Optimal countermeasures selection against cyber attacks: A comprehensive survey on reaction frameworks

Pantaleone Nespoli, Dimitrios Papamartzivanos, Félix Gómez Mármol, Georgios Kambourakis

***Abstract*—It is without doubt that today the volume and sophistication of cyber attacks keeps consistently growing, militating an endless arm race between attackers and defenders. In this context, full-fledged frameworks, methodologies, or strategies that are able to offer optimal or near-optimal reaction in terms of countermeasure selection, preferably in a fully or semi-automated way, are of high demand. This is reflected in the literature, which encompasses a significant number of major works on this topic spanning over a time period of 5 years, that is, from 2012 to 2016. The survey at hand has a dual aim, namely: first, to critically analyze all the pertinent works in this field, and second to offer an in-depth discussion and side-by-side comparison among them based on 7 common criteria. Also, a quite extensive discussion is offered to highlight on the shortcomings and future research challenges and directions in this timely area.**

***Index Terms*—Cyber attack countermeasures, Security risk assessment, Intrusion Prevention and Response systems, Decision support systems, Optimal countermeasure strategy, Dynamic reaction selection.**

I. Introduction

Nowadays, computer networks play a central role in most enterprises and critical infrastructures, determining their economic success. Billions of diverse interconnected devices exchange data among each other, having as an ultimate goal to provide services to humans. According to estimations [1], 38.5 billion of devices will be connected to the internet by 2020, creating and sharing sensitive information around the globe. In this setting, two main challenges are introduced:

- *Heterogeneous data*: Different kind of devices, use standardized but also diverse protocols to exchange information [2];
- *Large amount of events*: A massive amount of information and events flow into information and communication systems [3].

On the downside, these densely interconnected ecosystems became a playground for ill-motivated entities, which not only aim to manipulate the exchanged information and threaten their confidentiality and integrity, but also inflict negative financial consequences to organizations by affecting the availability of the provided services. In this direction, both the academia and the information industry struggle to come up with advanced strategies against offensive incidents. This is mostly achieved via the design of innovative detection systems and defense plans [4]. Although significant effort has been put in protectively detecting harmful activities, this is not enough for protecting the targeted systems. Nowadays, there is more than ever a need to reactively counteract intrusive events upon their occurrence in order to dynamically eradicate potential consequences on the protected systems with the aim of minimizing the security risk and financial losses. In this work, we will refer to this approach as *Countermeasure Strategy*, explaining our understanding of the problem and underpinning its major components.

Battling against intrusive incidents occurring in a network infrastructure is a challenging task due to several parameters, including the attacker's level of expertise, the variety and sophistication of attacks, the network size and topology, the number and diversity of zero-day vulnerabilities, the robustness of the deployed Intrusion Detection Systems (IDS), etc. [5]–[8]. Security administrators bare the burden of dealing with such demanding tasks, and most of the time, they have to manually react against intrusions while having security budget constraints and a strict reaction time frame. Putting it another way, security administrators often face multi-criteria decision making (MCDM) problems that have to be solved in a timely and cost-effective manner [9]. Also, by taking into consideration other fundamental parameters of the problem, one can safely argue that it is almost infeasible for a human to appropriately deal with all these requirements. As a result, efficient fully or semi-automated decision support systems are needed to address the resource bottleneck by human operators and security officials [10].

Until now, several works in the literature propose cost-benefit quantitative approaches in applying optimal countermeasure strategies. At a high level, all these contributions share a common goal; they define an optimal set of countermeasures to counteract cyber attacks. In this direction, a plethora of methodologies have been introduced by incorporating a diversity of fundamental notions. More specifically, as detailed in the following sections, cyber attack modeling methodologies such as Attack Graphs (AGs), Attack Trees (ATs) [11] and other graphical structures are used to accu-

P. Nespoli is with the Department of Information and Communications Engineering at the University of Murcia, 30100 Murcia, Spain, e-mail: pantaleone.nespoli@um.es

D. Papamartzivanos is with the Department of Information & Communication Systems Engineering, University of the Aegean, 83200 Samos, Greece, e-mail: dpapamartz@aegean.gr.

F. Gómez Mármol is with the Department of Information and Communications Engineering at the University of Murcia, 30100 Murcia, Spain, e-mail: felixgm@um.es

G. Kambourakis is with the Department of Information & Communication Systems Engineering, University of the Aegean, 83200 Samos, Greece, and with Computer Science Department, George Mason University, Fairfax, VA 22030, USA. e-mail: gkamb@aegean.gr.

DRAFT

rately portray the interconnections and possible dependencies among the various network assets. In particular, ATs aim to formally represent the security states of a system and to visualize the different ways in which it can be attacked using a tree structure. On the other hand, AGs combine information pertaining to network topology and possible vulnerabilities and exploits, with the aim of providing a visual representation of the attack paths that a potential aggressor could traverse in an effort to reach a specific network target. Alongside with these graphical representations, several probabilistic models have been used to describe the system's security state transitions, which constitute the actual search space of the problem. To do so, several optimization algorithms and quantitative risk assessment methods have been recruited, and combined together, to deliver competent solutions able to provide optimal sets of countermeasures regarding the system's security states. In a nutshell, the works presented in the literature can be divided into two major categories, namely *static* and *dynamic* reaction systems. The former are used to proactively secure a monitored system, while the latter are destined to operate reactively upon the occurrence of an attack incident [12].

The numerous solutions and their diverse above-mentioned characteristics spur us on to provide a comprehensive analysis of the pertinent literature and present the state-of-the-art frameworks in this ecosystem. Specifically, to the best of our knowledge, this is the first survey to focus on works which aim to provide countermeasure recommendations as a result of automated processes driven by quantitative security metrics. By going through the literature, we identified across all works 7 common basic qualitative features (detailed in Subsection III-B) and used them as the basis for comparing the capabilities of the proposed frameworks. The goal of our analysis is to identify the pros and cons of the presented works and to shed light on the limitations of this particular field. That is, by breaking down the proposed solutions and comparing their characteristics, we identified several research challenges that should be taken into account by researchers intending to contribute in the field.

It has to be stressed out that our work should not be analogized to others (such as those in [13]–[15]) that concentrate on recommending individual countermeasure actions against specific types of attacks and Information and Communications Technology (ICT) environments. Contrary to these contributions, we intend to provide a detailed overview and side-by-side comparison of solutions which incorporate methods aiming to propose an optimal set of countermeasures to deal with ongoing attacks. Moreover, the reviewed works are field-independent as their concepts could be applied in a wide range of ICT domains. It is also worth underlining that although there exist other surveys in the area [16]–[18], they are mostly outdated and their focus is on the implementation aspect of the solutions rather than on the components used as a basis for their deployment. That is, our analysis emphatically focuses on the methods and theories applied fundamentally on the background of such solutions. Additionally, as already pointed out, we provide a side-by-side comparison of the reviewed contributions based on 7 key criteria as identified throughout the analysis of the various works.

The rest of the paper is organized as follows. The next section presents background information needed to introduce the reader in the field. An in-depth overview of the problem and the qualitative features used to compare the various works are given in Section III. A detailed description, analysis, and comparison among the works gathered from the literature is included in Section IV. The last but one Section provides a discussion on the research challenges in the field and offers pointers to future research. The last Section concludes by summarizing the most significant findings of our work.

II. Background

Cybersecurity is a term used to describe a set of technologies, strategies, practices, and policies used for protecting the cyberspace. The tremendous growth of ICT environments gave room to cyber crooks to threaten the confidentiality, integrity, and availability of the provided information and services. By exploiting system vulnerabilities, security policy flaws and even physical security gaps, attackers are able to get unauthorized access to systems and confidential information, while they can disrupt the availability of services and cause financial losses to network and service providers. As already pointed out, dealing against offensive incidents is a particularly difficult task because of the dynamic nature of ICT systems, the size of the attack surface, and the diversity of attack vectors [19]. New system vulnerabilities are reported on a daily basis, while the problem of cybersecurity is not usually limited by physical borders as the source of a malevolent action could be anywhere around the globe.

In light of those threats, the scientific community and the ICT industry strive to find ways to increase the robustness of cyber defense solutions. According to NATO Cooperative Cyber Defence Centre of Excellence, the term "cyber defense" can be defined as "*a proactive measure for detecting or obtaining information as to a cyber intrusion, cyber attack, or impending cyber operation or for determining the origin of an operation that involves launching a preemptive, preventive, or cyber counter-operation against the source*" [20]. As illustrated in Fig. 1, the previous definition can be conceptualized as a cyber defense cycle, consisting of four constituents, namely *Prevention, Detection, Reaction* and *Forensics*. These building blocks are strongly connected, as each of them feeds the next one in the cycle.

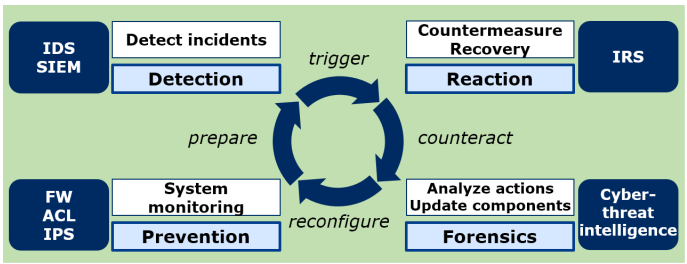


Fig. 1: Cycle of cyber defense

DRAFT

A. Prevention

The prevention phase is in charge of continuously monitoring the system aiming at discovering any vulnerability or misconfiguration within it. Once a weakness is found, violating security policies or increasing the risk level of the system, appropriate actions must be taken so as to patch it.

In this stage, different defense methodologies and tools may be applied. For instance, *firewalls* (FWs) are used to analyze packet contents and to enforce security access policies and logging in a centralized manner [21], [22]. *Access Control Systems* (ACS) manage user privileges regarding the access to specific assets within an organization [23]. Also, the use of *Intrusion Prevention Systems* (IPSs) is essential, since they are able to actively analyze network traffic flows, taking automated actions to prevent vulnerability exploits [24], [25]. Last but not least, *Risk Assessment* techniques [26], [27] combined with vulnerability scanning [28] represent powerful means to enumerate potential threats against the system, identify potential flaws, and assess its security risk level.

Yet, even in an ideal scenario, it would be unrealistic to consider the monitored system completely protected against any possible cyber attack. This observation relies on the fact that preventive strategies can never reach a 100% level of accuracy, due to potential misconfiguration recklessly provoked by human administrators and to the fact that software scanning procedures are prone to errors [29]. Moreover, the task of minimizing the attack surface is extremely hard, due to the size and complexity of the modern networks [30]. Considering also the presence of possible zero-day exploits and unknown vulnerabilities, it is clear that this phase is not a panacea towards defending the system.

B. Detection

Assuming that the system is or will be vulnerable, the detection phase is responsible for identifying and reporting every intrusive incident that entails the exploitation of a vulnerability or misconfiguration. Any detected activity or violation is typically logged and reported either to an administrator or collected centrally using a *security information and event management* (SIEM) system. The latter stores events stemming from multiple sources, and uses filtering, aggregation, and correlation techniques to distinguish between malicious activities and false alarms [31].

The most commonly used approach to perform the detection phase is the use of an IDS. There is a wide spectrum of IDS, which can be classified based on two main features, namely the placement of the detection entity (or sensor) and the detection method used. For the first feature, we distinguish:

- *Network IDS* - It is placed in strategic points of the network, monitoring all the traffic exchanged among the devices [32], [33];
- *Host-based IDS* - It runs on individual hosts within the network, monitoring the inbound and outbound traffic for these specific devices [34].

For the detection method, we distinguish:

- *Signature-based IDS* - It performs the detection matching of the incoming traffic using specific predetermined rules or patterns [32], [33];
- *Anomaly-based IDS* - It compares a model of normal behavior against the incoming traffic in order to find anomalies [35], [36].

In addition, a new FW technology is increasing its importance in the market. Precisely, Next-Generation FW [37] is a network security system which combines the capabilities of a traditional FW with other functionality, including antivirus inspection, IDS-IPS, TLS (Transport Layer Security) decapsulation, and identity management. By doing so, these firewalls move far beyond the usual port/protocol stateless/stateful inspection and blocking to add, say, deep application-level inspection, and thus bring intelligence from outside the firewall. That is, by simultaneously working on different layers of the OSI model, this new generation of firewalls improve the filtering capabilities of network traffic to a great deal.

C. Reaction

Once an intrusive incident has been reported by means of a correct detection, the reaction phase has to be fired, evaluating the impact of this event on the security level of the system [38]. This phase is tasked with a dual goal; first, it must provide the optimal (effective and efficient) set of countermeasures to quickly eradicate the attack, and second to indicate the set of actions to heal the system and bring it back to its normal state.

Regarding countermeasures provision, this task looks after the selection of the appropriate set of countermeasures, which must be taken for the sake of blocking the ongoing malicious activity. This selection has to balance an inherent trade-off between countermeasure cost (considering both economic expenses and impact on the system) and effectiveness in eradicating the attack.

In the literature, there is not a commonly agreed understanding on the meaning of *countermeasure*. ISO/IEC 27002:2005 defines countermeasure (a synonym of control or safeguard) as "*control means of managing risk, including policies, procedures, guidelines, practices or organizational structures, which can be of administrative, technical, management, or legal nature.*" [39]. Since risk management covers the entire life cycle of the cyber defense, adopting the aforementioned definition a countermeasure can be found in more than one component in the cyber defense cycle. That is, it can be associated to the reaction block for its ability to mitigate or eliminate attacks, and to the prevention one, as it can prevent them from happening. To resolve this ambiguity, in the context of this work, we shall refer to a *static* countermeasure, to indicate its preventive aspect, and to a *dynamic* one, to indicate its reactive capabilities.

A possible way to react is via the use of *Intrusion Response Systems* (IRSs), as they are IDSs capable of counteracting suspicious activities. Although intrusion response components are often integrated with the detection ones, they have received considerably less attention than IDS research. This discrepancy is mainly due to the challenges that an automated response poses, such as its adaptability to dynamic environments or

DRAFT

the complexity of the solution [40], [41]. Recently, significant effort has been put to provide commercial SIEMs with automated response capabilities [42], [43]. The main problem however is that the characteristics of these capabilities typically depend on the organization in which they are deployed; this mostly makes the reaction strategy unique to each possible scenario.

Recovery on the other hand aims at repairing the system and bringing it back to a safe state. This however entails the evaluation of the impact of the corresponding countermeasure [44]. This means that while the enforcement of the countermeasure must be done rapidly, its design, implementation, and maintenance introduce costs which should be also carefully considered [45]. For short-term reaction, say, blocking the incoming traffic on a specific port or redirecting the traffic of a specific source IP address, the invoked action can be easily removed reverting its effects, and thus restoring the system to a previous safe state . On the downside, for long-term reaction, say, patch an exploited vulnerability or add a security control in the FW, the implemented measure must be maintained for a long period, becoming in this way part of the security policy. In this case, information about the attack and the undertaken actions have to be shared with the other constituents of the cyber defense cycle.

D. Forensics

When the attack has been eradicated and the system has been healed, the forensics phase is in charge of analyzing the actions recorded in the log files of the system as those created by the previous phases. This would allow the interested parties to tell what went wrong and obtain feedback on how to avoid similar security incidents in the future. To achieve this goal, this component needs to collect and analyze a great quantity of events which contain evidences of attacks [46]. Once a deep inspection has been performed on the log files, a feedback is provided to every precedent component. Such feedback would be used to reconfigure the system, so as to make it more resilient against future attacks.

In this direction, threat sharing is a good practice to improve this phase. This means that companies worldwide can share information about the faced attacks and the used remediations, succeeding to build a stronger defense strategy [47]. This practice is known as *Cyber-threat intelligence*, in which defenders are at the same time consumers and producers of shared information about threats.

III. Problem Statement

As already pointed out, the goal of any reactive system (which is the focus of this work) is to assist the security administrators in the decision making for counteracting possible security incidents. Aiming at providing a holistic view of the problem, this section elaborates on the core ingredients of a *Countermeasure Strategy*, while at the same time underlines the most important features included in the various surveyed studies.

A. Countermeasure strategy

Current information systems contain a plethora of assets along with the associated security controls which aim to ensure a specific level of security for each of them. The volume of information produced by these controls is the baseline that has to be used for building a robust defense strategy, but at the same time, it represents also a problem by itself given its huge size.

In this context, security administrators have to face many security issues, including multistep, new and sophisticated type of attacks (or polymorphic ones [48]), asset exposures, distributed and heterogeneous physical network topologies, etc. Moreover, they have to work within specific budget constraints that may preclude them from implementing all possible hardening measures or even those that provide remedies to hopefully cover the great majority of weaknesses. In addition, their decision must consider time limitations, because usually in any reaction strategy the time is of essence. Thus, system administrators need to find the optimal trade-off between the implementation cost of a specific countermeasure and the overall security level of the system, also known as system administrator dilemma [49]. In the context of this paper, we define the term *countermeasure strategy* as follows.

Definition. [Countermeasure Strategy] A generic set of methodologies, procedures, and processes that aim at reacting to security incidents in a given system and eradicate them.

As detailed in Section IV, so far, many approaches have been presented in the literature to address the problem of finding the optimal combination of security and cost parameters in order to achieve the optimal collection of countermeasures. Based on our analysis, Fig. 2 shows the basic components of a countermeasure strategy. We define them as follows:

- *Monitored system*: the physical system to be protected; from this system, core information is extracted, including *network topology* and *asset configuration*.
- *Detection tools*: collection of tools that send all events happening within the monitored system to the appropriate controls of the countermeasure strategy. Such events include intrusion alerts, software updates, hardware installations, and so on. Examples of these tools are IDS, *Antiviruses* (AVs) and *FWs*.
- *Countermeasure knowledge*: initial raw knowledge about the reaction steps which must be triggered to cope with security issues. As detailed in Sections V-B and V-C, this knowledge is acquired from external sourcesand the security administrator, and is typically represented in heterogeneous formats. Few examples of such ilk of countermeasures are “close TCP/UDP ports”, “redirect incoming traffic”, “apply a patch”, etc..
- *Vulnerability reports*: it represents knowledge on the vulnerabilities. The sources of this information are the expertise of the system administrators, the public available databases, such as CVE [50], and the open threat sharing networks [47].
- *System model*: given the information gathered from the monitored system, such as network topology and config-

DRAFT

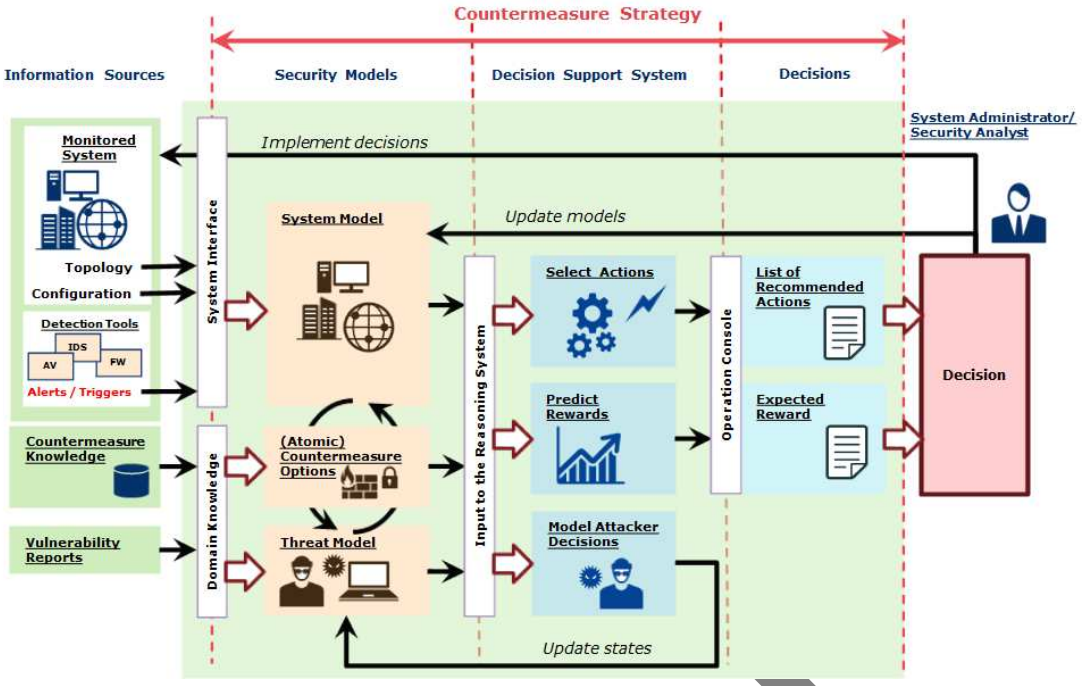


Fig. 2: An abstract view of countermeasure strategy architecture.

uration, a model which synthesizes these pieces of data is createdto report the core parameters which will be used for further analysis. This model uses *architecture description languages*, like ArchiMate [51], to conceptually model the structure, behavior, and components of the system [52].

- *Atomic countermeasure options*: given the raw knowledge about the remediation steps, represented in the countermeasure knowledge, a list of possible countermeasures is created, trying to combine the above-mentioned remediation steps to effectively defeat possible attacks.
- *Threat model*: based on the vulnerability reports and system model, a threat model is created, which represents possible attack patterns. As detailed in Section III-B1, AGs and ATs are prominent examples for this model, as they are usually employed as a formal representation which aims to describe possible attacker's actions against the monitored system [11].
- *Select actions*: in this component, the selection of the specific countermeasures is conducted, balancing the existing trade-off between the security level of the system and the cost of the reaction. Specifically, this component considers not only the monetary cost of activating the selected actions but also the possible negative impact due to their enforcement, including the possible availability decrease for one or more services. Therefore, for a specific attack, a set of possible countermeasures are analyzed and ranked, trying to maximize their effectiveness and to minimize their cost.
- *Predict rewards*: depending on the selections done by the previous component, a calculation of the produced security and economic benefit is performed and forwarded to the system administrator.
- *Model attacker decisions*: based on the choice of the "Select Actions" component, the threat model is updated, so as to be able to predict similar kind of attacks. When a countermeasure is selected, the attack patterns are modified reflecting this choice.
- *List of recommended actions/expected reward*: based on the selection made by the decision support system, a list of countermeasures is prepared to be presented to the system administrator. This report must include the expected revenue derived from the implementation of the selected countermeasures, such as an improvement of the overall risk level of the system or an economic reward due to the attack blocking.
- *System operator decision*: by getting the previous list and associated data, the administrator may decide to implement the reaction procedure on the monitored system, and update the configuration on the previous models (i.e., system and threat models).

