding what to fix and deciding whether it is worth doing can help saving resources. Moreover, we have provided an analysis of a case study that suggest that the use of DEFIDNET helps to determine the optimal allocation of resources in big complex networks.

References

[1] K. A. Scarfone, P. M. Mell, SP 800-94. Guide to Intrusion Detection and Prevention Systems (IDPS), Technical Report, Gaithersburg, MD, United States, 2007.

[2] D. E. Denning, An intrusion-detection model, IEEE Transactions on Software Engineering 2 (1987) 222–232.

[3] G. Suarez-Tangil, J. E. Tapiador, P. Peris, A. Ribagorda, Evolution, detection and analysis of malware for smart devices, IEEE Communications Surveys & Tutorials PP (2013) 1–27.

[4] M. Al Ameen, J. Liu, K. Kwak, Security and privacy issues in wireless sensor networks for healthcare applications, Journal of medical systems 36 (2012) 93–101.

[5] C. V. Zhou, C. Leckie, S. Karunasekera, A survey of coordinated attacks and collaborative intrusion detection, Computers & Security 29 (2010) 124–140.

[6] M. Roesch, Snort: Lightweight intrusion detection for networks., in: Proceedings of the 13th Systems Administration Conference, USENIX, Seattle, WA, USA, 1999, pp. 229–238.

[7] S. A. Crosby, D. S. Wallach, Denial of service via algorithmic complexity attacks, in: Proceedings of the 12th USENIX Security Symposium, Washington: USENIX, 2003, pp. 29–44.

[8] R. Smith, C. Estan, S. Jha, Backtracking algorithmic complexity attacks against a NIDS, in: Computer Security Applications Conference, 2006. ACSAC'06. 22nd Annual, IEEE, 2006, pp. 89–98.

[9] S. Pastrana, A. Mitrokotsa, A. Orfila, P. Peris-Lopez, Evaluation of classification algorithms for intrusion detection in MANETs, Knowledge-Based Systems 36 (2012) 217 – 225.

[10] R. Bye, S. A. Camtepe, S. Albayrak, Collaborative intrusion detection framework: characteristics, adversarial opportunities and countermeasures, in: Proceedings of CollSec: Usenix Workshop on Collaborative Methods for Security and Privacy, 2010.

[11] P. Fogla, W. Lee, Evading network anomaly detection systems: Formal reasoning and practical techniques, in: Proceedings of the 13th ACM Conference on Computer and Communications Security, ACM, Alexandria, VA, USA, 2006, pp. 59–68.

[12] S. Shin, G. Gu, Conficker and beyond: a large-scale empirical study, in: Proceedings of the 26th Annual Computer Security Applications Conference, ACM, 2010, pp. 151–160.

[13] F. Almenares, P. Arias, A. Marin, D. Diaz-Sanchez, R. Sanchez, Overhead of using secure wireless communications in mobile computing, Consumer Electronics, IEEE Transactions on 59 (2013).

[14] C. Xenakis, C. Panos, I. Stavrakakis, A comparative evaluation of intrusion detection architectures for mobile ad hoc networks, Computers & Security 30 (2011) 63–80.

[15] A. Patel, M. Taghavi, K. Bakhtiyari, J. Celestino Junior, An intrusion detection and prevention system in cloud computing: A systematic review, Journal of Network and Computer Applications 36 (2013) 25–41.

[16] M. Gil-Pérez, F. Gómez-Mármol, G. Martínez-Pérez, A. F. Skarmeta-Gómez, RepCIDN: A reputation-based collaborative intrusion detection network to lessen the impact of malicious alarms, Journal of Network and Systems Management 21 (2013) 128–167.

[17] C. Fung, R. Boutaba, Intrusion Detection Networks: A Key to Collaborative Security, CRC Press, 2013.

[18] M. Gil-Pérez, F. Gómez-Mármol, G. Martínez-Pérez, A. F. Skarmeta-Gómez, Building a reputation-based bootstrapping mechanism for newcomers in collaborative alert systems, Journal of Computer and System Sciences 80 (2014) 571–590.

[19] C. Fung, Collaborative intrusion detection networks and insider attacks, Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications 2 (2011) 63–74.

[20] I. Corona, G. Giacinto, F. Roli, Adversarial attacks against intrusion detection systems: Taxonomy, solutions and open issues, Information Sciences 239 (2013) 201–225.

[21] D. Mutz, G. Vigna, R. Kemmerer, An experience developing an IDS stimulator for the black-box testing of network intrusion detection systems, in: Proceedings of the 19th IEEE Annual Computer Security Applications Conference, 2003, pp. 374–383.

[22] B. I. Rubinstein, B. Nelson, L. Huang, A. D. Joseph, S.-h. Lau, S. Rao, N. Taft, J. D. Tygar, Antidote: Understanding and defending against poisoning of anomaly detectors, in: Proceedings of the 9th ACM SIGCOMM Conference on Internet Measurement Conference, IMC '09, ACM, New York, NY, USA, 2009, pp. 1–14. URL: `http://doi.acm.org/10.1145/1644893.1644895`. doi:`10.1145/1644893.1644895`.