It can be safely argued that the presence of the aforementioned components in the above abstract strategy is expected to result in more accurate and efficient countermeasures. This is basically because the different components can offer a modularized but holistic view of the problem and provide the correct kind and amount of information for aiding the administrator to decide optimally on an appropriate reaction plan. In this vision of the countermeasure strategy, the role of security administrators is of major importance. This view relies on the fact that the administrator not only has a deep knowledge of the particular system but also the responsibility and the privileges to take an important decision. Moreover, the administrators usually manage system hardening procedures with budget limitations, hence they need to balance the trade-off between impact and cost of the countermeasure's

DRAFT

enforcement and system security level. This procedure is twofold; on the one hand, it represents the actions which will be implemented on the system. On the other, it represents an important feedback for the decision support system. This learning process for the decision support system is crucial, since this component has to provide a decision on which action must be implemented on the system maximizing the reward.

B. Features for comparison of works

So far, numerous works have been proposed in the literature dealing with a countermeasure strategy [53], [54]. Most of them share a common ground but also exhibit significantly different features in terms of the adopted system representation and countermeasure derivation methodologies. To this end, after analyzing the various works, we have identified the following 7 features that can be used as a basis for our analysis. Note however that the scope of countermeasures selection is moderately wide. Hence, it is unlikely to find solutions that simultaneously address all the posed challenges of the field, while the process of extracting the necessary information to enable a foolproof comparison proved to be demanding. The aforementioned challenges led us to conclude to the following comparison features:

- Attack modeling
- Countermeasures provision techniques
- Outcomes assessment
- Type of reaction
- Used standards
- Automation level
- Performance

It has to be mention that we treat the aforementioned features as having an equivalent importance weight. That is, since each work may have different characteristics and it may have been evaluated in a specific environment, we preferred not to assign diverse (unequal) importance weights to the identified features, as this could bias unjustifiably the provided analysis. To this end, Section IV-N offers a comprehensive comparison of the various works analyzed as part of this survey based on these equivalent features.

1) Attack modeling: An attack model is a formal representation aiming to describe possible attacker's actions focusing on the vulnerabilities and misconfigurations which are present within systems [55]. This knowledge about the possible attack steps is essential to counteract malefactors. That is, via these steps it is possible to block the most probable paths that an attacker can follow, eradicating in this way the intrusion. Next, we describe the most common modeling techniques, highlighting on their advantages and drawbacks.

Attack Graph (AG): This technique is widely used in the literature for modeling cyber attacks [56]. Its popularity is due to its ability to synthesize several system-related aspects to construct a complete representation of the infrastructure intended to be protected. AGs combine information about the network topology, possible vulnerabilities, and exploits appearing on the assets of the network, aiming to provide a visual representation of attack paths that an attacker could traverse in an effort to reach specific network targets. In other words, an AG visualizes the vulnerability dependencies in a network and enumerates its possible states. The states of the network are represented as nodes in the graph, while the edges represent the interconnections among them. The edges reveal a *cause-consequence* relationship between the nodes of the graph. AGs enable the defender to identify the weaknesses of a network and/or pinpoint risky paths on the graph, so that to proceed to the necessary actions removing or remediating nodes and/or edges that threaten the network assets. Fig. 3 depicts an AG describing a possible File Transfer Protocol (FTP) buffer overflow attack carried on asset 1.

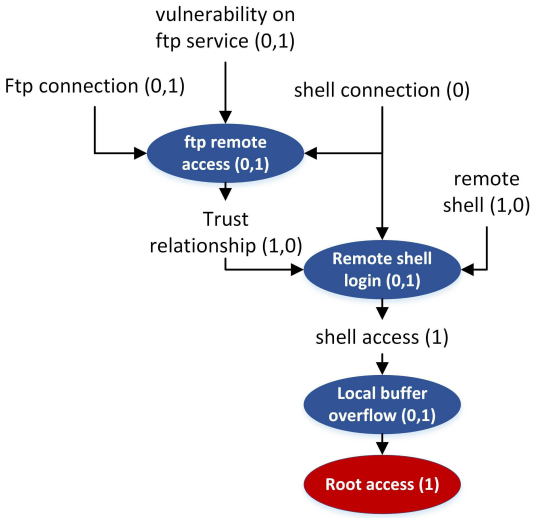


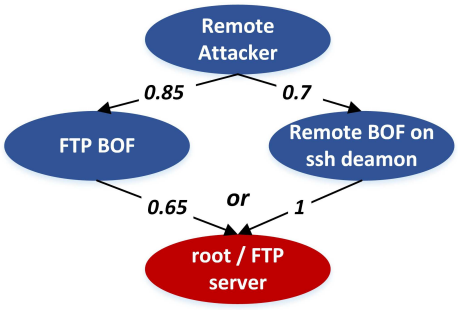
Fig. 3: Attack graph showing ftp buffer overflow attack executed by asset 0 on asset 1

This technique poses limitations as well. Discovering all the dependencies in a network cannot be considered an easy task, thus inefficiencies may appear. Moreover, a known limitation is the scalability of the graphs [56]. As the size of the network increases, an AG becomes bulky since the dependencies among the nodes are numerous. As a result, the defender faces difficulties in understanding and analyzing the graph. Several tools are available to create AGs such as NetSPA [57] and MulVAL [58].

Bayesian Attack Graph (BAG) [59] can be considered as an extension to legacy AGs. Specifically, this type of AGs introduces probabilities on the edges for modeling the uncertainty in the state transitions. In the example depicted in Fig. 4, the vulnerabilities discovered in a system are marked with likelihood values on the edges, representing the overall probability for an external attacker to exploit them. Then, the score for the final goal is computed considering the possible combinations of the previous conditions, which in the example are presented in disjunction. In this way, BAGs are able to take into account the likelihood of exploitation for a certain vulnerability with finer granularity. On the downside, this technique inherits the limitations of AGs, adding also the computation and assignment of the probabilities as extra parameters, which in turn augments the overall complexity.

Attack Tree (AT): ATs introduced in [11] aim to formally represent the security states of a system and to visualize

DRAFT



ssh	ftp	Pr (root)	Pr (¬root)
1	1	1	0
1	0	0.65	0.35
0	1	1	0
0	0	0	0

Fig. 4: Bayesian attack graph showing ftp and ssh buffer overflow attacks with an OR condition on the target node

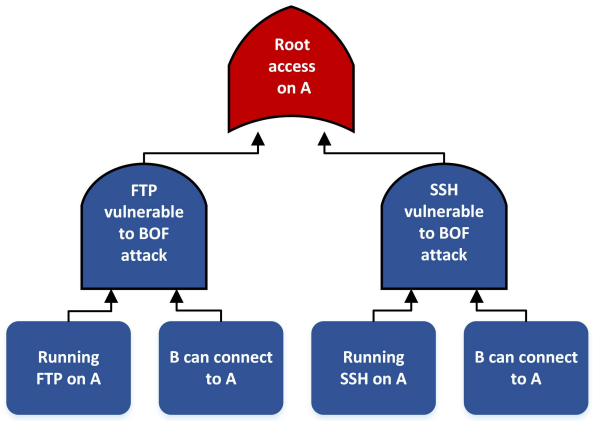


Fig. 5: Attack tree showing ftp and ssh buffer overflow attacks with an OR condition on the root node

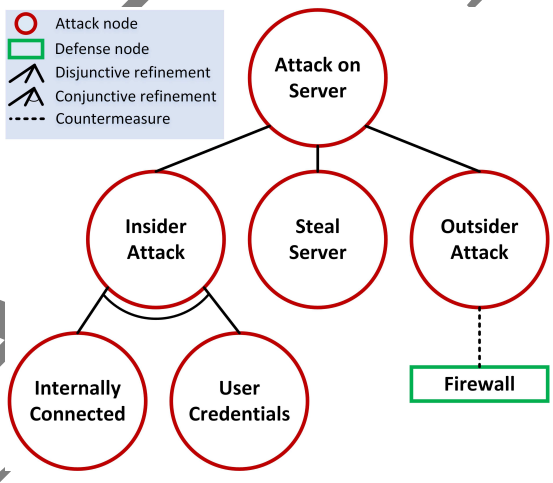


Fig. 6: Attack Defense Tree

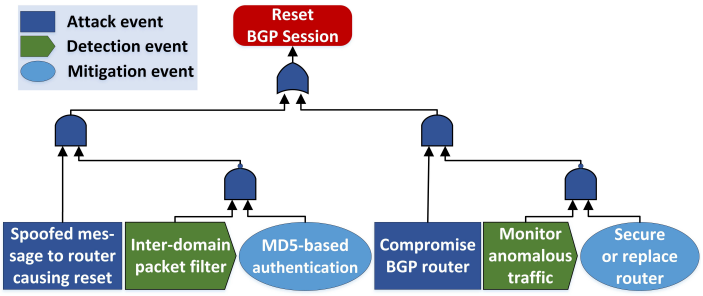
the different ways in which it can be attacked. The root of the tree represents the attacker's ultimate goal, while its leafs correspond to the entry points for the attacker. The intermediate nodes are the attacker's sub-goals which are connected with AND/OR conditions. These conditions create multiple paths that connect the leafs of the tree with the root. An example of AT is shown in Fig. 5, where SSH and FTP buffer overflow attacks are connected with an OR condition on the root node. In the AT representation, the nodes usually include also values (either continuous or nominal) to describe the attack paths in a more detailed way. The attack likelihood, the financial cost of exploitation, time, and the cost of defense resources are just some examples of node values. Using these values and starting from a leaf node, the defender is able to sum up the total defense cost up to the root. As with the previous technique, an AT can become large and complex containing many nodes and numerous paths between the root and the leaves. Therefore, in large and complex topologies, where the interconnections of the network assets and the possible vulnerabilities are numerous, the increased number of security states could result to ATs with hundreds or thousands of different paths. As a result, the addition of only one security state (i.e. a new node) results to several new interconnected paths, leading to state explosion in the search problem. In addition, due to the fact that the root can represent only one ultimate goal, it is reasonable to say that several trees may be needed in order to create an holistic security overview of an infrastructure. This can result in a forest of ATs.

Among others, extensions of ATs are *Attack Countermeasure Tree* (*ACT*) [60], *Attack Response Tree* (*ART*) [61], and *Attack Defense Tree* (*ADT*) [62]. These techniques enable particular features keeping the same tree representation. In particular, the ACT formalism uses a non-state-space approach to represent attack, detection, and mitigation events on the same tree structure. These concepts are depicted in Fig. 7a, where the possible steps of an attacker to reset a BGP session are represented using ACT formalism. ARTs use a state-space model (partially observable stochastic game model [63]) to find the optimal set of countermeasure, including both attacks and responses on the same tree. In this regard, Fig. 7b shows an example of ART to visualize a possible attack against a web service. Lastly, ADT is a node-labeled rooted tree describing the actions an attacker can take to attack a system and the defenses that a defender can employ to protect it. Fig. 6 illustrates an instance of ADT for a possible attack against a server. Note that the node labeled as "Firewall" represents a defense action against the attack node "Outsider Attack".

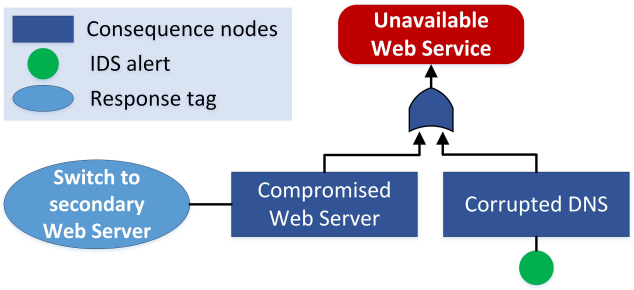
Service Dependency Graph (SDG): These are dependency graphs which represent the relationships among multiple services in a system [64]. To exemplify this situation, Fig. 8 shows the interdepencies for a web mail service. The relationships are defined as privileges, which have been granted to the dependent service from the antecedent one. The dependencies can reveal how a dependent service can be affected in terms of confidentiality, integrity, and availability, if a related service faces an intrusive incident [65]. However, identifying and representing the interdependences of all the services in an infrastructure can be proved a cumbersome task, which in turn can lead to inefficiencies.

Markov Decision Process (MDP): This technique provides a mathematical methodology to model decision making in situations where outcomes are not totally under the control

DRAFT



(a) Attack countermeasure tree



(b) Attack response tree

Fig. 7: (a) ACT illustrating an attack against a BGP router; (b) ART showing an attack against a Web Server

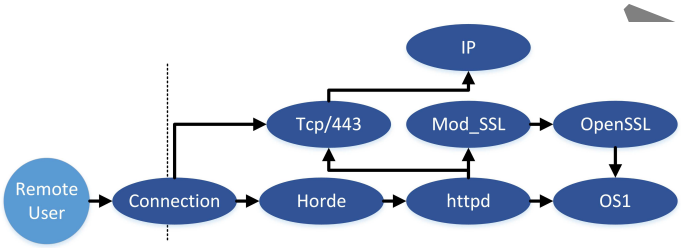


Fig. 8: Service dependency graph for a mail web service

of the decision maker, since some variables in the process are stochastic [66]. In particular, one can realize a MDP as a *discrete time stochastic control process*, where at each time step the decision maker can cherry-pick between a set of actions for a specific state. A reward is associated with each of these actions. In this way, the probability of moving to a new state is influenced by the previous state and by the selected action. In this respect, MDPs are an extension of the Markov chains [67] by adding to them the concepts of action and reward. In Fig. 9 an example of MDP is depicted, in which a system is represented using two states and two possible actions, together with the associated rewards. However, identifying all the possible states and actions in a system represents a difficult task mainly due to the complexity of the modern networks.

The Markov models are often used alongside the other three attack modeling techniques in order to statistically assign probabilities to the paths on the graph. With this probabilistic analysis, the defender can become aware of the most probable paths that an attacker could follow [68].

Among others, extensions of MDP are *Competitive Markov Decision Process* (*CMDP*) [69] and *Partially Observable*

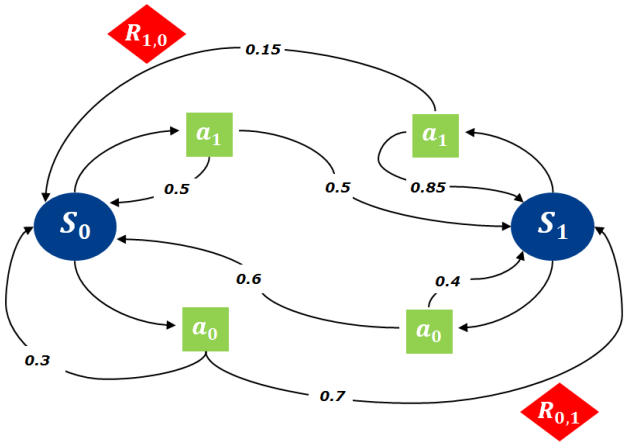


Fig. 9: Markov decision process with two states, two possible actions and associated rewards

Markov Decision Process (*POMDP*) [70], in which Markov processes are adapted to represent different situations. More specifically, CMDP is used to model the system states as a stochastic game where the competitors are the adversary and the defender who both aim to increase their profit. POMDP on the other hand is applied when the states of the system are not entirely observable.

For an easy reference, TABLE I summarizes all the analyzed attack modeling techniques, highlighting on their main advantages and drawbacks. Both main attack models and their extensions have been summarized. The reader should keep in mind that the extended representations (i.e., those marked with ▷) inherit the characteristics of the main model or try to address the limitations found in the main representations.

2) Countermeasure provision techniques: The ultimate goal of the countermeasure strategies is to come up with the optimal countermeasure or a set of them depending on the events occurring in the system and its current security state. To this end, a methodology needs to take into consideration several metrics and balance the trade-off among them in order to conclude to the optimal solution that achieves the desired level of asset protection with respect to the protection cost.

A metric can be defined as "the assignment of a value to the characteristics of an object or event, which, in this manner, can be compared with other objects or events" [72]. Metrics such as the attack cost, defense implementation cost, attack impact, operational cost, and others are only some examples of factors that can be used for defining the optimal solution [73].

For finding the optimal solution, some methodologies consider also statistical-based optimization techniques for optimizing and calibrating metric functions. In this context, both *single objective optimization problem* (SOOP) and *multiple objective optimization problem* (MOOP) [74] have been proposed to support the decision making in identifying the optimal countermeasures.

In this direction, until now, several methodologies have been recruited. Evolutionary computing techniques, i.e., *Genetic Algorithms* (GAs) [75], have been used in constructing solutions using in their fitness functions synthesized metrics for tuning

DRAFT

TABLE I: Overview of the presented attack modeling techniques with a focus on strengths and weaknesses

Attack Representation	Strengths	Weaknesses
Attack Graph *[56]*	✓ Holistic view of the system ✓ Visual representation of possible attack paths	✗ State explosion for complex network ✗ Probabilities and defense points are not represented
▷ *Bayesian Attack Graph [59]*	✓ Likelihood on the edges to model uncertainties	✗ Extra computation for the probabilities calculation and assignment
Attack Tree *[11]*	✓ Formal representation of the system states ✓ Visual illustration of possible attacker paths using AND/OR conditions ✓ Absence of defense nodes	✗ Numerous paths between leaves and root ✗ Forest of trees to protect a complex system
▷ *Attack Countermeasure Tree [60]*	✓ Attack, detection, and mitigation events on the same tree structure	✗ Countermeasure nodes cannot be refined over time
▷ *Attack Response Tree [61]*	✓ State-space model, including attacks and responses on the tree	✗ State explosion due to the use of POMDP
▷ *Attack Defense Tree [71]*	✓ Improvements of the tree structure though defense point and countermeasure representation	✗ Detection and mitigation points are represented in an unique node
Service Dependancy Graph *[64]*	✓ Visual representation of interdependencies between services ✓ Quantitative attack impact assessment using CIA attributes	✗ Identification of service dependencies requires huge effort from a security expert ✗ Integration with an attacker-centric representation is needed to model possible attack decisions
Markov Decision Process *[66]*	✓ Representation of decision making process ✓ Concepts of state, reward, and action	✗ State explosion for complex systems ✗ Often used alongside with other attack modeling techniques
▷ *Competitive Markov Decision Process [69]*	✓ System states modeled as a stochastic game between attacker and defender	✗ Computation of the attacker and defender steps augments the problem complexity
▷ *Partially Observable Markov Decision Process [70]*	✓ Representation of unobservable system states	✗ Interaction with the environment to receive information on unobservable states increases the complexity

the solution. Also, *Arm Race* is a bio-inspired technique which describes the competition between two different populations (attacker and defender), which compete with each other to defeat the opponent [76]. The evolution of the competitive populations progresses in parallel. The fitness of the individuals in the one population competes against the fitness of the individuals evolved in the other. Similar proposals utilize *Ant Colony Optimization* (ACO) [77] with graphs with the aim of defining the optimal set of countermeasures based on the pheromone paths constructed by the ants. Precisely, the ants roam probabilistically on the graph based on the probabilities indicated by two parameters, namely the attractiveness and trail level of the move. Note that the latter parameter also incorporates the cost of the trace. *Tabu Search* [78] has been also used in this field; this technique tries to find an optimal solution avoiding sticking in local sub-optimal regions.

Generally, when facing an MDP, the *Bellman's optimization method* is usually applied in order to solve it with dynamic programming [79]. Bellman equation solves discrete-time problems regarding the optimal control theory. This is done by optimizing iteratively the objective function and keeping track of the changes. However, other researchers prefer to use *heuristic* implementations for countermeasure selection's algorithms which best fit to their needs.

3) Outcomes assessment: In the context of this work, we consider the evaluation of the surveyed systems with regards to their outcome, that is, providing countermeasures, as a critical feature in our analysis. To assess the results produced by the analyzed works, we extract two commonly used characteristics regarding the outcome assessment, namely *testbed* and *admin's role*.

Testbed: In the context of the reaction strategies, the testing environment plays a significant role, because it directly refers to the applicability of the proposed solution. In this survey, we refer to the used testbed, using the terms *simulated, emulated, and real*.

Admin's role: As already stated, the role of the system administrator is central to our vision of automated reaction strategy. Specifically, based on the surveyed works, we identify two distinct roles for the system administrator:

- *Tuning* - The administrator is tasked with setting the goals, objectives and metrics, which are subsequently used by the countermeasure system.
- *Feedback* - The administrator assesses the outcomes of the countermeasure system, selects the optimal solution, and provides feedback to it.

According to the literature, both the aforementioned tasks can be benefited from the use of reinforcement learning [80].

4) Type of reaction: Reaction to security incidents can be achieved by following two main approaches, namely *static* and *dynamic*. As static approaches we perceive those which are capable of acting proactively against security incidents,

DRAFT

TABLE II: Standardization attempts for security automation

Category	Acronym	Name	Description
Vulnerability management	CVE [50]	*Common Vulnerabilities and Exposures*	Provides a reference method for publicly known vulnerabilities and exposures. It is available in several format, such as CVRF, XML and HTML
	CVRF [81]	*Common Vulnerability Report Format*	XML-based language that enables different stakeholders across different organizations to share critical security-related information in a single format
	NVD [82]	*National Vulnerability Database*	U.S. government repository of standards based vulnerability management data represented using SCAP. It can be accessed using JSON, XML or RSS feeds
	OVAL [83]	*Open Vulnerability and Assessment Language*	XML-based Information security community effort to standardize how to assess and report upon the machine state of a computer system
Configuration management	CCE [84]	*Common Configuration Enumeration*	Provides unique identifiers to system configuration issues for facilitating fast and accurate correlation of configuration data across multiple info sources. It is available in XML and Excel format
	CCSS [85]	*Common Configuration Scoring System*	Set of measures of the severity of the software security configuration issues
Asset management	CPE [86]	*Common Platform Enumeration*	Standardized XML-based method for describing and identifying class of application, operating systems, and hardware devices present in enterprise's computing assets
	ASR [87]	*Asset Summary Report*	XSD data model to express the transport format of summary information about one or more set of assets
Software assurance	CWE [88]	*Common Weakness Enumeration*	Provides a common language to discuss, find and deal with the causes of software security vulnerabilities as they are found in code, design or system architecture. It is available in several format, including CSV, XML and HTML
	CWSS [89]	*Common Weakness Scoring System*	Provides a mechanism for prioritizing software weakness in a consistent, flexible and collaborative manner
	CMSS [90]	*Common Misuse Scoring System*	Set of measures of the severity of software feature misuse (trust assumptions made when designing software features abused to violate security)
	CWRAF [91]	*Common Weakness Risk Analysis Framework*	Part of the Common Weakness Enumeration (CWE) project. It provides a graphical framework for scoring software weaknesses
Attack taxonomy	CAPEC [92]	*Common Attack Pattern Enumeration and Classification*	Offers a publicly available catalog of common attack patterns classified in an intuitive manner. It can be acquired in XML and CSV format
Remediation information	CRE [93]	*Common Remediation Enumeration*	Suite of XML-based remediation specifications that enables automation and enhanced correlation of remediation activities
	ERI [94]	*Extended Remediation Information*	XML dictionary with additional data about each CRE, including references to CPE, CVE, and CCE
Intrusion detection	IDMEF [95]	*Intrusion Detection Message Exchange Format*	Using XML schema, it defines data formats and exchange procedures for sharing information of interest to IDS/IPS and to the management systems that may need to interact with them
Cyber threat information sharing and analysis	TMSAD [96]	*Trust Model for Security Automation Data*	Common trust model that can be applied to XML specification within security automation domain
	OpenIOC [97]	*Open Indicator Of Compromise*	An extensible XML schema that allows the description of the technical characteristics that identify a known threat, an attacker's methodology, or other evidence of compromisation
	STIX [98]	*Structured Threat Information eXpression*	Collaborative community-driven effort to define and develop a language to represent structured threat information. It is based on XML schemes
	TAXII [99]	*Trusted Automated eXchange of Indicator Information*	Open transport mechanism that standardizes the automated exchange of cyber threat information
	CybOX [100]	*Cyber Observable eXpression*	Standardized XML-based language for encoding and communicating high-fidelity information about cyber observables, that are noticeable events or properties in the operational cyber realm
Security benchmark	XCCDF [101]	*eXtensible Configuration Checklist Description Format*	XML-based specification language to write security checklists, benchmarks and related documents
Incident management	IODEF [102]	*Incident Object Description Exchange Format*	Defines a data representation that provides a framework for sharing information commonly exchanged by Computer Security Incident Response Teams (CSIRTs) about computer security incidents
Malware management	MAEC [103]	*Malware Attribute Enumeration and Characterization*	Standardized language for encoding and communicating high-fidelity information about malware based upon attributes such as behaviors, artifacts, and attack patterns. It is available in XSD and HTML format

while as dynamic those which are able to react upon the occurrence of cyber threats. These methods differ in the intent, the effort required to implement them, and their outcome from a defender's viewpoint.

Static reaction: It deals mainly with *security risk assessment* [26], which represents the process to identify potential security risks that may reside in an ICT system. It begins with the identification of the system characteristics, including weaknesses and exposures, and potential threat sources. The accurate estimation of the amount of risk per asset in the system is a effortful task, and often this judgment is driven by the administrator's subjective belief. This estimation could be useful to locate weak spots during the system design phase, but also proactively, via the use of penetration testing tools [104]. The main limitation of this approach is the lack of a dynamic model capable to follow the flow of events of a potential attacker who could try to exploit system's vulnerabilities. In such a scenario, countermeasures must be taken on-the-fly, in order to limit if not eradicate the intrusion efficiently.

Dynamic reaction: It is concerned with the system response capabilities to a possible ongoing attack. In this case, a deep knowledge of the system vulnerabilities, together with the evaluation of attacker's skills and response time are pivotal factors. Due to its features, this approach covers the limitation of the static one, making more effective the countermeasure selection. On the downside, it requires more computational power. Considering all the parameters which must be taken into account, the task to identify the optimal set of actions for dynamically blocking an advancing intrusion is hard. The probability of the attack paths, the countermeasure's effectiveness as well as their cost are just few examples of the parameters whose computation has to be done in real-time.

DRAFT

5) Use of standards: For conducting a quantitative analysis of the adopted reaction strategy, one needs to use *security metrics*, which can measure network security in an objective and cross-platform manner. In computer and network security, a plethora of metrics has been proposed, with the goal of capturing different aspects, including attacks, intruders, network topologies, costs, and vulnerabilities metrics [73].

Nevertheless, to be effective, a metric should belong to a highly shared and used measurement system. The various *standards* help in addressing this problem. The use of standards allows the comparison among published works and solutions, giving quantitative and qualitative measurement of their effectiveness. In the context of this survey, we concentrate on the standards reported in TABLE II.

CVSS: The *Common Vulnerability Scoring System* (CVSS) is an open framework for communicating the characteristics and severity of software vulnerabilities [105]. In the recent years, CVSS has become the *de facto* standard, adopted from the research community to measure the effectiveness of the proposed works dealing with vulnerabilities' impacts and scores [106], [107].

Moreover, great effort has been put to standardize the information flow regarding security threats intending to help enterprises worldwide to use common means of fighting against them. Many of the cited standardization attempts in TABLE II are part of the *Security Content Automation Protocol* (*SCAP*) [108], which is maintened by NIST, and MITRE corporation [109]. This source database duality sometimes results in redundancies in the standards. For instance, regarding the "cyber threat information sharing and analysis" category, both TMSAD [96] and TAXII [99] aim to create a trust model to exchange information in the field of security automation.

6) Automation level: As with the previous features, especially for large network topologies, the great amount of parameters to consider in the process of finding the optimal set of countermeasures often becomes unmanageable. In addition, the response time is a crucial factor, therefore it can be said that a certain level of automation is required in the deployment of a reaction strategy. Until now, several level of automation (LOA) taxonomies have been presented, dealing with human/machine interaction [110]. In the context of this work, we classify them into three levels:

- *Manual* - The administrator performs the tasks concerning the reaction strategy, including monitoring the state of the system, selecting the appropriate action to perform and physically implementing it.
- *Semi-Automated* - The system generates a ranked list of decision options based on specific criteria. The system administrator can select one of them, thus giving a feedback used to calibrate the future system decisions.
- *Fully Automated* - The system selects the best option to implement and directly enforces it based on predefined directives given by the administrator.

As observed, the concept of automation is strictly connected to the role of the system administrator. That is, their participation in the countermeasure strategy is a core parameter, not only due to their expertise and privileges, which could lead to a better solution, but also because they have to deal with a specific economic budget.

7) Performance: Few will oppose that performance is of great importance in any countermeasure strategy.. This means that quality metrics [111] must be considered when a big project has to be developed. In others words, the reaction plan needs to take into account some inherent limitations of the existing devices in the system. For example, in an *Internet of Things* (IoT) network topology, the various devices are generally resource-constrained [112].

From the analysis of the surveyed works, we concluded that the performance factors which strongly influence the implementation of the countermeasure strategy are *scalability, time complexity,* and *response time*.

Scalability: It is one of the most desirable attributes of a network, system or process. It refers to the ability of a system to accommodate an increasing number of elements or nodes, to process growing volumes of work gracefully, and to be susceptible to enlargement [113]. In Section IV, we refer to the scalability of the analyzed works as the property of the proposed solution to preserve a polynomial behavior on the response when the number of nodes in the system increases. We classify this factor using the linguistic values in the following set: $\{Low, Medium, High\}$.

Time complexity: It quantifies the amount of time taken by an algorithm to run as a function of the length of the string representing the input [114]. In Section IV, we refer to this factor as the property of the countermeasure provision solution to preserve a polynomial behavior when the input (cardinality of the countermeasure set, number of nodes, etc.) increases. This factor is estimated using the same scale of three values as that in Scalability.

Response time: This third factor can be defined as the elapsed time that a computer system takes to respond to a given input. In Section IV, we define this factor as the ability of the examined solutions to provide a reaction in an acceptable time frame in the context of dynamic reaction. To estimate response time we rely on the following scale: $\{Fast, Average, Slow\}$.

IV. Survey of works

This Section contains a detailed survey of the current major works on the cyber attack countermeasure strategies ecosystem. The survey is given in chronological order with oldest proposals first. For each work, we first provide a succinct description, followed by a constructive analysis of its features.

A. Dewri et al. [115]

Descriprion: The authors in [115] introduce an evolutionary technique for defining an optimal defense strategy. They conduct a cost-benefit analysis to determine the optimal defense strategies that have to be taken against dynamically changing attack endeavors by maximizing the *Return-On-Investment* (ROI) index. ATs are chosen as the modeling technique to describe the dependencies among the security states of the protected system. The AT induction methodology used here was firstly introduced in [49]. The authors developed their own solution in which the tree induction process considers

DRAFT

as input an initial vulnerability table, the network topology, and online vulnerability exposure databases, namely BugTraq [116], CERT/CC [117], and NetCat [118].

Specifically, the first is an electronic mailing list dedicated to issues about computer security such as vulnerabilities, vendor security-related announcements, methods of exploitation, and possible remedies [116]. The second is the coordination center of the computer emergency response team (CERT) for the Software Engineering Institute (SEI), which researches software bugs that impact software and internet security [117]. The latter is a computer networking utility for reading from and writing to network connections using TCP or UDP. It is also used as a vulnerability scanner during penetration testing [118].

The evolutionary process is based on the non-dominated sorting GA NSGA-II [119]. This algorithm is suitable for applying multi-objective optimization, while sustaining the diversity of the solutions to a high level. The defense and attack strategies are modeled as binary vectors that represent the leafs of the AT. The binary values for a defense strategy signify whether a defense measure is enabled on a leaf (value = 1) or the leaf is unprotected (value = 0). From the adversary's point of view, if a leaf is chosen in an attack strategy, then the corresponding value is 1, otherwise 0. The GA generates defense and attack strategies, while fitness functions are used to infer on the superior strategies.

To conduct a quantitative assessment of the problem, the authors adopt the Butler's multi-attribute risk assessment framework [120], [121]. They introduce several complex metrics which are based on different types of individual metrics like the installation and operation cost of a defense (monetary metrics), system downtime (time), law penalty (severity), and others. In total, authors define the following metrics:

- Potential Damage of an attribute (P)
- Residual Damage (RD)
- Total Security Control Cost (SCC)
- Damage Inflicted (DI)
- Attack Strategy Cost (ASC)
- Breach Loss (BL)

Using P on an attribute of the tree, an augmented AT is built. More specifically, this corresponds to a normal AT which also bares the aggregated cost information to every attribute.

The authors initially define and then evaluate their solution with reference to the following 4 problems: SOOP, MOOP, Multi-objective Robust Optimization problem, and Attacker-Defender Arms Race problem. For the first problem, they define a weighted function that minimizes RD and SCC metrics based on the selection of a boolean vector of possible security controls. For the second problem, the multi-objective strategy aims to minimize several metrics individually, thus the computed solution is more reliable considering the correlation of several parameters. For the third problem, the authors conduct a fault tolerance analysis, where they try to define the most robust defense solution, that is, a solution that is less sensitive to failures in security controls. This is achieved by calculating RD and considering potential failures in these controls. The latter problem differs significantly from the previous ones. Namely, the authors utilize a competitive co-evolution technique to emulate an "arms race" between the attacker and the defender. As already pointed out in the previous section, this methodology incorporates the evolution of two competitive populations in parallel. The fitness of the individuals in the one population competes against the fitness of the individuals evolved in the other. This means that the success of a defense strategy implies the defeat of the attacker and vice versa. The fitness functions are correspondingly based on the Payoff Functions for the Defender (POD) and the attacker (POA). These functions incorporate combinations of the above mentioned metrics introduced by the authors to reflect the benefit of the two competitors. The optimization goal for a defender is to find a defense strategy that maximizes POD against all possible attacks, whereas the attacker aims to maximize POA.

The experimental results for the arms race approach revealed that it was easier for the attacker to find strategies to bypass the measures of the defender. At the same time, a low improvement in the attacker's payoff resulted into a significant drop in the defender's payoff. However, it seems that the system finds an equilibrium point after several generations.

Analysis: The authors adopt the AT representation and they also integrate the potential damage metric to build an augmented AT. While this methodology is able to hierarchically represent possible sub-goals of the attacker and enable a quantitative approach of the problem, it remains rather complex. Even worse, the complexity of the proposed solution is increased further as the competitive co-evolutionary process demands the evolution of two antagonistic populations. However, the notion of the "arms race" introduced by the authors reflects the relationship between the attacker and defender and emulates the dynamic engagement between them. Defining an equilibrium point is an appropriate technique to achieve the goal of adequately protecting an asset without over-protecting it. On the downside, the approach of single and multi-objective optimization seems incapable of providing the optimal hardening solution. This is mainly due to the difficulty of tuning the weighted fitness function and the extreme values of RD metric produced, which in turn overprotect the assets. The robust optimization approach employed by the authors is an interesting notion, while the fault tolerant scheme they use approaches the problem realistically as it cannot be taken for granted that a defense is unbreakable.

The quantitative evaluation of the proposed scheme is based on a framework proposed in [120] and [121], while the metrics introduced by the authors consider monetary, severity, and time values. This handling is suitable for creating a holistic quantitative evaluation for a scheme. Even so, the metrics employed are not aligned with globally accepted standards, as showed in TABLE II, to enable a quantitative comparison with other solutions.

A limitation of the proposed methodology is that the attacker's expertise level and the dynamic nature of the network are not taken into consideration. This however may have a severe impact on the implementation of any similar system. Under realistic conditions, a decision maker should adapt their action by considering also the abilities of the adversary,

DRAFT

while an accurate representation of the network is vital. The authors' proposal is not destined to deliver an automated reaction system but a static tool to conduct risk assessment for identifying the exploitation possibilities through the evolutionary approach. Besides, in the experimental phase, the authors consider 19 defenses, whereas the AT has 13 leafs. This is translated into 2^{19} different defense strategies and 2^{13} attacks. It can be inferred that even for a middle-sized network and a considerable number of countermeasures the search space of the problem is being expanded exponentially. It is noteworthy that the countermeasures proposed are limited to disabling and patching actions as this is the case for the majority of the documented works. Finally, the competitive "arms race" approach reflects the relationship between the attacker and the defender, yet defining the equilibrium point could be a really effortful task if the model is to be extended to fulfill the requirements of a dynamic response framework.

B. Poolsappasit et al. [59]

Description: In this work, the authors propose a security risk assessment methodology based on BAGs. These can be considered as an extension of AGs as presented in Section III-B1, adding also a likelihood to the occurring events, which in turn can modify their state. That is, the main difference of a BAG compared to a classical AG is the existence of a non-zero probability corresponding to the case that the attacker is not able to take advantage of the exploit, even if all the preconditions are met. This probability per network node is represented by a *Local Conditional Probability Distribution* (*LCPD*).

In order to compute these probability distributions, the authors use CVSS [105] to estimate the attack likelihood. Specifically, they calculate this probability using three base metrics, namely: *access vector, access complexity*, and *authentication instances*. Using the proposed technique, they are able to perform:

- *Static Risk Assessment* - It identifies system characteristics, potential threat sources and attacker capabilities, often judged by the system administrator with the aim of creating the initial probability values.
- *Dynamic Risk Assessment* - It uses the knowledge about the happened attack incident to update the probabilities using the bayesian inference technique.
- *Risk Mitigation Analysis* - It is aimed at countering risks either or both in a proactive or reactive manner.

Regarding the last point, the authors define a security control as a *preventive* measure that minimizes or eliminates the likelihood of an attack; the enforcement of this control modifies the LCPD of a node, and indirectly influences the unconditional probabilities of other attributes in the network. Therefore, a *security mitigation plan* is intended as a set of security controls, where each of them has a specific cost.

By merging the concepts of LCPD and mitigation plan, the authors define the *Augmented-BAG*, which incorporates the security controls with the expected loss/gain per node. From a system administrator point of view, the derived cost model represents a hard problem to solve. Toward this direction, the authors present *SOOP* and *MOOP*, with GA as means of resolution [119].

The authors include an evaluation section to assess these choices, showing the feasibility of the model in a dynamic context as well. In the conducted experiments, the calculated mitigation plans for the static and dynamic environment present many similarities, suggesting the application of similar sequence of security controls.

Analysis: The authors use BAGs to model attack's probabilities into the system. By doing so, they are able to consider also the intrinsic *uncertainty* of a real attack incident. Even though this representation can provide a more detailed and realistic view of the problem, the likelihood assignment and computation are extra parameters that augment the complexity of the AG. Particularly, the algorithm of generation for the BAGs cannot go beyond $O((A \times M)^3)$, where A represents the number of attributes to consider, and M represents the number of machines in the system. Even if the initial AG generation is a one-time cost, for a large-scale and power-constraint network this complexity may be unsustainable. Furthermore, the computation of the marginal probabilities on the BAG is even more challenging; both for prior and posterior probability, when thinking of a dynamic scenario, the algorithmic complexity is exponential. It is therefore obvious that this complexity should be decreased by, say, using a heuristic-based algorithm to realize a dynamic reaction in an acceptable time.

The authors also demonstrate the methodology of providing countermeasures using the cost model as a core parameter. Specifically, the problem is focused on the *system administrator's dilemma* [49], which states that the administrator often has to deal with a limited budget that could preclude them to implement all the possible hardening measures. Thus, they demonstrate the need of optimization techniques to provide a priority ranking of the countermeasures. They claim that by using a greedy selection or the subjective belief of an expert, the results may be inaccurate. So, to solve this problem, they propose the use of GAs and demonstrate that this methodology presents many advantages both for SOOP and MOOP, which do succeed in achieving a more precise classification of the security controls. One can safely argue that this procedure is extensible for a bigger set of hardening measures, making the process more realistic for a real network.

The proposed reaction strategy by the authors is mainly static. For a given network, they show how it is possible to plan a mitigation strategy for the security incidents that decreases the overall risk level, taking into account the cost model. They also test their framework in a dynamic environment, and via the use of two different attack scenarios, they compute the corresponding reaction plan. It can be said that further experimentation is needed in this direction, considering as a core parameter also the *time of the reaction*, that cannot be easily considered using BAGs.

C. Roy et al. [122]

Description: The work in [122] aims to provide a cost-benefit countermeasure system for dealing with cyber attacks.

DRAFT

By capitalizing on their previous work in [60], authors rely on ACTs with the purpose of providing a scalable solution for the problem at hand. According to the authors, ACTs perform better than ARTs which use POMDP as a solution technique. This is because the latter model leads to state-explosion issues. More specifically, ACTs provide a non-state-space approach, and according to the authors, this technique is less expensive than state-space driven approaches.

In their model, ACTs are able to represent three types of events as internal nodes in the structure, namely, *atomic attack*, *detection*, and *mitigation*. The authors build two example trees to conduct their experiments and to evaluate and compare the performance of their model against the ART-based solution proposed in [40]. They also suggest two algorithmic approaches to define the optimal set of countermeasures, namely *explicit enumeration* (greedy approach) and *branch and bound*, and they introduce the following four objective functions:

- Selecting the minimum number of countermeasures.
- Selecting countermeasures by minimizing the Security investment Cost.
- Maximization of profit in terms of ROI for a set of countermeasures.
- Multi-objective function for minimizing the probability of attack success and security investment cost.

The first objective function may not be able to provide an optimal set of countermeasures for an attack due to limitations, including security budget constrains or lack of countermeasures for specific attacks. In this case, partial solutions for defending only a critical part of the infrastructure should be taken. The use of the second objective function can restrain the countermeasure cost to specific investment cost. According to the authors, the next two functions pose computational challenges to the proposed framework as the optimization problem is non-linear. To compress the computational requirements, the authors use Watters' transformation [123]. Specifically, this transformation converts zero-one polynomial formulations to equivalent linear zero-one formulations by simply adding constraints on the product terms. This downgrades the problem to a linear integer programming one.

The authors provide experiments to prove the ability of their proposal to converge fast to an optimal solution. The results show that their framework can yield a solution in 38 seconds for an ACT of 5000 leaf nodes. Specifically, the authors noticed that for large ACTs, the first objective function tends to apply countermeasures to the higher nodes of the trees. For the rest of the functions, the countermeasures are applied at the lower levels as the goal is to minimize the security cost and measures at higher nodes tend to be more sophisticated. Also, the authors give a comparison between their solution and the Response and Recovery Engine (RRE) framework proposed in [40] (recall that the latter uses an ART structure). Their approach resulted to an optimal solution in approximately 17 seconds, while RRE needed approximately 3 minutes when using the third objective function.

Analysis: The authors make use of an ACT structure to model the system's atomic attacks and countermeasure events with the intention of protecting the goal residing in the root of the tree. For a small-scale network consisting of 7 hosts (with 12 vulnerabilities each) the proposed solution converge to an optimal solution in a reasonable time frame. Particularly, the ACT-based method computes an optimal countermeasure set within 17 seconds ± 2 seconds. The authors technique to cast the optimization problem to an integer programming one with linear complexity is reflected to the improved performance of the system. However, even if ACTs perform better than ARTs, it has to be stated that the testbed used corresponds to a small-scale network topology. Another potential limitation of this work is the number of ACTs needed to represent the possible attacker's goals in the system. As every tree has one root, one can argue that for protecting multiple assets or properties in the infrastructure there is a need to deploy several ACTs, thus increasing the computational requirements.

Albeit the authors introduce a multi-objective function, they do not elaborate on the time needed for their proposal to provide a solution when using it. However, a multi-objective function is of major importance in a countermeasure system, where the optimal goal is to define optimal solutions which achieve a balanced trade-off between the several metrics. In addition, the diverse cost functions used to compose the objective functions do not follow any specific standard, while the attack cost and attack impact values used are derived from rather aged works.

The ability of the system to adapt dynamically during an attack incident is questionable. Specifically, the authors claim that ACT could be transformed in a hierarchical model that supports Markov chains for modeling sequential attack events. Even so, no evaluation results are given on this aspect of the problem, and thus the authors' proposal can be characterized as static. The administrator's engagement level is not defined by the authors, but we can infer that the creation of the ACTs has to be supervised by a human. This is because a decent knowledge of attack patterns is needed for identifying the internal nodes, the relationships among them, and finally the objective residing at the root of the tree.

D. Viduto et al. [45]

Description: This work proposes a novel *Risk Assessment and Optimization Model* (RAOM) to cope with the problem of security countermeasure selection. Considering the budget limitations and the high demand in security measures, the authors present a model which addresses these issues and at the same time maintains an acceptable level of performance.

Based on NIST SP800-30 [124], which aims to provide guidance for conducting risk assessments of federal information systems and organizations, the proposed model offers a quantitative improvement to it mainly for conducting impact analysis and risk determination. For the first task, the authors calculate the impact of the identified vulnerabilities in relation to *Confidentiality, Integrity, Availability* (CIA) impact, expressed by CVE [50]. The different combinations of impact levels (Partial, Complete, None) are grouped into an impact scale with discrete level [10, 50, 100]. For the risk determination task, the methodology proposes to calculate a *Total Initial Risk* (TIR), which is defined as the initial risk in an organization when no security countermeasure has been applied.