[23] S. Chung, A. Mok, Allergy attack against automatic signature generation, in: Recent Advances in Intrusion Detection (RAID), Springer, 2006, pp. 61–80.

[24] S. Chung, A. Mok, Advanced allergy attacks: Does a corpus really help?, in: Recent Advances in Intrusion Detection (RAID), Springer, 2007, pp. 236–255.

[25] J. Newsome, B. Karp, D. Song, Polygraph: Automatically generating signatures for polymorphic worms, in: Security and Privacy, 2005 IEEE Symposium on, IEEE, 2005, pp. 226–241.

[26] D. Mutz, C. Kruegel, W. Robertson, G. Vigna, R. A. Kemmerer, Reverse engineering of network signatures, in: Auscert Asia Pacific Information Technology Security Conference, 2005.

[27] L. Minh Sang Tran, B. Solhaug, K. Stølen, et al., An approach to select cost-effective risk countermeasures, in: Data and Applications Security and Privacy XXVII, Springer, 2013, pp. 266–273.

[28] S. Sen, J. A. Clark, Evolutionary computation techniques for intrusion detection in mobile ad hoc networks, Computer Networks 55 (2011) 3441–3457.

[29] V. Yegneswaran, P. Barford, S. Jha, Global intrusion detection in the DOMINO overlay system., in: NDSS, 2004.

[30] A. Karim Ganame, J. Bourgeois, R. Bidou, F. Spies, A global security architecture for intrusion detection on computer networks, Computers & Security 27 (2008) 30–47.

[31] X. Zhang, Y. Sekiya, Y. Wakahara, Proposal of a method to detect black hole attack in manet, in: International Symposium on Autonomous Decentralized Systems. ISADS'09., IEEE, 2009, pp. 1–6.

[32] W. Li, J. Parker, A. Joshi, Security through collaboration and trust in MANETs, Mobile Networks and Applications 17 (2012) 342–352.

[33] H. Debar, D. Curry, B. Feinstein, The intrusion detection message exchange format (IDMEF), 2007. URL: `http://tools.ietf.org/html/rfc4765`, rFC-4765. http://tools.ietf.org/html/rfc4765.

[34] V. Paxson, Bro: a system for detecting network intruders in real-time, Computer networks 31 (1999) 2435–2463.

[35] D. Miller, B. Pearson, Security information and event management (SIEM) implementation, McGraw-Hill, 2011.

[36] M. Gil-Pérez, J. E. Tapiador, J. A. Clark, G. Martínez-Pérez, A. F. Skarmeta-Gómez, Trustworthy placements: Improving quality and resilience in collaborative attack detection, Computer Networks 58 (2014) 70 – 86.

[37] D. Wagner, P. Soto, Mimicry attacks on host-based intrusion detection systems, in: Proceedings of the 9th ACM Conference on Computer and Communications Security, ACM, Washington, DC, USA, 2002, pp. 255–264.

[38] G. Stoneburner, A. Goguen, A. Feringa, Risk management guide for information technology systems, Nist special publication 800 (2002) 800–30.

[39] M.-Y. Su, Prevention of selective black hole attacks on mobile ad hoc networks through intrusion detection systems, Computer Communications 34 (2011) 107–117.

[40] G. Vigna, W. Robertson, D. Balzarotti, Testing network-based intrusion detection signatures using mutant exploits, in: Proceedings of the 11th ACM Conference on Computer and Communications Security, ACM, Washington, DC, USA, 2004, p. 21.

[41] S. Marti, T. J. Giuli, K. Lai, M. Baker, et al., Mitigating routing misbehavior in mobile ad hoc networks, in: International Conference on Mobile Computing and Networking, volume 6, 2000, pp. 255–265.

[42] S. Pastrana, A. Orfila, J. E. Tapiador, P. Peris-Lopez, Randomized anagram revisited, Journal of Network and Computer Applications 41 (2014) 182–196.

[43] R. Shirey, Rfc 4949: Internet security glossary, 2007. URL: `http://tools.ietf.org/html/rfc4949`, status: INFORMATIONAL.

[44] E. Zitzler, M. Laumanns, L. Thiele, SPEA2: Improving the strength pareto evolutionary approach, EUROGEN 2001. Evolutionary Methods for Design, Optimization and Control with Applications to Industrial Problems (2001) 95–100.

[45] S. Sen, J. A. Clark, J. E. Tapiador, Power-aware intrusion detection in mobile ad hoc networks, in: Ad Hoc Networks, volume 28 of *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*, Springer, 2010, pp. 224–239. URL: `http://dx.doi.org/10.1007/978-3-642-11723-7_15`.

[46] S. Luke, The ECJ Owner's Manual – A User Manual for the ECJ Evolutionary Computation Library, Technical Report, 2010. URL: `http://www.cs.gmu.edu/~eclab/projects/ecj/docs/manual/manual.pdf`, department of Computer Science, George Mason University.

[47] K. Deb, S. Agrawal, A. Pratap, T. Meyarivan, A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA2, LNCS 1917 (2000) 849–858.