DRAFT

During this procedure, an analysis of the system vulnerabilities is conducted with the purpose of identifying their source and properties. This stage can be executed in different ways; specifically, using automated vulnerability scanning tools, performing penetration tests on systems, employing vulnerability modeling techniques, and assessing ICT documentation of a previous risk assessment. The next step in the model is to perform a threat analysis, because the idea of the authors is that vulnerabilities can only be translated into risk if there is a threat able to exploit them. To gather information about threats, they suggest to use historical databases of previous attacks recorded by the organization, or to use threat modeling techniques, such as AGs or ATs. A matching between threats and vulnerabilities is conducted to estimate the likelihood of a threat over a specific vulnerability. Also, in this case, the likelihood can adopt values in the discrete range [0.1, 0.5, 1], where the values of 0.1, 0.5, 1 correspondingly represents low, medium, and high likelihood of a specific threat exploiting a certain vulnerability.

Then, the authors propose a list of generic countermeasures, following a classification similar to NIST report in [125]. In particular, three categories are presented (Technical, Management, Operational) with four subcategories (Support, Prevent, Detect, Recovery). The countermeasures are then associated with vulnerabilities. Actually, the association method between countermeasures and vulnerabilities they follow has been initially proposed in [126] and later demonstrated in [127], where a matching value is assigned to each countermeasure. This value is within the discrete range [-1, -0.5, 0, 0.5, 1], where [0.5, 1] represents a partial or total addressing of the vulnerability, 0 means no match, and [-1, -0.5] implies that the application of the countermeasure directly or indirectly creates another vulnerability.

Each of the listed security measures has an associated cost of implementation, namely purchase cost, operational cost, training cost and manpower. The total cost for the security controls, together with the total risk level of the system, compose the multi-objective optimization function, which must be minimized.

To solve this MOOP, the authors develop a *Multi-Objective Tabu Search* (MOTS) technique [78], where the objective function is evaluated for different values of the binary countermeasure vector. Also, an experimental section is given, where the characteristics of MOTS are compared with the traditional exhaustive search, which can find the exact set of optimal solutions. In particular, comparisons in terms of speed and quality are shown.

Analysis: In the described work, the authors introduce a model to optimize the selection of the security countermeasures. Given a set of k generic countermeasures, represented as a single bit in the countermeasure vector, the search space has 2^k possible solutions. Precisely, as the size of the problem increases, that is, the number of the possible security hardening options becomes considerable, the time required to solve the problem increases exponentially.

The solution presented by the authors consists of a multi-objective tabu search. This technique is able to find the Pareto optimal solutions for the problem, avoiding sticking in a local minimum. Using this methodology, they were able to present a stable pool of founded solutions after 8000 iterations of the algorithm. In particular, 54 non-dominated solutions are discovered in 163 seconds. Overall, the authors argue that in comparison to exhaustive search, which finishes its execution after 2466 seconds, the tabu search offers superior performance. Another experiment is conducted on the quality of the solutions. In this regard, the authors show that their algorithm can reach up to 30% of the optimal solutions, and also the rest differ slightly from the optimum. From these results, we can infer that MOTS methodology performs better than exhaustive search, but in our opinion a study should be conducted to compare it with other multi-objective optimization techniques, so that the eventual advantages are clearly visible. In this mindset, Saraha et al. [128] argue that a pure tabu search is not enough to comprehensively explore the search space, proposing instead a solution which incorporates also some ilk of GA.

The paper deals mainly with risk assessment, which as we stated in Section III-B4, represents the process to identify potential risks that may be involved in an ICT system. Even though this phase has a crucial role in cyber defense, it represents a preventive action to defeat eventual cyber crooks, lacking of dynamic capabilities to react against an ongoing attack. Obviously, an interesting improvement is to try to adapt this methodology for use in a dynamic environment.

Similar to other surveyed works, the authors neglect attack modeling. They only suggest threat modeling techniques, such as AGs, to gather information about potential attacks. Instead, the assignment of the probability of a specific threat acting over vulnerability is done using logged attack attempts and self-expertise. The value of this likelihood belongs to a discrete interval composed of three values. However, the effectiveness of this procedure is debatable, even for a static type of reaction. That is, the use of historical databases lacks in considering first experienced or zero-day attacks, thus exposing the system to undue risk. Moreover, the know-how of the security experts is a good source of knowledge, which has to be used, but relying on human subjective belief may lead to a number of errors too. In this direction, the use of a decision support system can be valuable in helping the system administrator in the decision making process, so that a limitation to these errors can be imposed.

The authors highlight in their work the possibility of matching countermeasures and vulnerabilities. Note that this idea is not novel in this particular field, but represents an interesting feature of the model. This correlation considers also the chance that a countermeasure can indirectly or directly create a new vulnerability. Nonetheless, this match is effective if a complete knowledge about security controls and vulnerabilities exists. Gathering these pieces of data is an arduous task, so a broader study should be conducted in this direction, since such correlation is useful to build a dynamic strategy.

E. Chung et al. [129]

Description: In this work, the authors concentrate on the protection of cloud infrastructures. In such scenario, there

DRAFT

are several peculiarities which need to be considered by the security expert: (1) administrators do not have the complete control over virtual machines (VMs), so they may be unable to patch the system vulnerabilties as in the case of common data centers; (2) cloud users can install vulnerable software on their own VMs; (3) the compliance of the *SLAs* (Service Level Agreements) is a priority, so the reaction strategy to cyber attacks should be included in the Business Continuity Plan [130]. Furthermore, an attacker can benefit from these security issues exploiting vulnerabilities on a much greater number of machines, which can be used as *zombies* to carry out a large-scale Distributed Denial-of-Service (DDoS) attack.

To address these problems, the authors propose *NICE* (Network Intrusion detection and Countermeasure sElection in virtual network systems), a framework which is able to detect and counteract possible attacks against the cloud infrastructure. Taking advantage of the SDN network control approach [131], where network functions can be controlled and programmed through software switches and the OpenFlow protocol [132], the authors deploy NICE with the following constituents:

- *NICE-A* - A mirroring-based NID agent installed on each cloud server, which filters and analyzes the incoming traffic.
- *VM profiler* - It uses the knowledge about services, connections and states to create an accurate profile of each VM.
- *Network Controller* - It supports the programmable network capabilities to realize the virtual network reconfiguration feature based on OpenFlow protocol. This feature is crucial for the entire framework, because it controls also the traffic flows within the network clusters.
- *Attack Analyzer* - It uses a classical AG representation to correlate the alerts stemming from the agents and select the most appropriate countermeasure.

When a vulnerability is discovered or some VMs are identified as suspicious, several countermeasures can be triggered to limit attacker's capabilities. Based on CVSS [105], CVE [50], and NVD [82], the system is able to calculate the score for each vulnerability and use them for constructing on the AG. Based on the above metrics and scores, a *virtual-networking-based* countermeasure pool is built, where each countermeasure is presented with (I) cost, defined in a range from 1 to 5, (II) intrusiveness, which represents the possible negative impact on the SLAs, (III) condition, which is the requirement to enable the countermeasure implementation, and (IV) effectiveness, expressed as a percentage of probability that the state of a specific node can change after the application of a countermeasure. The optimal countermeasure selection is shown as a MOOP, which has to minimize both the cost and impact while maximizing the benefit (via the use of the ROI index [133]).

The authors offer an initial implementation of NICE in a small public cloud environment, and then they extend their analysis using a bigger private one. They monitor the introduced overhead and the security performance using a VM Security Index (VSI) [134].

Analysis: NICE is presented as a framework which is capable to cover two different phases of the network life cycle pertaining to attack incidents, namely detection and reaction. However, the target of this work is the countermeasure strategy, so our analysis focuses on the latter.

The idea of developing a countermeasure system in a cloud environment is innovative. To the best of our knowledge, this is the first attempt to deploy an architecture that is able to detect and react in a virtual scenario. This feature carries also the disadvantage that without a solid background of research, this work could not address all the issues which are present in a real environment. Moreover, the novelty in the usage of cloud infrastructure makes the attacker modeling still incomplete. That is, it is very difficult to find a great number of known attacks in the literature, and this makes also the defense strategy still incomplete.

AG is the selected model to represent the possible attack paths into the cloud system. It is built based on (1) cloud system information, collected from the controller, (2) the virtual network topology and configuration information, which include also the raw traffic data, and (3) vulnerabilities information generated from periodic vulnerability scanning and via penetration testing using the public available vulnerability databases. Once a new vulnerability is discovered or a countermeasure is implemented, the graph is updated. Apart from the scalability problem regarding this attack representation, the authors make a strong assumption; the hypervisor is secure and free of any vulnerability. However, the literature is full of documented attacks targeting it [135].

To select the appropriate countermeasure, the presented algorithm has a complexity of $O(V \times CM)$, where V is the number of vulnerabilities, and CM represents the cardinality of the countermeasure set. In this way, the authors claim that they solve a MOOP based on the ROI index in an effort to avoid a negative impact on the SLAs. A sample of possible actions is presented with subjective values of intrusiveness and cost. Specifically, as expected in a cloud environment, these countermeasures pertain to layer-2 and 3 of the OSI stack of protocols. The exploration of a wide range of possible reaction steps with different values for the suggested security metrics would greatly benefit this work.

The described system incorporates both static and dynamic reaction. To enable them, the authors consider both periodic vulnerabilities and agent-based traffic controls. A dynamic reaction to an attack is presented on a small-scale cloud system, showing that the countermeasure selection process works well with a limited number of countermeasures, and the performance is better in comparison with other proxy-based Network IDS. However, further work is needed to extend the deployment in a large-scale network. In such a setting, there is also the option of distributing the computational overhead of the control center, which naturally represents a *single point of failure*.

An interesting feature of the authors' framework is the capability to react in a *completely autonomous* way; once an alarm is generated from the *alert correlation engine* (due to the exceed of a specific threshold), the system selects a countermeasure from the pool. From an administrator's viewpoint, this reduces the human effort, especially in a virtual environment where many users share the same physical resources.

DRAFT

Nevertheless, one can argue that in particular situations, and for critical resources, the reaction strategy cannot completely exclude the administrator approval, but it has to consider their expertise in the decision process.

A notable shortcoming of this work is the handling of zero-day vulnerabilities. The solution proposed by the authors is a profile database for the VMs, but one can argue that this is insufficient to solve the problem, and the proposed IDS agent by the authors (Snort [32]) is a rule-based one, thus incapable of detecting zero-days.

F. Wang et al. [107]

Description: The work in [107] introduces a probabilistic approach for optimal security hardening. The authors aim to bridge the gap between system vulnerabilities and organization-level security metrics. To do so, they propose a methodology which incorporates AGs and Hidden Markov Model (HMM) to describe probabilistically the interconnections of the numerous security states of a system. Precisely, the authors extend Multiple Prerequisite (MP) graphs [57] by introducing 3 types of labels to the observable subjects of the network. The tags *Solid*, *Soft*, and *Dark* are used for marking the physical assets (e.g. servers), any measurable notion (e.g. network traffic), and the system vulnerabilities in the system. According to the authors, this approach reduces the size of the graphs, while important characteristics of the attacks can be identified which in turn leads to a better estimation regarding the security state of the system.

By taking advantage of the AGs portraying the interconnections in the network, the authors apply HMM to estimate probabilistically the possible security states of the system. HMM enables the quantitative analysis of the security hardening problem, while it can be used to model uncertainties introduced in the process. By employing HMM the authors can probabilistically predict the system's state transitions given a set of network observations. In this way, the defender can be notified about which state the target of the attacker is going to be, and thus take the appropriate decisions for blocking potentially harmful transitions.

For conducting a cost-benefit analysis, the authors extend the AG into a *directed state contact* one which is able to represent the state transitions [107]. Based on the interconnections of the graphs, the authors use a cost function to quantify the cost induced due to system transitions. The cost function is a weighted one which considers two individual costs, namely the *attack cost* caused by potential vulnerabilities, and the *defense cost* derived from hardening measures. Both these costs are modeled using the Butler's risk assessment framework [120]. This enables the synthesis of several problem-specific security metrics, which can be reviewed and put in practice by the security administrators. The problem is modeled as an optimization one aiming to minimize the cost function.

In this direction, the authors utilize a heuristic algorithm based on the ACO family [77]. Precisely, the *ants* roam probabilistically on the graph based on the probabilities indicated by the HMM and add pheromone on the edges of the graph. Ant path selection decisions are taken by considering two parameters, namely the *attractiveness* and *trail level* of the move, where the latter incorporates also the cost of the trace.

In the conducted experiments, the authors manually construct a *directed acyclic graph* (DAG) and determine the HMM elements. They also assign values on the edges considering NVD-CVSS [105] framework and experts' knowledge. Throughout the experiments, the authors tested the ability of their solution to identify the root causes of three different attacks, and they demonstrated that their proposal is able to balance the trade-off between the defense and attack costs and deliver near-optimal solutions.

Analysis: As already pointed out, the authors make use of AGs and HMM for representing the problem and rely on ACO to define optimal hardening solutions. Following the discussion of Section III-B, the AG technique faces scalability problems as the size of the network increases and the possible interconnections between the different nodes lead to a system state explosion. On top of AGs, authors utilize HMM to create a probabilistic mapping between the AG and the various system states, and to model the uncertainties. As with the majority of works in the literature, the experiments are based on a rather small network topology with a limited number of system states and 4 types of countermeasures, namely *Disconnect, Patch, Disable* and *Configure*. More specifically, two testbeds have been created having 14 and 18 states, respectively. In this context, the proposed framework is able to provide a near optimal defense solution in a reasonable time for both testbeds, respectively 27.23 and 29.11 seconds. However, as the time complexity of the algorithm is $O(M \times N^2)$ (where M represents the network states and N the number of hosts) it can safely be asserted that the solution is not scalable for large-sized networks.

The quantitative analysis of the framework is based on the Butler's framework [120], very similar to [115]. This framework enables the introduction of several types of security metrics in order to define complex cost functions. Further, the authors refer to CVSS [105] for defining security metrics that will allow them to quantify both attack and defense consequences. As already mentioned, the use of standards, can add value to a work as it caters for a solution which is aligned with globally accepted and deployed ICT practices.

It is to be mentioned that the proposed model is not destined to react dynamically, but to infer on countermeasures that can achieve the near-optimal trade-off between the attack and defense costs. That is, authors' proposal aims to assist the decision maker, while demanding the active engagement of a security administrator for determining the assets under protection and tuning the parameters of the framework.

The optimization strategy followed in this work is quite interesting. The authors argue that it is impractical to exhaustively traverse the search space of the problem to define the optimal defense strategy. This is why they utilize ACO to pinpoint the root causes of specific attack scenarios. This approach narrows down the focus of the problem significantly and it can be used to identify critical assets in the infrastructure. In addition, ant colonies are a fast solution that can provide a satisfactory solution quickly [136]. ACO also guarantees that the algorithm always converges to a solution.

DRAFT

The aforementioned ACO qualities could be proved beneficial in dynamically changing graphs, where new systems states may appear and a defense mechanism should define fresh optimal solutions in a reasonable time frame.

G. Zonouz et al. [137]

Description: The authors in [137] introduce a framework called EliMet with the aim of estimating whether the security requirements in a system are effectively met. EliMet combines expert knowledge and reinforcement learning to support decision making against intrusive incidents. The system is driven by system-specific security measures to infer and identify risky system states with the purpose of suggesting the administrator appropriate healing actions. The system passively observes administrator's reactions against offensive incidents to calculate the aforementioned security measures. According to the authors, by using these measures, the system not only is able to respond in an autonomous manner, but also to predict potential security threats that administrators should take care in advance.

More specifically, the state of the system is modeled as CMDP [69], where the competitors are the adversary and the defender who both aim to increase their profit. For estimating the system's states, the proposed model considers the following parameters: i) the security state space, ii) a set of actions divided into *adversarial* and *response* actions, iii) a security measure function, and iv) a probability of shifting to a new system's state. In addition to these parameters, EliMet models the insufficiency introduced by potentially false alerts of the IDS. Thus, the *belief* about every security state is measured probabilistically based on the previous state of the system combined with the alerts observed for the current state.

Given the CMDP model, the system identifies the optimal defensive solutions based on the security measures. EliMet treats the reaction procedure as a *maxmin* problem and finds the optimal defence policy by maximizing the security measures through Bellman's optimization method [138], known as dynamic programming.

As already pointed out, EliMet passively observes the security expert's reaction on intrusive incidents. In parallel with passive observation, the system uses an inverse reinforcement learning algorithm. The latter incorporates the expert's response policy and the CMDP model with the aim of iteratively refining the security measures so that they converge with the expert knowledge. The proposed algorithm takes also into consideration the defender's expertise level in calculating security measures. After a specified period of time, the system concludes to an optimal policy for the corresponding CMDP model, which could be used later in an automated response system.

The reinforcement procedure is able to result in efficient policies when it comes to common incidents that appear frequently in a system's defense life cycle. However, for rare system states, where the policy uncertainty is high in terms of Shannon entropy [139], the system queries the expert for the appropriate action. The incidents which will be queried to the expert are decided based on two criteria. On the one hand, the less information the system knows about the rare state the higher the possibility for the system to generate a query to the expert. On the other, a query for a system's state may arise or not, based on the potential return benefit for the defended system. In other words, the more information gain stems from a state transition the higher the chances for a query to be generated.

Another feature of EliMet is its ability to perform contingency analysis. The system imitates the adversary by choosing the actions that increase the offensive benefit, while at the same time takes the optimal defense actions. In this way, the system identifies risky states that the administrator should take care of and critical assets which should be protected more intensively.

Analysis: EliMet is an interesting framework as it combines several techniques to deal with the problem of providing optimal countermeasures. More specifically, the authors utilize CMDP graphs to represent the system's security states. According to their evaluation, the graph creation occurs in a reasonable time frame, that is, 400 milliseconds for networks with 37 nodes, while the number of possible system states affects the response time of the system proportionally. CMDP is built based on the topology of the system and global access control policies, while the system considers IDS alerts as the sole input events. However, possible critical system states can appear upon system updates or the emergence of new vulnerabilities. It seems that such information is not taken under consideration by the authors during the graph creation. In general, the CMDP graph representation seems promising, but as the authors state, dynamic changes in the system topology are sure to pose a challenge.

Besides its ability to dynamically apply defense policies, another strong aspect of this proposal is its capacity to perform also static risk assessment for identifying critical assets proactively. However, both the risk assessment and the security metrics utilized to quantitatively evaluate the system's effectiveness do not follow any specific standard. The authors provide a general purpose framework, yet it is unclear how the security measures are defined and how complex and multi-objective measures could affect the overall performance of the framework.

The authors utilize reinforcement learning in an effort to capture and integrate the experts' wisdom in their framework. In their experiments, the refined security measures generated by the reinforcement learning algorithm are compared with the measures produced by the expert's actions. According to the produced security measures, the algorithm seems able to imitate the expert's actions, but in some cases, the actions taken by the algorithm overprotect the states of the system. Naturally, the ability of EliMet to converge with the expert's action is a positive feature. However in the context of a dynamic reaction system, if the system fusses over its assets may cause service disruptions and monetary losses.

Finally, the authors do not provide any information about the set of actions which constitute the defensive policies and the ways the size of this set can affect the performance of the system. Even though the CMDP seems to be scalable, a possibly hefty set of defensive actions could increase significantly the response time of the system.

DRAFT

H. Zonouz et al. [40]

Description: The authors in [40] propose an automated system for cost-sensitive intrusion response. They model the intrusion response problem as a multi-step, sequential, hierarchical, non-zero-sum, two-player stochastic game between the adversary and the defender. The two entities present conflicting interests, and their utmost goal is to increase their benefit by taking sequential actions. The authors utilize ARTs to model system states which later are transformed automatically into partially observable competitive Markov decision processes (POCMDPs). In fact, they adopt this representation in order to apply Bellman's optimization method [138] that will allow them to define an optimal defense policy based on the state of the system. Bear in mind that this approach is used by the authors in EliMet framework [137].

Here, authors utilize a game-theoretic approach to model the relationship between the attacker and the defender. Under this prism, the one entity adapts its behavior according to the strategy of the other. The defender is the first who makes a move in the game and then the attacker responds in a sequential Stackelberg stochastic game [140]. This sequential behavior continues to an infinite-horizon and every movement leads probabilistically to a new system state. The state transitioning is modeled in POCMDP by also considering the uncertainty for the exact state of the system caused due to uncertainties derived from the IDS. In this way, POCMDP conceptualizes the system's states as *belief states* instead of exact ones.

The authors proposed a decentralized architecture to implement their mechanism. Their notion incorporates local engines placed in individual hosts in the network and a global engine located in the RRE server. The local engines are subscribed to an intrusion alert database in order to get notified when an intrusive incident relative to their host appears. Additionally to the IDS alerts, the local engines undertake also the creation of ARTs for the local host. Hence, the local engines are responsible for modeling the local system and react against offensive events. Every local engine has at most 3 ARTs where each one is used to infer on possible violations against the CIA of the host. The root node of each ART holds an accumulative security estimation value (δ) which is sent to the global engine of the RRE. This engine is responsible to inspect the security level of the whole infrastructure.

The global engine takes as input the network topology, the acknowledged system vulnerabilities, and the connectivity matrix and creates a CMDP automatically. The same engine aims to safeguard the system using global objectives defined by the administrator of the system. For this purpose, the authors utilize a Fuzzy control-based technique to enable the administrator define Fuzzy rules in the form of IF <premises> THEN <consequent>, which are then used to infer on the security level of the system. The δ values derived from the local engines are converted to qualitative values (high, medium, low) to meet the ones set in the rules by the administrator. Then, a center of gravity defuzzification method [141] takes place to provide a quantitative score for the security level.

Analysis: The authors introduce several methodologies to deal with the problem of the dynamic intrusion response. The main contribution of this work is the distributed nature of the proposed framework. Based on the literature, we can safely extrapolate that finding the optimal defense solution against an attack is a computationally intensive task. In this direction, a distributed model to deal with this problem can improve the scalability and performance of a response system as the computational burden is relocated to its hosts. On the downside, this approach poses also certain limitations. The purpose of every host in a system is to deliver one or more services, and thus adding an additional intensive task could lead to service availability issues. This resolution can be used in networks where the participating hosts are able to manage this kind of procedures. However, in environments with limited processing power on the edge nodes the same approach may be impractical. Furthermore, trust issues emerge in distributed mechanisms destined to deliver security services. As a result, trust models [142] have to be used to ensure the legitimacy of the nodes, given that misleading results may occur due to contributions stemming from compromised nodes. Finally, decisions made in an automated and autonomous manner to the edges of a network may disrupt its normal behavior, while at the same time the administrator may have partial observability of the system state.

The authors use ARTs to represent the security state of the system, but for defining the optimal defense solution the methodology instructs the transformation of ARTs to a CMDP. This of course comes at the price of additional overhead. Moreover, authors do not rely on specific and global accepted metrics to support the optimization process. Instead, they provide a formula which can feed a cost function to enable a quantitative analysis. As already pointed out, every local node has 3 ARTs, one per CIA property. Even though it is essential to quantify the impact on CIA properties, it is debatable whether or not a defense solution should consider more metrics to provide a complete quantitative defense analysis.

The local engines are capable of reacting against intrusive incidents automatically. Still, the system demands the complete engagement of the administrator for defining the global security objective of the system in terms of Fuzzy rules. According to authors' evaluation, the system performs well for quite large sized ARTs. However, they do not elaborate on the number of available countermeasures to cope with every possible state of the system. In case there is a pool of likely reactions against an incident, this is translated to an expanded search space and the problem's complexity could be further expanded by incorporating multi-objective cost functions in the optimization process.

I. Granadillo et al. [143]

Description: This contribution proposes a geometrical model to select the optimal combination of countermeasures based on the *Return-On-Response-Investment* (RORI) index with the aim of counteracting cyber attacks against critical systems.

The authors present their improved version of RORI index, discussed in [144], [145] by extending the approach proposed initially by Kheir et al. [64]. Specifically, they modify the

DRAFT

initial formula to also include the infrastructure value expressed as *Annual Infrastructure Value* (AIV), and to handle the possibility of applying a null set of countermeasures in specific scenarios.

Also, the authors extend the definition of attack surface of a given system, presented in [30]. To do so, they present a "volumes" model, which represents systems, attacks and countermeasures in a three dimensional coordinate system. The dimensions correspond to the users, communication channels and system resources. Specifically, the volumes are defined as follows:

- *System Volume* - It represents the maximal space a given system exposes to users and attackers.
- *Attack Volume* - The portion of the total volume being targeted by a given attack based on the vulnerabilities it is able to exploit.
- *Countermeasure Volume* - It represents the percentage of system volume that is covered and controlled by a given countermeasure.

The three dimensions are defined following the access control methodology [146], [147], and are identified as the ones which contribute directly to the execution of a given attack. That is, user account as the *subject*, resource as the *object*, and channel as the way to execute an *action*. The dimensions are then populated, and by following the CARVER methodology [148], a weighting factor is assigned to each element represented in the Cartesian system. In this way, a numerical bijection is created between the real elements and their representation within the coordinate system.

In this three dimensional system, the represented volumes appear as 3D geometrical figures (parallelepipeds) within the system volume. Also, the attack and countermeasures volumes are calculated and represented considering the used dimensions. For multiple attacks and countermeasures, a study on the volume union and intersection is conducted, showing the possible ways to calculate the contribution of both joint and disjoint volumes.

The main idea of the authors is that by using this graphical representation, it is possible to determine not only the impact of each attack and countermeasure (or the impact of a group of them), but also the residual risk (i.e., the volume of the system that is being attacked but is not covered by any countermeasure) as well as the potential collateral damage (i.e., the volume of the system that is not being attacked, but is covered by a countermeasure).

The authors also offer a prototype implementation, in a form of an application, which generates the above mentioned 3D graphical representation of the system, attacks, and countermeasures. The same application is able to automatically evaluate, rank and select the optimal set of countermeasures against complex attacks. The platform is composed by two modules, namely the Attack volume application, and the RORI application. The first one is responsible to map the attacks and the countermeasures into the Resource-Channel-User 3D system by calculating their monetary impacts. The second is in charge of performing the evaluation of individual and combined countermeasures, taking as input all the needed parameters for RORI calculation. If some of them are missing (i.e., *Annual Loss Expectancy* (ALE) or *Risk Mitigation* (RM)), they will be requested to the Attack Volume engine, as they depend on the specific attack context.

To demonstrate their solution, the authors offer a case study pertaining to a critical infrastructure control system. Using this paradigm, they show the selection of combined countermeasures for a particular attack. To complete this task, the authors follow the approach presented in [145], which also considers any possible restrictions among the countermeasures (mutually and partially exclusive, restrictive).

***Analysis**:* This proposal builds on previous works by the same authors with the aim of using the RORI-based countermeasure selection together with an interesting geometrical representation of the involved defensive or offensive quantities. More specifically, in [144], they highlight two main limitations of the RORI-based model, in particular the accuracy in the estimation of the different parameters in the formula, and the unconsidered interdependence among the various countermeasures. Actually, the first limitation derives from the difficulty of getting a real assessment of variable parameters in the RORI formula. That is, they identify the ALE of an attack and the RM level of a specific countermeasure as the most challenging, thus requiring a considerable effort. A great improvement toward this goal is made in [149], where the authors propose statistical methodologies to estimate the above parameters using epistemic uncertainty [150]. The second limitation is only partially addressed in [145], where the authors study the results of their model applying a combination of two or more countermeasures.

It is to be noted that the authors decide to neglect the attack modeling. This is obvious because in the presented case study, the process starts with the evidence of an attack in the system. In this way, they are able to demonstrate the feasibility of their procedure in a dynamic environment. Nevertheless, even though the detection belongs to a different phase in the network life cycle, modeling the attacker's steps aids the reaction phase as well. This is because the defender is able to predict the attacker's trajectory in the system more easily. Overall, attack modeling is essential, because nowadays cyber attacks are increasingly disruptive, and the reaction time is a decisive factor. Moreover, the process in charge of generating the system, attack, and countermeasure volumes adds a further delay. It seems that more experimentation needs to be done to obtain a better view of the authors' proposal in terms of performance. Another idea would be to consider extending the volume representation with extra dimensions, that is by adding time as an extra axis.

Recall that for calculating the overall contribution to the volume representation of each selected category within an axis, the authors follow the CARVER methodology, whose goal is to measure the priority of each element in a given system. This measurement is based on 6 factors, namely *Criticality*, *Accessibility*, *Recuperability*, *Vulnerability*, *Effect*, and *Recognizability*. The proposed methodology assigns a numerical value to each considered factor on a scale of 1-10 and places them in a decision matrix. The sum of the values indicate the severity of a given dimension. The CARVER matrix was developed from US special operation forces, and it

DRAFT

can be used both from an offensive or defensive perspective. Nevertheless, this methodology is not widely adopted by ICT community, which rather prefer to use other open standards.

In the authors' model, countermeasures are proposed to be implemented for a short period, that is, from the moment of the detection of the attack incidents until the system returns to a safe state. According to the authors, this approach is preferable because it does not need them to compute long-term investments in the proposed procedure. Due to the complexity of the search space and the inaccuracy of the results, they discourage the usage of genetic and heuristic algorithms. However, this option is debatable, because nature-inspired techniques may be proved particularly effective, especially when the pool of the available countermeasures becomes large.

In the context of the 3D representation used by the authors, the coverage of a countermeasure is defined as the percentage of system volume it is able to cover. In this mindset, they are able to calculate the percentage of volume that a countermeasure can cover for a specific attack. However, in practice, it is not straightforward to establish a direct mapping between countermeasures and attacks, especially if one considers representing them in another reference system. In addition, they present this countermeasure volume coverage as a percentage. In this way, the evaluation of joint or disjoint volumes is not simple, and it requires a preventive analysis. The negative impact of a combined solution is not considered as well, while only the cost impact is computed in the model. Nevertheless, the latter requires more effort, especially for evaluating the impact on the availability of the service for a combined countermeasure, whose effects are not expected to be negligible.

J. Miehling et al. [151]

Description: In [151] authors model the defense problem using the notion of *POMDP*. Their goal is to provide optimal dynamic defense policies to counteract in-progress cyber attacks occurring against the protected system. To do so, they utilize BAGs and Moving Target Defense (MTD) [152] schemes for implementing defense policies able to adapt to the adversary's behavior in a dynamic manner. The proposed model is guided by security metrics that quantify the trade-off among the *CIA* of the system in an effort to secure it without interrupting the provided services.

More specifically, the authors utilize BAGs to represent the adversary's possible movements in the system by also considering probabilities in every step (edge) on the paths that connect the entering (leaf) nodes toward the root nodes of the graph. The spreading ability of an attacker depends on the type of the node expressed by AND/OR relations, the previous state of the attacker, and the exploitation probability of the node. In addition to these aspects, the authors engage also a factor to emulate the defender's partial observability. Under realistic terms, malicious actions may occur in a system without raising intrusion alerts. This is why the authors introduce the *probability of detection* factor in their model as well. This uncertainty about the current state of the system forms a *belief* about the present network's state.

The countermeasure strategy is defined as a set of individual actions where, based on an incident, an optimal subset of those can be used to counteract an intrusion. Every defending action directly affects CIA metrics. More specifically, a cost function should be able to reflect both the negative and the positive impact to the CIA metrics for the applied actions. To this end, the authors treat the defense problem as an optimization one where the ultimate goal is to define an optimal defense policy. They take into consideration both the *belief* for the current system state and the future states to optimize the cost function through dynamic programming (Bellman's method [138]).

The evaluation of the proposal has been conducted on a small BAG with 12 attributes and only one of them as critical (root). The graph has two leafs and two possible countermeasure actions. According to the authors, the possible states of the system is 2^{12}. In order to reduce the search space of the problem, the authors add restrictions by considering only *AND* relationships among the attributes and monotonic threat propagation (i.e., the attacker aims to constantly increase their benefit) and result to a minimized problem with 29 states. The POMDP model of the example network was solved in terms of identifying the optimal solution using a POMDP-solver written in C programming language. The outcome is a network *heat map* which represents the critical nodes of the system. The countermeasure strategy is shown intuitively and can be inferred from the heat map. The authors suggest the adaptation of those countermeasures that reduce the exploitation probability of the risky nodes.

Analysis: As stated previously, the authors make use of BAGs to model the system's security states. Albeit this type of representation is suitable for representing also the uncertainties which are introduced, the limitation of scalability affects significantly the efficacy of this solution. Specifically, as shown by the authors' experiments, the conceivable states of the system combined with the numerous possible countermeasures expand significantly the search space of the problem. The experiments conducted in a small-scale scenario and under specific assumptions, aiming to minimize the search space. On top of that, the example scenario engages only two countermeasures and one root node in the graph. To this end, one can argue that the complexity of the solution is high and can be significantly increased if the solution is utilized in a dynamic environment trying to model the behavior of a moving attacker. However, the employment of the detection probability as well as the probabilistic contagion spreading model they use emulate the imperfect environment in which a security expert has to take actions.

The cost function used for quantifying the cost of an attack or the deployment of a defense action is calculated based on CIA metrics. CIA properties should be in the core of the countermeasure strategy, but we argue that a framework should include several types of metrics to achieve a better quantification of the problem. Apart from that, it seems that the authors do not adopt any globally accepted security automation standards in their model.

Finally, although the authors claim that their proposal suggests an automated defense tool, they do not evaluate the performance of their proposal in this direction, while the

DRAFT

countermeasures taken in the conducted experiments aim to block or disable a given service in the system. These actions, however, can be considered as extreme measures to deal with an attack. The goal of a countermeasure strategy is to explore optimal policies to deal with a security incident instead of applying extreme measures.

K. Shameli-Sendi et al. [65]

Description: The work in [65] presents a model able to dynamically evaluate the positive and negative effects of defense actions on a system under attack. The problem of providing defense actions which maximize the security performance but simultaneously minimize the negative effects of the applied measures is treated as a MOOP. The ultimate goal of the framework called ORCEF is to provide optimal defense actions while simultaneously sustaining the quality of the services provided to the end-users.

Notably, ORCEF utilize AGs and SDGs in an effort to respectively allocate the defense points in the network and to evaluate the response negative impact. AGs are used to enable the system to find the attacker's position and final goals based on a confidence level, and to define the optimal network points where appropriate defense measures should be applied. In this way, ORCEF creates a dynamic mapping between the possible attack paths and the network topology. Further, SDGs are used for the sake of identifying the interconnections among the provided services. In this way, ORCEF is able to conduct a quantitative analysis by considering also the service dependencies and the number of users affected by the malfunctioning services.

ORCEF's response engine is triggered by IDS alerts and tries to locate the attacker on the AG. Also, it is aware of a number of parameters, including the network topology, the number of provided services in the infrastructure, and the population of users. Based on a pre-defined pool of responses and the aforementioned parameters, ORCEF calculates the positive and negative aftermath of every response by utilizing the MCDM methods SAW [153] and WP [154]. To do so, the positive effects take into account the outcomes on CIA and the performance of the system. The negative corollaries consider the effect induced on the availability of a service and other dependent on that services, the users' inaccessibility to services, and the setup cost of a defense. The positive outcomes and the cost setup of a defense are calculated statically, while the rest of the metrics are dynamically adjusted during an in-progress attack to the system. The determination of the static metrics and their importance is done during system bootstrapping, where the administrators are asked for their opinion. More specifically, ORCEF captures the security experts' opinions in the form of linguistic variables by utilizing a Fuzzy model.

Once the positive and negative effects for every response are calculated, a Pareto optimal set of defenses is generated. The optimal defenses are those which achieve the most efficient trade-off between the security level and the negative impacts. Finally, the Pareto optimal responses are ordered depending on the state of the system. If the attacker is highly skilled, then the solutions that minimize the damage cost should be selected. Also, if the resource is of high value, then those countermeasures which minimize the negative impacts should be selected.

The authors evaluated ORCEF in a topology consisting of 5 sub-networks under two attack scenarios modeling both an external and internal attacker. The system was able to react in about 489 milliseconds and 456 milliseconds for the two scenarios, respectively.

Analysis: ORCEF is a framework able to respond dynamically on ongoing attacks as it is capable to allocate the possible attack paths based on IDS alerts and the defense points of the network. Toward this goal, the proposed framework incorporates both AGs and SDGs. Although the authors' approach combines the advantages of the aforementioned representations to deliver a cost-benefit response framework, it seems that the authors do not consider the cost of initializing or adapting the graphs. Under this prism, we consider the protected topology to be static. That is, under realistic conditions a network topology can dynamically change, say, as new nodes join or leave the network. These kind of changes should be reflected also in the AG dynamically. Further, the new nodes may bare vulnerabilities, existing nodes may get vulnerable at any time, new users and services may be added, and this is why the SDG needs to be updated or recreated from scratch.

By utilizing the MCDM framework and Pareto optimal methodology, the proposed system is able to come up with optimal defense solutions on different defense points in the attack path of the topology within a short time frame. Still, the authors do not mention if their evaluation is limited in protecting a single service in the topology. In this case, the framework should be assessed under a situation where multiple services need to be protected and several countermeasures must be applied to diverse defensive points. If so, the complexity of the optimization problem is expected to further augment. Based on the results, ORCEF seems to perform fast for the given scenario. On the downside, the evaluation metrics for quantifying the impact of defense decisions on the target system do not follow any specific standard. An exception is that ORCEF incorporates the CVE standard when it comes to the alerts generated by Snort [32]. However, the authors' approach to utilize a fuzzy model for capturing the expert's opinion in the form of linguistic variables can improve the system's experience. Furthermore, the ORCEF administrators need to pass through a demanding phase of initializing the system as they have to assign a great amount of linguistic values to the system's parameters. The defenses used by the authors are applied on several guarding points according to the attack path in the topology. This is an interesting approach which minimizes the effort and the cost as the countermeasures are applied only in the part of interest on the graph. Finally, as the authors state, ORCEF is not able to generate combinations of defenses for combating an offensive incident in a more efficient way. This is because ORCEF concludes only to one single optimal solution.

L. Kotenko et al. [155]–[166]

Kotenko et al. [155]–[166] presented a series of works dealing with countermeasure strategies against cyber attacks.

DRAFT

We can categorize them into 1) those which cope with the countermeasure selection in SIEM systems based on AGs and SDGs [155]–[159], and 2) those which deal with attack modeling and security evaluation in SIEM systems [160]–[166]. Below, we elaborate on each category.

1) Countermeasure selection in SIEM system:

***Description**:* The authors define an *ontological representation* for security metrics [155], viewed as a core element of a countermeasure decision support module within a SIEM system. In their vision, the adequacy of the eventual application of a countermeasure action depends mainly on the speed and the reliability of selection and calculation of a subset of security metrics. However, in modern networks, this concept represents a hard task for the security administrator, given the huge amount of data. To solve this problem, they propose an *ontology*, which is seen as a flexible tool for describing objects, classes, relationships and attributes of a domain of arbitrary complexity. In this way, the ontology is used to select the most fitting countermeasures based on the calculated values of metrics and rules of logical reasoning. The main classes existing in the ontology are *reactions*, which in turn are divided into two subclasses, namely *alarms and countermeasures*, and *metrics*. The latter class is an abstract superclass from which several abstract subclasses are generated by applying the relationship *is-a*:

- *Malefactor metrics* - They incorporate important parameters regarding the attacker profile, e.g., *Attacker Skill Level* and *Successful Exploitation Probability*.
- *Topological metrics* - They integrate information about the network topology, e.g., hosts, applications, and vulnerabilities.
- *Attack metrics* - They describe the main characteristics of an offensive attempt, e.g., *Attack Impact* and *Confidence Level*.
- *System metrics* - They consider *Attack Surface* and overall *Security Level*.
- *Zero-day metrics* - They attempt to measure the impact of an eventual zero-day exploitation in the system, e.g., *Probabilistic Vulnerability Measure* [167] and *k-zero day safety* [5].
- *Cost metrics* - They take into account *Annual Lost Expectancy* and the *RORI* index [144].

Based on the presented classification, the authors introduce their approach to the countermeasure selection [157], [158], considering the following main requirements: (1) *security metrics*, as suggested in the corresponding ontology; (2) *AGs*, created on the base of existing vulnerabilities, network configuration and attacker capabilities by following an attacker-centric model; (3) *SDGs*, considering information of the interconnections between network services; (4) application of the SCAP for the specification of input data; (5) integration within *SIEM systems*, considered to be in active development in the recent years. In particular, the third requirement is defined by the necessity to consider possible negative impacts of the countermeasure selection in the objective functions of the system under protection. Instead, the fourth requirement is connected with the strong need to automate the security analysis process and reaction in the modern systems, especially when considering the possibility of reacting dynamically.

The authors also discuss the countermeasure selection technique, distinguishing between two main modes, namely *static* and *dynamic*. The first one is understood as a general improvement of the security level of the system, taking into account the values of metrics previously defined as input data. The latter one is seen as preventive actions for a specific ongoing attack, taking into account the SIEM events as its main input data. Specifically, in [159], the authors concentrate on the events level, because it allows considering the dynamic aspect of the security assessment and countermeasure selection. This level is based on the incoming security events, stemming from different sensors in the network and from the SIEM system. The *event model* connects this information with models of the previous level (AG, attacker models), mapping the attacker position on the AG and calculating security metrics that reflect with high accuracy the security state of the system. Based on these metrics, a list of possible countermeasures is generated. In a next step, the countermeasures are ranked taking into account the cost, effectiveness, and collateral damage.

***Analysis**:* The described works use both AGs, which allow them to define the possible steps of an attacker in the system, and SDGs, which consider also the interconnections between network services. By doing so, the authors are able to obtain the advantages of both these two attack modeling techniques, thus achieving a more accurate security evaluation. This includes the mapping of the attacker's position and their most probable paths, together with a cost-sensitive analysis, which represents the most important results. However, the algorithm proposed by the authors to evaluate these graphs based on the aforementioned metrics is not presented. The authors only describe the methodology used to achieve their goals. Yet, without an optimized methodology to analyze the abovementioned graphs, the caused overhead makes the evaluation unpractical in case of ongoing attacks which demand a dynamic reaction [56].

To specify a common approach in the development of a countermeasure model, standards from SCAP are applied. Particularly, authors use the *CRE* and *ERI* standards. The employment of these standards along with CPE for network configuration, and CVE, CWE, and CVSS for network vulnerabilities, makes the authors' model interoperable and quantitatively comparable to others. Nevertheless, the authors assume that the system has already a pool of countermeasures, which can be selected by applying an *ad-hoc* algorithm. The generation of such a pool is made based on the knowledge and expertise of the system administrators. On the downside, this assumption requires an initial significant effort from the expert, who has to fill the *knowledge database* of countermeasures, followed by a stable phase where the system is capable of reacting automatically. Therefore, one could argue that the authors' proposal lacks of flexibility, because this database of countermeasures needs to be continuously updated, and the participation of the administrators, at least in some specific and critical situations, is unavoidable.

From an attack response viewpoint, the proposed solution supports both static and dynamic modes. In this respect, it

DRAFT

is theoretically able to cover a wide range of assessments regarding the network security level. However, only few examples are presented by the authors, including some type of attacks on small-scale networks with the use of generated attack sequences and poor attacker modeling. Nevertheless, a more detailed experimental section is needed to elaborate on both the input data regarding the attacks and the network topology. Moreover, the authors claim that for a small number of security events and short attack sequences, the effectiveness of the countermeasure selection is reduced, but the accuracy and efficiency of the implemented solution satisfy the initial requirements. Still, the presentation of the results is not sufficient, which constitutes the authors' work not directly comparable against others in the same field.

2) Network Attack Modeling and Security Evaluation Framework:

***Description**:* This series of works by the same authors is different from the previous one, not only in their scope, but also because it is backed by a great implementation process. In [160], the authors present an *Attack Modeling and Security Evaluation Component* (*AMSEC*), which if deployed in tandem with a SIEM system is capable of (1) generating *ATs* and *SDGs* based on the topological vulnerability analysis along with zero-day vulnerabilities, (2) applying *anytime algorithms* to provide a near real-time attack modeling, (3) analyzing AGs to predict *future attacker's steps*, (4) calculating *security metrics* that also reflect the response impact, and (5) selecting the optimal security solution through an *interactive decision support* process.

The authors' proof-of-concept implementation supports two main modes, namely *design* and *exploitation*. In the first mode, AMSEC operates offline taking as input a model of the analyzed system. It then produces a list of weak spots and a set of ATs (one per attack scenario), which will eventually build the AG. In the latter mode, AMSEC operates in real-time or near real-time, so it can adjust existing ATs, succeeding in predicting attacker's steps and generating the appropriate countermeasures.

The proposed system is connected to external open databases of vulnerabilities, attacks, and configurations (NVD [82], CVE [50], CAPEC [92]), and translates the gathered information into a format recognizable by the AMSEC's security data repository. The latter also stores data obtained from network scanning tools and admin's knowledge. All the gathered information is used to build and analyze the AGs. The authors present a prototype which contains three basic components:

- *VDBUpdater* - It updates the internal database of vulnerabilities using information obtained from NVD.
- *Network Constructor* - It aims to create and modify network models.
- *Security Level Evaluator* - It evaluates the overall security level of the system starting from the analysis of the AGs and the associated security metrics.

An improvement of the previous model is presented in [163], where the authors present *CAMIAC* (*Cyber Attack Modeling and Impact Assessment Framework*), a framework which optimizes the AG building and analysis processes with the goal to enable their usage in near real-time operations.

In [165], the authors extend their AMSEC prototype, by proposing a novel approach to construct, modify, and analyze the AG in a faster way, showing that it can achieve better results if the analyzed network presents a limited number of changes. In this way, they claim that it is possible to monitor a large-scale network by updating only the topological changes.

***Analysis**:* In the presented works, authors use both AGs and SDGs, as with their previous series of works described in Section IV-L1. However, in this case, the difficulty of generating and maintaining these graphs in real-time is demystified. Also, it is made clear that these representations keep their actuality for a limited period of time, until significant changes in the security policies or in the network topology occur. The suggested solution for this problem is to use these models constructed *in advance*, and *updating* them with the help of ad-hoc algorithms. By doing so, the computational power required for the operations related with the construction and maintenance of the graphs decreases significantly, thus offering the possibility to represent large-scale networks and analyze the ongoing events in real-time. As already pointed out, the performance of the proposed framework in terms of applicability is critical. So, the scalability of the model should be further demonstrated with data obtained from real networks instead of simulated ones.

Regarding the type of reaction, the main idea of the authors is equivalent to that discussed in Section IV-L1. Nevertheless, their proof-of-concept is only partially implemented, thus limiting its capabilities to the detection of the attacker inside the monitored system. Therefore, the prototype shows limited risk analysis capabilities, and only recommendations are given to counteract security incidents. However, very few would argue that the framework capabilities of automatically (or semi-automatically) reacting to malicious activities is a key requirement for any countermeasure strategy.

The authors' framework works in combination with a *Security Data Repository*, used to store information updated from external sources and the results of the security evaluation of the system. This database makes an extensive use of standards (CPE, CVE, CVSS and CAPEC), thus providing a standard way to represent and report cybersecurity-related information. Note however that in the context of these works this information is mainly used for the attack modeling phase, without giving the proper importance to the reaction one.

Lastly, an interesting feature is the presence of an *Interactive Decision Support Module*. This component interacts with the admins through a graphical user interface (GUI) to let them select the most appropriate security solutions, and define their preferences regarding the different types of requirements (risks, costs, benefits). The GUI is also able to visualize the attacker's position on the network map and uses different colors to show different risk levels. Nevertheless, only recommendations are presented, without any real implementation of the suggested counteractions directly on the various network assets.

DRAFT

M. Shameli-Sendi et al. [168]

Description: In [168] the authors propose a novel IRS architecture to select and deploy the optimal countermeasure in the context of dynamic reaction to cyber threats. The problem of providing countermeasures which maximize the security performance and simultaneously minimize the negative effects of the applied measures (i.e. impact on the system services and cost) is treated as a MOOP.

Particularly, the proposed architecture leverages the capabilities of SDGs and ADTs in an effort to respectively allocate possible defense nodes and to evaluate the attack damage cost. SDG are used to identify the interconnections among the services in the network, so that both the negative impact of attacks and countermeasures can be quantitatively evaluated based on CIA attributes. Additionally, ADTs are employed to extract the paths relative to an incoming attack, thus specific countermeasures can be allocated on defense points to block the intrusion.

When an alert is raised by an IDS, the proposed IRS maps the alert to the ADT. In this way, attack paths and defense points are identified. Starting from a predefined pool of countermeasures, the IRS computes which of those can be implemented based on 3 parameters that are evaluated independently, namely *security benefit, security impact, and security cost*. Once the Pareto set is generated, SAW method [153] is used to extract the optimal solution. Moreover, the IRS evaluates the possibility to combine multiple countermeasures. To do so, the authors propose the *vulnerabilities surface coverage*, which represents the vulnerabilities a countermeasure is able to heal. In this way, joint and disjoint surfaces are computed and, consequently, the countermeasure combination which covers the maximum number of vulnerabilities is selected. Further, the selected countermeasure is deployed and its effectiveness is evaluated through a 5 secs window. Specifically, the reaction is effective if no other attacks are detected in this time frame. The effectiveness values per countermeasures are afterwards stored in the database.

Lastly, the authors present a detailed experimental section, in which they deploy a real cloud environment with 6 VMs and 15 vulnerabilities in total. An ad-hoc multi-step attack scenario is created to exploit the above-mentioned vulnerabilities. During 6 months, the authors generate 5691 attacks, recording the parameters relative to the countermeasures deployment. In addition, the SAW methodology is compared against other 2 well-known scoring methods, namely PW and TOPSIS [169]. The authors conclude that SAW is more reliable than the other, performing better for their scenario. Furthermore, a performance analysis is conducted, in which the authors show that the proposed framework is able to respond in 449 milliseconds for their attack scenario.

Analysis: As stated previously, the authors use ADTs and SDGs in an effort to model possible attack steps within the monitored system and estimate their impact on the provided services. Although this dual endeavor may result in a more accurate attack representation, it has to be noticed that the proposed framework performs with one ADT used to protect a single asset. Notably, in real conditions, the protection of a complex system will require a forest of ADTs. This increases the complexity of the problem and the time required to analyze these trees. That is, more research is needed in this direction to safely argue on the feasibility of the presented approach. Additionally, the difficulty of generating and maintaining SDGs remains an open challenge, since the authors state that the importance and dependence between the services is pre-defined by a security expert. However, identifying and representing the interdependencies of all the services in a complex infrastructure is a cumbersome task and therefore can lead to inefficiencies.

An interesting feature of this work is the dynamism of the defense strategy. That is, the ADT is created first based on the SDG, and then updated over time, when services or vulnerabilities are added or removed. This aspect is surely appreciable, particularly in the modern systems which are characterized by a strong fluidity in their topology and configuration. Moreover, the computation of the reaction considers also the already deployed countermeasures. Still, this feature is not reflected in the experimental section, in which the authors replicate an ad-hoc attack scenario on a static network. Thus, more experiments are needed to test this promising capability and to demonstrate its feasibility.

Notably, the authors define the security performance of a given countermeasure as the number of vulnerabilities it covers multiplied by the history of its success/failure. Since multiple countermeasures can be deployed simultaneously to counteract an ongoing attack, a quite extensive study on the joint surfaces is presented. Although this methodology allows to quantitatively estimate the countermeasures' performance, it is not straightforward to define a direct correlation between vulnerabilities and countermeasures. Actually, this association method has been initially proposed in [126] and later demonstrated in [127], and overall it is proved to be a complicated task.

The presented testbed includes only 6 VMs connected by 4 virtual switches. One could argue that such a scenario does not reflect the modern IT infrastructures, where hundreds of machines are connected for providing services to the end-users. Additionally, the complexity of the designed framework is reported as $O(|CM|^2 + (|S| + |W| + |V| + |CM_a|) \times |CM|)$, where $|CM|$ represents the number of possible countermeasures, $|S|$ is the number of services in the SDG, $|W|$ is the number of time windows used in the countermeasure goodness evaluation, $|V|$ represents the number of vulnerabilities within the system, and $|CM_a|$ is the number of current deployed countermeasures. It is clear that the presented framework lacks in scalability, thus it unsuitable in the context of dynamic reaction for complex networks.

N. Summary and comparison

This Section offers a comprehensive comparison of the various works analyzed as part of this survey based on the seven features introduced in Section III-B. For easy reference, a summary of features per examined work is included in TABLE III. In addition, an overview of the surveyed works with reference to the publication year and their chief characteristics

DRAFT

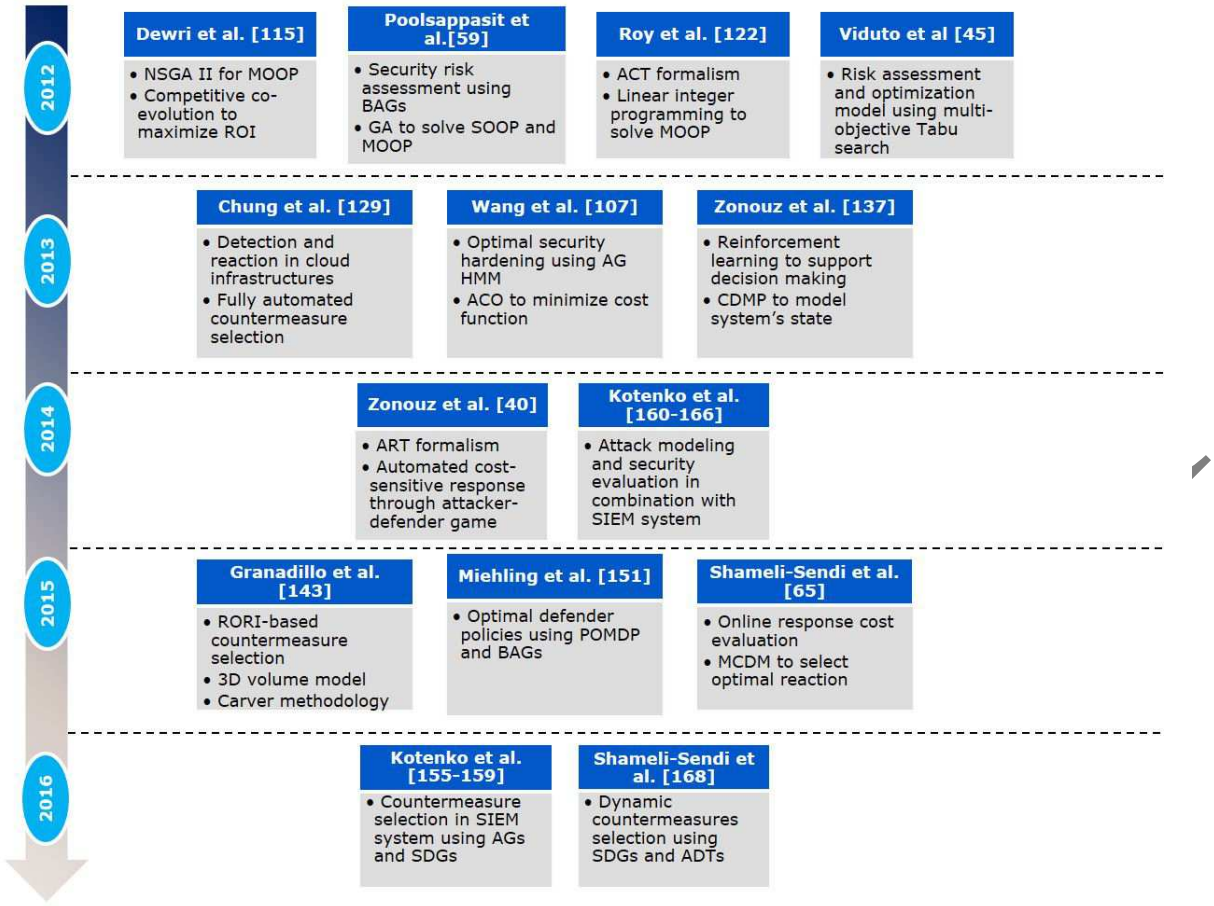


Fig. 10: A timeline of the surveyed works, highlighting on their novelty and core characteristics.

is depicted in Fig. 10. This figure summarizes the evolution of the surveyed research area, in which several authors contribute diverse reaction frameworks against cyber attacks. Finally, we elaborate on both the positive and negative aspects of each work.

Regarding the first feature, namely attack modeling decision, TABLE III reveals that all the works analyzed, except of those presented in [143] [45], make use of a graphical representation to reflect the dependencies and the interconnections among the possible assets and properties in the monitored system. The authors in [45] suggest the use of AGs for modeling the attack space, but eventually they do not integrate such a model in their solution. Likewise, the work in [143], instead of using an attack representation model, it focuses on applying countermeasures solely on one specific node representing an asset in the monitored system. From our analysis, it can be safely deduced that a representation which is able to formally model the system's numerous dependencies is a necessity both for depicting the system, but also for creating the search space of the problem. In parallel with a formal representation of the system, several probabilistic schemes are utilized in an effort to model the system's security states transitioning. In this direction, the authors adopt BAGs [59] or HMMs [107] to model the uncertainties which are introduced in the process.

Considering the numerous sources of uncertainty (e.g., IDS false alarms, possible misconfigurations, target choices of the attackers, etc.) and possible unpredictable states of the system (e.g., zero day attacks) the adoption of such a probabilistic approach contributes toward more realistic solutions that can effectively support the decision maker. However, all the reviewed works, besides those presented in [155]–[159], [168], treat the attack model representation in a static manner. Under realistic terms, a network's parameters (e.g., topology, vulnerabilities, configurations) are in a constant flux and this inevitably brings changes to the attack modeling representations as new states, nodes, and transitions occur. This ilk of changes happening in a system should be dynamically reflected in the modeling process, but unfortunately this feature is almost sure to increase the complexity of the derived models. Interestingly, none of the reviewed works considers this feature when it comes to the evaluation of the overall complexity of the framework.

Another important conclusion of our survey is that most of the analyzed works deal with the problem of cost-benefit attack counteraction by utilizing optimization techniques. This kind of solutions aim to define an optimal trade-off mainly between metrics that reflect the potential attack cost and those that quantify the impact and effectiveness of applying defense strategies. In this direction, GAs [115] and ACO [107] were utilized by leveraging single and multi-objective

optimization cost functions for providing optimal sets of countermeasures. The works presented in [40], [137], [151] make use of Bellman's optimization on top of MDPs to identify the optimal defense strategies, while integer optimization and Tabu Seach [78] were respectively used in [45], [122]. In addition, multicriteria decision-making techniques like SAW, PW and TOPSIS has been recruited in [168] in the context of MOOP. Yet, the works presented in [155]–[166], [129], [143] do not fall into this category of solutions, as the authors propose their own heuristic optimization method to guide their system to the optimal solution. The works presented in [115] and [40] are of special interest as their models reflect the dynamic relationship of the attacker and the defender. More specifically, [115] uses two competitive populations in the context of GAs that imitate an "arms race" between the two entities with the aim to define equilibrium points, that is, a set of countermeasures that can stop the attacker from increasing their gain. In the same direction, the work presented in [40] models the relationship of the attacker and the defender as a game, where the two players make sequential moves to increase their benefit. A different approach has been adopted in [65] where MCDM methodologies were used to infer on a set of possible countermeasures for eliminating the detected incidents. From this set, a Pareto optimal set is derived on a later step.

Another interesting fact stemming from TABLE III is that most of the proposed frameworks in the literature are destined to dynamically adapt to the events transpiring in the protected system. As stated in Section III, a dynamic approach fits better the needs of a countermeasure mechanism since the system is able to adapt in real time to ongoing offensive incidents. However, such a dynamic approach increases the complexity of a countermeasure mechanism, and if we additionally consider the complexity introduced by the attack modeling representation, then we can conclude that the performance of the proposed solutions is questionable. On the other hand, static solutions are designed to deliver a quantitative risk assessment for the protected system at a given time instance and to aid security administrators in diagnosing weak points on assets appearing during the life cycle of the system. Even though static solutions can be proved a valuable asset for identifying weaknesses in a network topology, the lack of adaptability against unfolding offensive incidents is not suitable to support a reactive countermeasure system.

In our analysis, we have distinguished the type of reaction (dynamic/static) from the automation level which actually reflects the administrators' engagement level in the reaction process. As it can be observed from TABLE III, only the works in [107] and [115] require the manual engagement of the administrator, while the rest of them provide a higher level of automation. The majority of the latter belong to the semi-automated category, where the administrators supervise the countermeasure system and the enforcement of any defensive policy requires their approval before applied on the protected system. Still, the works in [129], [151], [168] proposed a fully-automated system, in which the defense mechanisms react in an autonomous manner requiring minor intervention by the administrators. More specifically, the administrators assign predefined directives that have to be followed by the countermeasure system. The proposed system in [168] can be characterised as fully-automated, but the administrator has to identify the services' dependencies and maintain the SDG of the system. Overall, although fully-automated solutions can respond immediately against ongoing attacks, the applied decisions may differ from a security expert's perspective, resulting to unwanted after-effects like the over or under-protection of network assets.

All the analyzed works adopt assessment methodologies to provide a quantitative analysis of the defense strategy they propose, but as we can observe from TABLE III, this is being done mainly without adopting any security standards. A model which is able to quantify the trade-off between the attack versus defense cost in applying countermeasure policies is a vital feature for the countermeasure strategy. In this direction, as discussed in Section III-B, several methodologies that engage and combine diverse types of metrics have been proposed. However, it is notable that even though those methodologies can provide a quantitative security assessment, half of the surveyed works do not adopt any globally accepted standard. We argue that a countermeasure strategy should adopt the use of standards, so that security administrators can perceive in an accurate and foolproof way the security state of the system at any time. In addition, the adoption of standards ensures interoperability and enables a countermeasure mechanism to operate across diverse systems. As it can be observed in TABLE III, CVSS [105] is the most prominent standard used for describing the threats in a system. Even though, CRE [93] and CAPEC [92] are scarce among the used standards, we argue that they can not only significantly contribute in the automation of countermeasure solutions, but also improve the accuracy in predicting the security state of a system.

It is important to analyze the way the various works evaluate the effectiveness of their proposal and specifically the ways a security administrator (who relies on a countermeasure system) can assess the outcome of the provided defensive actions. To do so, we included in TABLE III the scale and the type of the environments used in the evaluation of the examined frameworks. We also report on the role of the administrator to reflect the way they interact with the system. As it can be observed from the table, 10 out of a total of 14 works ([115], [59], [122], [129], [107], [40], [143], [151], [155]–[159], [168]) rely on a rather small-scale environment to evaluate their solution, while 3 others ([45], [137], [65]) employed a medium-scale one. The sole work that considers a large-scale testbed is that in [160]–[166]. It is also notable that only the works in [129] and [168] utilized a real environment instead of a simulation. The role of the administrator in most of the surveyed works is to tune the system, while only in [137] this entity is in charge to respond by giving feedback. Also, the exact role of the administrator was not appreciable in contributions [115], [129], [143], and [151].

Continuing on the performance criterion, the works presented in [137] and [115] are of special interest because of the methods used to ensure the optimality of the result. Precisely, the authors in [137] adopt a reinforcement learning approach, where the administrator is able to give a feedback to the system

DRAFT

and boost the learning process to more accurate results. On the other hand, the framework proposed in [115] ensures the effectiveness of the defense policy as it tests the latter against several attack strategies based on the "arms race" method. The goal is to define a policy that will impede the attacker from penetrating further into the protected system. In the works presented in [40], [107], [160]–[166], [155]–[159], [59], [65], [122], [45] the administrator tunes the process of providing optimal countermeasures by setting objectives to the system. In [168] on the other hand, the administrator needs to update the dependences among the system's services. These objectives are represented in an abstract way, either by specific trade-offs in system's metrics or IF-THEN rules, which are used to infer on the security level of the system.

Finally, it is of significant importance to elaborate on the performance of the proposed frameworks in terms of scalability, time complexity, and response time.

Scalability: As observed from TABLE III, the majority of the reported works are characterized by a low scalability due to the inability of the attack modeling representations to scale. This feature seems to be a substantial limitation in the field. As already mentioned, this shortcoming becomes a major one if we consider that none of the reported works but those in [155]–[159], [168], consider dynamic changes in the network parameters. Moreover, even though the authors in [155]–[159], [168] consider a dynamically changing environment, this is not advocated by their experiments, as no change occurs in the employed topologies. However, according to their authors, the works presented in [40], [129], [137] have a high scalability. In fact, this is why the solutions in [40], [137] incorporate distributed architectures which disseminate the computational task to the edge nodes instead of dealing with the problem solely in a central point. Also, it can be said that the work in [137] presents a high scalability as it incorporates CMDP to the attack modeling phase. Although this approach looks potent, the level of information about the security state of the system, which can be reflected by the CMDP representation, is debatable. In addition, as with the vast majority of the surveyed works, this solution is tested in a small-scale simulated environment and therefore its overall complexity in a real environment might be considerably higher.

Time Complexity: Regarding the complexity sub-criterion, half of the reported solutions present a high complexity, while the rest can be classified into low or medium regarding the same metric. Unfortunately, the fact that the vast majority of the proposals were tested under simulated and small-scale environments raises questions about their performance in a real networking environment. In addition, another parameter that can augment the overall complexity is the number of countermeasure actions that can be applied in each particular case. In fact, the reviewed works engage only a small fraction of the possible countermeasures. The existence of several possible countermeasures for every possible security state of the system can lead to a search space explosion and challenge the optimization algorithms. On the contrary, the works proposed in [40], [129] adopt a different approach, as they model the system in a distributed manner, so that the computational cost is not undertaken solely by a specific machine. However, these solutions introduce other limitations inherited from the distributed environments, including trust issues and the lack of accurate insight of the system's state at a central node in the architecture.

Response time: It is obvious from TABLE III that it was practically infeasible to extract the response time for all the surveyed works. This is because 6 of them ([115], [59], [129], [143], [151], [155]–[159]) do not provide details about the response time of the framework they propose when activating the appropriate countermeasures. This is actually a controversial discovery as the effectiveness of a response system is also determined by its ability to react within a reasonable time frame. Especially for dynamic proposals, which have to respond instantly against ongoing incidents, this metric is of high significance. To sum up, only the works in [65], [137], [168] were classified as fast-performing solutions, while 3 presented average response time ([122], [45], [40]) and 2 seem to be slower regarding this metric ([107], [160]–[166]). At this point, we have to note that during the evaluation none of the works considered the time needed to create the attack modeling representation, while the small-size simulated environments cannot guarantee the preciseness in response time estimation.

All in all, it can be safely summarized that virtually all the so far proposed solutions in the field of countermeasures elicitation suffer from scalability issues due to the bulky attack modeling representations, the numerous possible countermeasures, and the dynamic nature of the monitored systems.

DRAFT

TABLE III: Side-by-side comparison of the surveyed works based on the features presented in Section III-B

Surveyed Works	Features for countermeasures strategy solutions								
	Attack modeling	Countermeasures provision techniques	Outcomes assessment	Type of reaction	Used standards	Automation level	Performance		
							Scalability	Complexity	Response
(2012) Dewri et al. [115], Section IV-A	• Augmented Attack Trees	• Genetic Algorithm for single & multi-objective optimization • Attacker-Defender "Arms Race"	• Test bed: Small scale simulation • Admin's role: N/A	Static	No	Manual	Low	High	N/A
(2012) Poolsappasit et al. [59], Section IV-B	• Bayesian Attack Graphs	• Genetic algorithm for single & multi-objective optimization	• Test bed: Small scale simulation • Admin's role: Tuning	Static	CVSS	Semi-automated	Low	High	N/A
(2012) Roy et al. [122], Section IV-C	• Attack Countermeasure Trees	• Branch & Bound integer optimization algorithm	• Test bed: Small scale simulation • Admin's role: Tuning	Static	No	Semi-automated	Med	Low	Avg
(2012) Viduto et al. [45], Section IV-D	• No attack modeling. An attack model could be applied	• Multi-objective tabu search	• Test bed: Medium scale simulation • Admin's role: Tuning	Static	CVE	Semi-automated	N/A	Low	Avg
(2013) Chung et al. [129], Section IV-E	• Attack Graphs	• Heuristic optimization method	• Test bed: Small scale real virtual network • Admin's role: N/A	Both	CVSS, CVE NVD	Fully-automated	High	Low	N/A
(2013) Wang et al. [107], Section IV-F	• Attack Graphs • Hidden Markov Model	• Ant Colony Optimization	• Test bed: Small scale simulation • Admin's role: Tuning	Static	NVD, CVSS	Manual	Med	Med	Slow
(2013) Zonouz et al. [137], Section IV-G	• Network connectivity matrix • Competitive Markov decision process	• Bellman's optimization method	• Test bed: Medium scale simulation • Admin's role: Feedback	Both	No	Semi-automated	High	Med	Fast
(2014) Zonouz et al [40], Section IV-H	• Attack Response Trees • Partially observable Markov decision Process • Game theory two- player Stackelberg stochastic game	• Bellman's optimization method	• Test bed: Small scale simulation • Admin's role: Tuning	Dynamic	No	Semi-automated	High	N/A	Avg
(2015) Granadillo et al. [143], Section IV-I	• No attack modeling. Static allocation of attacker using network evidence	• Heuristic RORI-based optimization method	• Test bed: Small scale simulation • Admin's role: N/A	Dynamic	No	Semi-automated	Low	High	N/A
(2015) Miehling et al. [151], Section IV-J	• Partially observable Markov decision Process • Bayesian Attack Graphs	• Bellman's optimization method	• Test bed: Small scale simulation • Admin's role: N/A	Dynamic	No	Fully-automated	Low	High	N/A
(2015) Shameli-Sendi et al. [65], Section IV-K	• Attack Graphs • Service Dependency Graphs	• Simple Additive Weighting and Weighted Product MCDM • Pareto optimality	• Test bed: Medium scale simulation • Admin's role: Tuning	Dynamic	CVE	Semi-automated	Low	Low	Fast
(2013-2016) Kotenko coutermeasure selection in SIEM [155]–[159], Section IV-L1	• Attack Graphs • Service Dependency Graphs	• Heuristic optimization method	• Test bed: Small scale simulation • Admin's role: Tuning	Both	SCAP : CRE, ERI, CVE, CVSS, CWE, CPE	Semi-automated	Low	High	N/A
(2012-2014) Kotenko attack modeling [160]–[166], Section IV-L2	• Attack Graphs • Service Dependency Graphs	• Heuristic optimization method	• Test bed: Large scale simulation • Admin's role: Tuning	Both	CPE, CVE, CVSS, CAPEC	Semi-automated	Low	High	Slow
(2016) Shameli-Sendi et al. [168], Section IV-M	• Attack Defense Trees • Service Dependency Graphs	• Simple Additive Weighting • Pareto optimality	• Test bed: Small scale real virtual network • Admin's role: Tuning	Dynamic	CVE	Fully-automated	Low	Medium	Fast

DRAFT

V. Research Challenges and Future Directions

This Section builds on top of the previous one by detailing on the challenges in the area of countermeasure strategies against cyber attacks. The discussion revolves around 6 factors that, according to the conducted survey, seem to be the most prominent in the development of solutions in this particular area.

A. Scalability

As it is shown in TABLE III, one of the main limitations of the analyzed works is the poor scalability of the proposed solutions. Only a few of them propose an approach which is scalable, in the sense that complexity does not increase exponentially with the number of parameters included in the attack model. In fact, this characteristic is clearly reflected in the pilot implementations of the described works; one can easily notice that there is a lack in including an experimental section which assesses the feasibility of the solution on a large-scale environment in terms of number of hosts and interconnections. It is therefore clear that the presented implementations can be seen only as prototypes and they cannot reliably reflect the size and complexity of real-life networks.

We identify the attack modeling as the main cause of this deficiency. That is, most of the reported works use AGs and ATs as the referring model for exploring possible paths, which may be selected by an attacker. As stated in Section III-B1, these representations are widely used by the research community to model the attacker's steps, because they are able to reveal the cause-consequence relationship among the represented nodes of the graph, taking into account some elemental parameters of the network, including its topology, connections, vulnerabilities, and so forth. The described features are really useful in case of multi-step attacks, where the ability of predicting the attacker's pathway has a key role. However, the size of these representations becomes quickly unmanageable as the size of the network grows and the interconnections among the nodes become denser. Moreover, alongside with the graphical representation of the attacker's steps, a computation of the paths' probability must be executed. However, this task increases the overall complexity of the attack modeling.

Some methodologies, like that in [56], do attempt to reduce this limitation. One obvious improvement can be achieved by building the graphs *off-line*, and then update them when the input parameters change. In this way, the graphs do not have to be generated continuously, and the saved resources can be allocated for the analysis phase [59]. However, this process does not consider that, especially for dynamic environments, there is a high probability that the input used to build the graph (i.e., network topology, asset vulnerabilities, and others) may frequently change over time. In this case, the graph generation cannot be considered as a one-time cost. That is, a dynamic procedure is needed to update this representation by adding or removing nodes without regenerating the graph from scratch.

Another possible amelioration in the analysis of the graphs is the usage of special algorithms which are able to decrease the analysis complexity and calculate the path probabilities faster [170], [171]. The contribution of these algorithms lies mainly in a reduction of the search time in graphs, thus it can be safely argued that the application of these methodologies is suitable for this kind of problems. Given the evidence of an intrusion on a graph node, all the connected paths must be extracted. Besides this, the probabilities assignment on the paths has to be computed in an efficient way, so as to make the graph computation affordable for dynamic scenarios.

Perhaps, this inherent scalability issue can be solved if looking at it from a broader perspective, that is, by focusing on all the development stages of the reaction system:

- At the design stage, the system should be arranged as a distributed architecture, which scales better for complex networks.
- At the implementation stage, more efficient analysis algorithms should be used for the purpose of calculating faster the likelihoods and the possible connections of the attacker paths, and predicting their next steps.

B. Countermeasure knowledge

Another shortcoming that emerged during this study is the lack regarding the *countermeasure knowledge*. Each one of the surveyed works presents only a limited set of countermeasures, which is used to counteract specific attacks reported to the monitored system. In this way, the selection process loses a great part of its importance and effectiveness, because the usage of an optimization technique on a limited search space is not advantageous, but rather it increases the algorithmic complexity.

In our opinion, a comprehensive pool of countermeasures is a *sine qua non* for any reactive system. This pool must contain atomic actions (or a combination of them) which can be undertaken to block and eradicate possible attacks. By doing so, the number of possible choices for the defender to react will be considerable, giving them two main advantages: (a) the possibility to defeat more potential intrusions, and (b) the potential to select the most appropriate countermeasure (or a set of them), which satisfies the required trade-off between cost, impact, and effectiveness of the hardening measures. Moreover, with a large pool of countermeasures, the use of an optimization algorithm is sure to offer its own advantages, giving its ability to react based on a combination of countermeasures.

Nevertheless, there is a need for reliable sources of information to build this knowledge. One possible solution is to use security administrators' expertise. As already pointed out, we do consider the important role that security administrators should play in this strategy. These people have certain budget constraints, so the selection of countermeasures cannot exclude their analysis and approval. Moreover, the task to define and control the trade-off between effectiveness and cost of the countermeasures is assigned to them, as they represent in this vision a central point in which the flow of knowledge must pass through. Having in mind these considerations, a security expert should not be the *only* source of knowledge; rather, they should directly act only in critical situations, and provide feedback to the countermeasure selection process.

As described in Section III-B5, another possible way to tackle this problem is to use open standard platforms. The

DRAFT

CVE database [50], for example, presents a list of known vulnerabilities which have been discovered in computer systems during the recent years. The format of a CVE entry provides a *reference* field, which normally provides a link to an HTML page describing the problem and, in most cases, possible workarounds. Following this methodology, a translator, say, in the form of a software gateway can be used for acquiring knowledge in text format and transform it into another that is understandable by the underlying machines [172]. Extending this concept to more vulnerability platforms of this kind, one can anticipate the acquisition of a nearly exhaustive knowledge of countermeasure solutions to perform an accurate and successful reaction.

C. Standard representation

Another deficiency which arose during the study of the various works consists in the absence of a *standard representation* for the countermeasures. This issue is directly linked with the previous one, meaning that with the presence of a common and shared reaction intelligence, a standard format which represents the counteractions will be greatly appreciated.

Following the same reasoning exposed for the countermeasure knowledge, we consider a standard countermeasures representation as a key feature in this context, as it can enable essential information sharing among the different actors, thus leading to an increase in the effectiveness of the implemented actions in the mid or long-run.

It is true that so far some attempts have been done towards this direction. An example is the *Common Remediation Enumeration* (CRE) [93], which is a component of the SCAP [108]. A CRE entry is a set of properties that describe a specific remediation instance, including a single configuration setting change, the application of a patch, installation/deinstallation of software, a system reboot, and many others. Specifically, a CRE entry has the following parameters:

- A *CRE-ID*, a textual identifier which can be used to uniquely name a single remediation instance.
- A textual, human-understandable *description* of the entry detailing the method and the effect of the remediation instance;
- An optional list of *parameters* applicable to the entry with specific constraints.
- The platform on which the entry is valid, expressed as a *Common Platform Enumeration Applicability Language* expression. The latter defines a standardized way to describe IT platforms by forming complex logical expressions out of individual CPE names and references [173].
- Supporting *references* and *metadata* related to the CRE entry.

A first draft of the CRE was given in 2011, but it was discontinued as of March 2016 for reasons not publicly known. Obviously, more research is needed to better appreciate the helpfulness of this representation, and to find ways in which it can be exploited for maximizing its benefits.

D. Correlation between countermeasures and attacks

Once a comprehensive pool of countermeasures has been created, a thorough study on the *correlation* between atomic reaction steps and attacks is also needed. This is another arduous aspect regarding the countermeasure strategy, especially when considering the great mass and the complexity of possible attacks. Until now, a handful of attempts have been conducted to solve this issue in [45], [127], but all of them require a great effort from the security expert. This is because this kind of solutions rely on administrator's knowledge about each threat. Hence, a systematic methodology is needed toward intertwining the available attack set with the appropriate countermeasures.

The first action to achieve such a correlation is to study per given countermeasure the number (or category) of attacks it can cover. For example, let us consider the countermeasure "*block a certain range of IP addresses*". This can be easily done by adding a specific access control list in the FW. Also, such a rule is able to defeat different categories of attacks, including probing, DoS and bruteforce, among others.

Then, an analysis needs to be conducted on the effect of the combined countermeasures against a specific attack. This requirement is needed for complex attacks, when applying a single countermeasure may be not enough, or when a combination of security measures is preferred because their execution is more effective or more convenient as the case may be. For instance, suppose that an information leakage is detected in a specific machine in the network. The first action which should be undertaken is to remove all the privileges assigned to that machine in order to block the malicious activity. However, this may be not enough to stop the intrusion; the machine should be also isolated by blocking its network connections.

Once the *one-to-many* relationships between countermeasures and attacks have been constructed, the logical sequel is to build a *many-to-many* relationship between them. This interlinking should consider also the *context* in which the countermeasures are applied. This means that a specific reaction can be useful in a specific context, but ineffective or useless if it is applied in another scenario.

E. Metrics and scoring system

As discussed in Section III-B5, the use of metrics is needed to *quantitatively* analyze the experimental results provided by each of the surveyed works, where applicable. So far, in the literature, a plethora of security metrics has been proposed, with the aim of capturing different aspects of the problem, including attacks, networks topologies, cost and vulnerabilities metrics [73]. Nevertheless, from our study, we realized that there is a prominent lack of specific and commonly used measurement systems for reliable countermeasure assessment. While the application of a specific countermeasure is for blocking an attack, it may also involve side effects, which should be considered and quantified for optimizing the whole process.

DRAFT

As already pointed out in Section III-B5, currently a scoring system for the vulnerabilities has been developed by the *Forum of Incident Response and Security Teams* (FIRST) [105].

In this mindset, the creation of a countermeasure scoring system is highly desirable. An important aspect to consider in this regard is the possibility of adapting the countermeasure score depending on temporal and environmental aspects, as CVSS does for the vulnerabilities. That is, as a vulnerability evolves during time and changes its impact depending on the system where it is applied, also a countermeasure should be considered as an *evolving entity* which, depending on the above factors, updates its score.

From the survey of works conducted in the context of this paper, a countermeasure scoring system should consider the following parameters:

- *Effectiveness* of the solution. It is expressed as a percentage of coverage or a probability of success.
- *Scope* of the reaction. It is expressed as the ability of impacting other components in the system, which could be directly or indirectly affected by its enforcement.
- *Maturity* of the solution. It reflects the elapsed time from which the solution has been deemed as functional.
- *Impact* on the system. It is expressed as a function of *availability*, *confidentiality* and *integrity* impact of the countermeasure on the ICT system.
- *Implementation cost*. It is part of the direct costs of the countermeasure, regarding its activation in the system.
- *Maintenance cost*. It represents the economic cost to sustain the implementation of a countermeasure for long-term reactions; it is calculated only for the long-term countermeasures.
- *Indirect cost*. It reflects the collateral economic damage caused by a countermeasure, as the effect of the countermeasure on a legitimate user of the system, the deactivation cost for a particular invasive remediation, and so forth.

Considering the possible adjustments using temporal and environmental aspects, the score assigned to a given countermeasure can be considered as *adaptable*, solving in this way also the lack of adaptability existing in the analyzed works. Once a suitable measurement system has been developed, a testing phase to judge upon its capabities is desiderable. In this mindset, the construction of a full-fledged benchmark dataset containing both single and multi-step attacks based on real network data would be a great pointer for future research in this field. The proposal of Shiravi et al. [174] can be of great help in this direction as it introduces a set of guidelines on how to build valid datasets, which can be followed to create new ones.

F. Mitigating zero-day attacks

A last research challenge extracted from the survey contucted in Section IV is the mitigation of zero-day attacks. The dissection of the works reveals a notable shortcoming of the literature, as the vast majority of the proposed countermeasure provision solutions neglect any kind of reaction against unknown offensive incidents. Even though the authors in [115], [137], [156], [160], [163] elaborate on the problem of responding upon zero-day attacks, their implementations and the experimental testbeds advocate that they cannot be considered as concrete solutions. Without doubt, detecting attacks derived from zero-day vulnerabilities is a challenging task. One could say that counteracting such attacks is even more challenging. In fact, until a fix is published to patch the zero-day vulnerability, the corresponding systems remain unprotected. This gives additional value to a countermeasure system which is able to provide cost-benefit mitigation strategies, and to a comprehensive static countermeasure planning.

The inefficiency of the current proposals stems mainly from the fact that the attack modelling representations are capable of representing specifically defined states of the system, thus omitting attack paths that could appear under zero-day vulnerabilities. POMDP and HMM representations have been used in [5], [151] respectively to quantify uncertainties concerning the attacker's position or their chosen attack path, but not to model the potential existence of unknown system's security states. To this end, such models could be proved beneficial in conjunction with zero-day vulnerability metrics [5], [167] and security automation standards such as CWE [88]. Additionally, an approach of generating AGs by taking into account zero-day vulnerabilities [160] sounds very promising for building countermeasure systems in this direction. Without doubt, the effectiveness and responsiveness of such a system constitute an additional research challenge.

VI. Conclusions

The convergence of network technologies around IP and the openness to the Internet and IoT, present major challenges from a security viewpoint. Today, more than ever, organizations are facing a plethora of highly diversified cyber attacks, which tend to be more ingenious and decisive. In this highly offensive and dynamic terrain, the need for full-blown, fine-grained, and possibly automated reaction strategies in terms of optimal countermeasure selection is highly demanded and urgently needed. In fact, this necessity is observed in the recent literature of ICT security by a number of works published in well-respected journals and conferences during the last 5 years. In this context, the paper at hand is the first to our knowledge to offer a comprehensive study of these works, fulfilling the following three goals. First, it extracts common criteria that can be used as a basis for comparing the various existing (and future) works in this evolving field. Second, it delves into each of the surveyed works, and through a critical discussion, pinpoints its advantages and disadvantages. This, in synergy with the identified criteria, leads to a comprehensive side-by-side comparison of the included works and helps the reader to obtain a holistic view of this particular field. Last, it elaborates on the future research directions and challenges in this topic, which can be used as a reference to anyone interested in grasping the diverse facets of this area of research. We anticipate that the current work will foster the development of well-designed reaction frameworks and/or strategies capable of nipping cyber attacks in the bud in an effective and cost-efficient manner.

DRAFT

As future directions, we will investigate on the development of a methodology which is capable of addressing the above-mentioned research challenges.

ACKNOWLEDGMENT

This work has been supported by the European Commission Horizon 2020 Programme under grant agreement number H2020-ICT-2014-2/671672 - SELFNET (*Framework for Self-Organized Network Management in Virtualized and Software Defined Networks*), by the Spanish MICINN (project DHARMA, *Dynamic Heterogeneous Threats Risk Management and Assessment*, with code TIN2014-59023-C2-1-R), by the European Commission (FEDER/ERDF), by a Ramón y Cajal research contract (RYC-2015-18210) granted by the MINECO (Spain) and co-funded by the European Social Fund, as well as by a Leonardo Grant 2017 for Researchers and Cultural Creators awarded by the BBVA Foundation.

APPENDIX
ACRONYMS

Acronym	Reference abbreviation
ACO	Ant Colony Optimization
ACT	Attack Countermeasure Tree
ADT	Attack Defense Tree
AG	Attack Graph
ART	Attack Response Tree
AT	Attack Tree
AV	Antivirus
BAG	Bayesian Attack Graph
CIA	Confidentiality Availability and Integrity
CMDP	Competitive Markov Decision Process
FTP	File Transfer Protocol
FW	Firewall
GA	Genetic Algorithm
HMM	Hidden Markov Model
ICT	Information and Communications Technology
IDS	Intrusion Detection System
IPS	Intrusion Prevention System
IRS	Intrusion Response System
MCDM	Multi-Criteria Decision Making
MDP	Markov Decision Process
MOOP	Multi-Objective Optimization Problem
POMDP	Partially Observable Markov Decision Process
POCMDP	Partially Observable Competitive Markov Decision Process
ROI	Return-On-Investment
RORI	Return-On-Response-Investment
RRE	Response and Recovery Engine
SDG	Service Dependency Graph
SIEM	Security Information and Event Management
SLA	Service Level Agreement
SOOP	Single-Objective Optimization Problem
VM	Virtual Machine

REFERENCES

[1] Juniper Research, "'Internet of Things' connected devices to almost triple to over 38 billion units by 2020," http://www.juniperresearch.com/press/press-releases/iot-connected-devices-to-triple-to-38-bn-by-2020, July 28th 2015, accessed: 2016-09-15.

[2] S. Castano and V. D. Antonellis, "Global viewing of heterogeneous data sources," *IEEE Transactions on Knowledge and Data Engineering*, vol. 13, no. 2, pp. 277–297, Mar 2001.

[3] A. McAfee, E. Brynjolfsson, T. H. Davenport, D. Patil, and D. Barton, "Big data," *The management revolution. Harvard Bus Rev*, vol. 90, no. 10, pp. 61–67, 2012.

[4] C. Kolias, G. Kambourakis, A. Stavrou, and S. Gritzalis, "Intrusion detection in 802.11 networks: Empirical evaluation of threats and a public dataset," *IEEE Communications Surveys and Tutorials*, vol. 18, no. 1, pp. 184–208, 2016. [Online]. Available: http://dx.doi.org/10.1109/COMST.2015.2402161

DRAFT

[5] L. Wang, S. Jajodia, A. Singhal, and S. Noel, "K-zero day safety: Measuring the security risk of networks against unknown attacks," in *Proceedings of the 15th European Conference on Research in Computer Security*, ser. ESORICS'10. Berlin, Heidelberg: Springer-Verlag, 2010, pp. 573–587.

[6] A. Lenin, J. Willemson, and D. P. Sari, *Attacker Profiling in Quantitative Security Assessment Based on Attack Trees*. Cham: Springer International Publishing, 2014, pp. 199–212.

[7] P. García-Teodoro, J. Díaz-Verdejo, G. Maciá-Fernández, and E. Vázquez, "Anomaly-based network intrusion detection: Techniques, systems and challenges," *Computers & Security*, vol. 28, no. 1-2, pp. 18 – 28, 2009.

[8] Symantec, "Internet security threat report | government," https://www.symantec.com/content/dam/symantec/docs/reports/istr-21-2016-government-en.pdf, April 2016, accessed: 2016-09-21.

[9] S. Greco, J. Figueira, and M. Ehrgott, "Multiple criteria decision analysis: State of the art surveys," *Springer's International series*, 2005.

[10] A. Fielder, E. Panaousis, P. Malacaria, C. Hankin, and F. Smeraldi, "Decision support approaches for cyber security investment," *Decision Support Systems*, vol. 86, pp. 13 – 23, 2016.

[11] B. Schneier, "Attack trees," *j-DDJ*, vol. 24, no. 12, pp. 21–22, 24, 26, 28–29, Dec. 1999.

[12] D. O. Díaz López, G. Dólera Tormo, F. Gómez Mármol, and G. Martínez Pérez, "Dynamic counter-measures for risk-based access control systems: an evolutive approach," *Future Generation Computer Systems*, vol. 55, pp. 321–335, 2016. [Online]. Available: http://dx.doi.org/10.1016/j.future.2014.10.012

[13] Y. Yu, K. Li, W. Zhou, and P. Li, "Trust mechanisms in wireless sensor networks: Attack analysis and countermeasures," *Journal of Network and Computer Applications*, vol. 35, no. 3, pp. 867 – 880, 2012, special Issue on Trusted Computing and Communications.

[14] O. Kosut, L. Jia, R. J. Thomas, and L. Tong, "Malicious data attacks on smart grid state estimation: Attack strategies and countermeasures," in *Smart Grid Communications (SmartGridComm), 2010 First IEEE International Conference on*, Oct 2010, pp. 220–225.

[15] C. Kolias, G. Kambourakis, and S. Gritzalis, "Attacks and countermeasures on 802.16: Analysis and assessment," *IEEE Communications Surveys and Tutorials*, vol. 15, no. 1, pp. 487–514, 2013. [Online]. Available: http://dx.doi.org/10.1109/SURV.2012.021312.00138

[16] N. Stakhanova, S. Basu, and J. Wong, "A taxonomy of intrusion response systems," *International Journal of Information and Computer Security*, vol. 1, no. 1-2, pp. 169–184, 2007.

[17] A. Shameli-Sendi, M. Cheriet, and A. Hamou-Lhadj, "Taxonomy of intrusion risk assessment and response system," *Computers & Security*, vol. 45, pp. 1 – 16, 2014.

[18] A. Shameli-Sendi, N. Ezzati-Jivan, M. Jabbarifar, and M. Dagenais, "Intrusion response systems: survey and taxonomy," *Int. J. Comput. Sci. Netw. Secur*, vol. 12, no. 1, pp. 1–14, 2012.

[19] F. Gómez Mármol, M. Gil Pérez, and G. Martínez Pérez, "I don't Trust ICT: Research challenges in Cyber Security," in *10th IFIP WG 11.11 International Conference on Trust Management (IFIPTM 2016)*, ser. IFIP AICT, vol. 473, Darmstadt, Germany, 2016, pp. 129–136.

[20] NATO Cooperative Cyber Defence Centre of Excellence, "Cyber definitions," https://cedcoe.org/cyber-definitions.html, accessed: 2016-09-14.

[21] OPNsense, "open source firewall," https://opnsense.org/, accessed: 2016-09-20.

[22] "pfSense open source firewall," https://pfsense.org/, accessed: 2016-09-20.

[23] "HID secure identity solutions," https://www.hidglobal.com/, accessed: 2016-09-14.

[24] "Mcafee network security platform," http://www.mcafee.com/us/products/network-security-platform.aspx, accessed: 2016-09-14.

[25] "Cisco next generation intrusion prevention system (NGIPS)," http://www.cisco.com/c/en/us/products/security/ngips/index.html, accessed: 2016-09-14.

[26] J. Magdych, T. Rahmanovic, J. McDonald, B. Tellier, A. Osborne, and N. Herath, "Network-based risk-assessment tool for remotely detecting local computer vulnerabilities," Aug. 22 2006, uS Patent 7,096,503.

[27] K. Haslum, A. Abraham, and S. Knapskog, "DIPS: A framework for distributed intrusion prediction and prevention using hidden markov models and online fuzzy risk assessment," in *Third International Symposium on Information Assurance and Security*, Aug 2007, pp. 183–190.

[28] "Nessus vulnerability scanner," http://www.tenable.com/products/nessus-vulnerability-scanner, accessed: 2016-09-14.

[29] H. Holm, T. Sommestad, J. Almroth, and M. Persson, "A quantitative evaluation of vulnerability scanning," *Information Management & Computer Security*, vol. 19, no. 4, pp. 231–247, 10 2011.

[30] P. K. Manadhata and J. M. Wing, "An attack surface metric," *IEEE Transactions on Software Engineering*, vol. 37, no. 3, pp. 371–386, May 2011.

[31] D. Miller, S. Harris, A. Harper, S. VanDyke, and C. Blask, *Security information and event management (SIEM) implementation*. McGraw Hill Professional, 2010.

[32] "Snort, network intrusion detection & prevention system," https://snort.org/, accessed: 2016-08-26.

[33] "Suricata, open source IDS/IPS/NSM engine," https://suricata-ids.org/, accessed: 2016-09-14.

[34] "OSSEC, open source HIDS security," http://ossec.github.io/, accessed: 2016-09-12.

[35] "Bro, network security monitor," https://www.bro.org/, accessed: 2016-09-14.

[36] D. J. Weller-Fahy, B. J. Borghetti, and A. A. Sodemann, "A survey of distance and similarity measures used within network intrusion anomaly detection," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 1, pp. 70–91, 2015.

[37] J. Pescatore and G. Young, "Defining the next-generation firewall," *Gartner RAS Core Research Note*, 2009.

[38] M. Gunasekharan, S. Basu, and G. R. Santhanam, "Selecting the minimal set of preferred responses to counter detected intrusions," in *Proceedings of the 12th Annual Conference on Cyber and Information Security Research*. ACM, 2017, p. 5.

[39] I. ISO and I. Std, "Iso 27002: 2005," *Information Technology-Security Techniques-Code of Practice for Information Security Management. ISO*, 2005.

[40] S. A. Zonouz, H. Khurana, W. H. Sanders, and T. M. Yardley, "RRE: A game-theoretic intrusion response and recovery engine," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 2, pp. 395–406, Feb 2014.

[41] N. Stakhanova, S. Basu, and J. Wong, "A cost-sensitive model for preemptive intrusion response systems," in *Proceedings of the 21st International Conference on Advanced Networking and Applications*, ser. AINA '07. Washington, DC, USA: IEEE Computer Society, 2007, pp. 428–435.

[42] "LogRhythm SIEM," https://logrhythm.com/products/security-intelligence-platform/, accessed: 2016-09-14.

[43] "Gartner, magic quadrant for security information and event management SIEM," https://www.gartner.com/doc/reprints?id=1-2JM4RVZ&ct=150720&st=sb, accessed: 2016-09-23.

[44] T. Toth and C. Kruegel, "Evaluating the impact of automated intrusion response mechanisms," in *Computer Security Applications Conference, 2002. Proceedings. 18th Annual*, 2002, pp. 301–310.

[45] V. Viduto, C. Maple, W. Huang, and D. López-Peréz, "A novel risk assessment and optimisation model for a multi-objective network security countermeasure selection problem," *Decis. Support Syst.*, vol. 53, no. 3, pp. 599–610, Jun. 2012.

[46] E. S. Pilli, R. C. Joshi, and R. Niyogi, "Network forensic frameworks: Survey and research challenges," *Digit. Investig.*, vol. 7, no. 1-2, pp. 14–27, Oct. 2010.

[47] "AlienVault Open Threat Exchange (OTX)," https://www.alienvault.com/open-threat-exchange, accessed: 2016-09-14.

[48] P. Fogla, M. Sharif, R. Perdisci, O. Kolesnikov, and W. Lee, "Polymorphic blending attacks," in *Proceedings of the 15th Conference on USENIX Security Symposium - Volume 15*, ser. USENIX-SS'06. Berkeley, CA, USA: USENIX Association, 2006.

[49] R. Dewri, N. Poolsappasit, I. Ray, and D. Whitley, "Optimal security hardening using multi-objective optimization on attack tree models of networks," in *Proceedings of the 14th ACM Conference on Computer and Communications Security*, ser. CCS '07. New York, NY, USA: ACM, 2007, pp. 204–213.

[50] "CVE, common vulnerabilities and exposures," https://cve.mitre.org/, accessed: 2016-08-18.

[51] "Archimate, architecture description language," http://www.opengroup.org/subjectareas/enterprise/archimate, accessed: 2016-09-23.

[52] P. C. Clements, "A survey of architecture description languages," in *Proceedings of the 8th International Workshop on Software Specification and Design*, ser. IWSSD '96. Washington, DC, USA: IEEE Computer Society, 1996, pp. 16–.

[53] A. Schilling and B. Werners, "Optimal selection of it security safeguards from an existing knowledge base," *European Journal of Operational Research*, vol. 248, no. 1, pp. 318–327, 2016.

DRAFT

[54] K. Durkota, V. Lisy, C. Kiekintveld, and B. Bosansky, "Game-theoretic algorithms for optimal network security hardening using attack graphs," in *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems*. International Foundation for Autonomous Agents and Multiagent Systems, 2015, pp. 1773–1774.

[55] B. Kordy, L. Piétre-Cambacédés, and P. Schweitzer, "DAG-based attack and defense modeling: Don't miss the forest for the attack trees," *Computer Science Review*, vol. 13-14, pp. 1 – 38, 2014.

[56] X. Ou, W. F. Boyer, and M. A. McQueen, "A scalable approach to attack graph generation," in *Proceedings of the 13th ACM Conference on Computer and Communications Security*, ser. CCS '06. New York, NY, USA: ACM, 2006, pp. 336–345.

[57] K. Ingols, M. Chu, R. Lippmann, S. Webster, and S. Boyer, "Modeling modern network attacks and countermeasures using attack graphs," in *Computer Security Applications Conference, 2009. ACSAC '09. Annual*, Dec 2009, pp. 117–126.

[58] X. Ou, S. Govindavajhala, and A. W. Appel, "Mulval: A logic-based network security analyzer," in *Proceedings of the 14th Conference on USENIX Security Symposium - Volume 14*, ser. SSYM'05. Berkeley, CA, USA: USENIX Association, 2005, pp. 8–8.

[59] N. Poolsappasit, R. Dewri, and I. Ray, "Dynamic security risk management using bayesian attack graphs," *IEEE Transactions on Dependable and Secure Computing*, vol. 9, no. 1, pp. 61–74, Jan 2012.

[60] A. Roy, D. S. Kim, and K. S. Trivedi, "Attack countermeasure trees (ACT): towards unifying the constructs of attack and defense trees," *Security and Communication Networks*, vol. 5, no. 8, pp. 929–943, 2012.

[61] W. H. Sanders, "RRE: A game-theoretic intrusion response and recovery engine for process control applications," in *Critical Infrastructures, 2009. CRIS 2009. Fourth International Conference on*, March 2009, pp. 1–1.

[62] B. Kordy, P. Kordy, S. Mauw, and P. Schweitzer, *ADTool: Security Analysis with Attack–Defense Trees*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 173–176.

[63] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra, "Planning and acting in partially observable stochastic domains," *Artificial Intelligence*, vol. 101, no. 1, pp. 99 – 134, 1998. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S000437029800023X

[64] N. Kheir, N. Cuppens-Boulahia, F. Cuppens, and H. Debar, *A Service Dependency Model for Cost-Sensitive Intrusion Response*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 626–642.

[65] A. Shameli-Sendi and M. Dagenais, "ORCEF: Online response cost evaluation framework for intrusion response system," *Journal of Network and Computer Applications*, vol. 55, pp. 89 – 107, 2015.

[66] R. Bellman, "A markovian decision process," *Indiana University Mathematics Journal*, vol. 6, pp. 679–684, 1957.

[67] J. G. Kemeny, J. L. Snell *et al.*, *Finite markov chains*. van Nostrand Princeton, NJ, 1960, vol. 356.

[68] S. Jha, O. Sheyner, and J. Wing, "Two formal analyses of attack graphs," in *Computer Security Foundations Workshop, 2002. Proceedings. 15th IEEE*, 2002, pp. 49–63.

[69] J. Filar and K. Vrieze, *Competitive Markov Decision Processes*. New York, NY, USA: Springer-Verlag New York, Inc., 1996.

[70] M. H. A. Davis and P. Varaiya, "Dynamic programming conditions for partially observable stochastic systems," *SIAM Journal on Control*, vol. 11, no. 2, pp. 226–261, 1973.

[71] B. Kordy, S. Mauw, and P. Schweitzer, "Foundations of attack-defense trees," *Formal Aspects in Security and Trust*, vol. 6561, pp. 80–95.

[72] E. J. Pedhazur and L. P. Schmelkin, *Measurement, design, and analysis: An integrated approach*. Psychology Press, 2013.

[73] A. Jaquith, *Security metrics*. Pearson Education, 2007.

[74] S. Bandyopadhyay and S. Saha, *Some Single- and Multiobjective Optimization Techniques*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 17–58.

[75] K. De Jong, "Evolutionary computation: A unified approach," in *Proceedings of the 14th Annual Conference Companion on Genetic and Evolutionary Computation*, ser. GECCO '12. New York, NY, USA: ACM, 2012, pp. 737–750.

[76] R. Dawkins and J. R. Krebs, "Arms races between and within species," *Proceedings of the Royal Society of London B: Biological Sciences*, vol. 205, no. 1161, pp. 489–511, 1979.

[77] M. Dorigo, V. Maniezzo, and A. Colorni, "Ant system: optimization by a colony of cooperating agents," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 26, no. 1, pp. 29–41, Feb 1996.

[78] F. Glover and M. Laguna, "Principles of tabu search," 2007.

[79] D. P. Bertsekas, *Dynamic Programming and Optimal Control, Vol. II*, 3rd ed. Athena Scientific, 2007.

[80] R. Sutton and A. Barto, *Reinforcement learning: An introduction*. Cambridge Univ Press, 1998, vol. 116.

[81] "CVRF, common vulnerability report format," http://www.icasi.org/cvrf/, accessed: 2016-08-18.

[82] "NVD, national vulnerability database," https://nvd.nist.gov/, accessed: 2016-08-23.

[83] "OVAL, open vulnerability and assessment language," https://oval.mitre.org/, accessed: 2016-08-22.

[84] "CCE, common configuration enumeration," https://cce.mitre.org/, accessed: 2016-08-18.

[85] K. Scarfone and P. Mell, "The common configuration scoring system (CCSS): Metrics for software security configuration vulnerabilities," *NIST Interagency/Internal Report (NISTIR) - 7502*, 2010.

[86] "Common platform enumeration," https://cpe.mitre.org/, accessed: 2016-08-18.

[87] "ASR, asset summary report," https://scap.nist.gov/specifications/asr/, accessed: 2016-08-22.

[88] "CWE, common weakness enumeration," https://cwe.mitre.org/, accessed: 2016-08-18.

[89] "CWSS, common weakness scoring system," https://cwe.mitre.org/cwss/cwss_v1.0.1.html, accessed: 2016-08-18.

[90] E. LeMay, K. Scarfone, and P. Mell, "The common misuse scoring system (CMSS): Metrics for software feature misuse vulnerabilities," *NIST Interagency/Internal Report (NISTIR) - 7864*, 2012.

[91] "CWRAF, common weakness risk analysis framework," https://cwe.mitre.org/cwraf/, accessed: 2016-08-18.

[92] "CAPEC, common attack pattern enumeration and classification," https://capec.mitre.org/, accessed: 2016-08-18.

[93] "CRE, common remediation enumeration," https://scap.nist.gov/specifications/cre/, accessed: 2016-08-18.

[94] W. David, J. Christopher, K. Matthew, W. Matthew, and W. John, "Proposed open specifications for an enterprise remediation automation framework (draft)," *NIST Interagency/Internal Report (NISTIR) - 7670*, 2011.

[95] H. Debar, D. A. Curry, and B. S. Feinstein, "IDMEF, intrusion detection message exchange format," 2007.

[96] "TMSAD, trust model for security automation data," https://scap.nist.gov/specifications/tmsad/, accessed: 2016-08-22.

[97] "OpenIOC, open indicator of compromise," http://www.openioc.org/, accessed: 2016-08-22.

[98] "STIX, structured threat information eXpression," https://stixproject.github.io/, accessed: 2016-08-22.

[99] "TAXII, trusted automated exchange of indicator information," https://taxiiproject.github.io/, accessed: 2016-08-22.

[100] "CybOX, cyber observable expression," https://cyboxproject.github.io/, accessed: 2016-08-22.

[101] "XCCDF, eXtensible configuration checklist description format," https://scap.nist.gov/specifications/xccdf/, accessed: 2016-08-22.

[102] R. Danyliw, J. Meijer, and Y. Demchenko, "IODEF, the incident object description exchange format," 2007.

[103] "MAEC, malware attribute enumeration and characterization," https://maecproject.github.io/, accessed: 2016-08-22.

[104] D. Maynor, *Metasploit toolkit for penetration testing, exploit development, and vulnerability research*. Elsevier, 2011.

[105] "Common vulnerabilities scoring system," https://www.first.org/cvss, accessed: 2016-08-18.

[106] M. Keramati, A. Akbari, and M. Keramati, "CVSS-based security metrics for quantitative analysis of attack graphs," in *Computer and Knowledge Engineering (ICCKE), 2013 3th International eConference on*, Oct 2013, pp. 178–183.

[107] S. Wang, Z. Zhang, and Y. Kadobayashi, "Exploring attack graph for cost-benefit security hardening: A probabilistic approach," *Computers & Security*, vol. 32, pp. 158 – 169, 2013.

[108] "SCAP, security content automation protocol," https://scap.nist.gov/, accessed: 2016-08-22.

[109] "Mitre corporation," https://www.mitre.org/, accessed: 2016-08-22.

[110] M. R. Endsley, "Level of automation effects on performance, situation awareness and workload in a dynamic control task," *Ergonomics*, vol. 42, no. 3, pp. 462–492, 1999.

[111] H. Gomaa, *Software modeling and design: UML, use cases, patterns, and software architectures*. Cambridge University Press, 2011.

[112] A. Sforzin, F. Gómez Mármol, M. Conti, and J.-M. Bohli, "Raspberry Pi IDS: A fruitful Intrusion Detection System for IoT," in *13th IEEE International Conference on Advanced and Trusted Computing (ATC 2016)*, Toulouse, France, 2016, pp. –.

DRAFT

[113] A. B. Bondi, "Characteristics of scalability and their impact on performance," in *Proceedings of the 2Nd International Workshop on Software and Performance*, ser. WOSP '00. New York, NY, USA: ACM, 2000, pp. 195–203.

[114] M. Sipser, *Introduction to the Theory of Computation; 2nd ed.* Cambridge: Thomson Course Technology, 2006.

[115] R. Dewri, I. Ray, N. Poolsappasit, and D. Whitley, "Optimal security hardening on attack tree models of networks: a cost-benefit analysis," *International Journal of Information Security*, vol. 11, no. 3, pp. 167–188, 2012.

[116] "BugTraq, security focus mailing list," http://www.securityfocus.com/archive, accessed: 2017-07-04.

[117] "CERT/CC, cert coordination center," http://www.cert.org/, accessed: 2017-07-04.

[118] J. Kanclirz, *Netcat power tools*. Syngress, 2008.

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

[120] S. A. Butler, "Security attribute evaluation method: a cost-benefit approach," in *Software Engineering, 2002. ICSE 2002. Proceedings of the 24rd International Conference on*, May 2002, pp. 232–240.

[121] S. A. Butler and P. Fischbeck, "Multi-attribute risk assessment," in *Proceedings of the 3rd Symposium on Requirements Engineering for Information Security*, ser. CERIAS '02. West Lafayette, IN: CERIAS - Purdue University, 2002, pp. 2:1–2:12.

[122] A. Roy, D. S. Kim, and K. S. Trivedi, "Scalable optimal countermeasure selection using implicit enumeration on attack countermeasure trees," in *Proceedings of the 2012 42Nd Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, ser. DSN '12. Washington, DC, USA: IEEE Computer Society, 2012, pp. 1–12.

[123] L. J. Watters, "Reduction of integer polynomial programming problems to zero-one linear programming problems," *Operations Research*, vol. 15, no. 6, pp. 1171–1174, 1967.

[124] G. Stoneburner, A. Y. Goguen, and A. Feringa, "SP 800-30. risk management guide for information technology systems," Gaithersburg, MD, United States, Tech. Rep., 2002.

[125] "PTA, a practical threat analysis case study: next generation call accounting," http://www.ptatechnologies.com/Documents/TACSCase.pdf, accessed: 2016-09-09.

[126] R. H. Anderson, P. M. Feldman, S. Gerwehr, B. Houghton, and R. Mesic, "Securing the us defense information infrastructure: A proposed approach." DTIC Document, Tech. Rep., 1999.

[127] M. Gupta, J. Rees, A. Chaturvedi, and J. Chi, "Matching information security vulnerabilities to organizational security profiles: A genetic algorithm approach," *Decis. Support Syst.*, vol. 41, no. 3, pp. 592–603, Mar. 2006.

[128] R. Sarala, G. Zayaraz, and V. Vijayalakshmi, *Optimal Selection of Security Countermeasures for Effective Information Security*. New Delhi: Springer India, 2016, pp. 345–353.

[129] C. J. Chung, P. Khatkar, T. Xing, J. Lee, and D. Huang, "NICE: Network intrusion detection and countermeasure selection in virtual network systems," *IEEE Transactions on Dependable and Secure Computing*, vol. 10, no. 4, pp. 198–211, July 2013.

[130] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, "A view of cloud computing," *Commun. ACM*, vol. 53, no. 4, pp. 50–58, Apr. 2010.

[131] O. N. Fundation, "Software-defined networking: The new norm for networks," *ONF White Paper*, 2012.

[132] "Openflow," https://www.opennetworking.org/sdn-resources/openflow, accessed: 2016-08-25.

[133] G. Gu, J. Zhang, and W. Lee, "Botsniffer: Detecting botnet command and control channels in network traffic," 2008.

[134] M. Tupper and A. N. Zincir-Heywood, "VEA-bility security metric: A network security analysis tool," in *Availability, Reliability and Security, 2008. ARES 08. Third International Conference on*, March 2008, pp. 950–957.

[135] D. Perez-Botero, J. Szefer, and R. B. Lee, "Characterizing hypervisor vulnerabilities in cloud computing servers," in *Proceedings of the 2013 International Workshop on Security in Cloud Computing*, ser. Cloud Computing '13. New York, NY, USA: ACM, 2013, pp. 3–10.

[136] C. Kolias, G. Kambourakis, and M. Maragoudakis, "Swarm intelligence in intrusion detection: A survey," *Computers & Security*, vol. 30, no. 8, pp. 625–642, 2011. [Online]. Available: http://dx.doi.org/10.1016/j.cose.2011.08.009

[137] S. Zonouz and P. Haghani, "Cyber-physical security metric inference in smart grid critical infrastructures based on system administrators' responsive behavior," *Computers & Security*, vol. 39, Part B, pp. 190 – 200, 2013.

[138] R. Bellman, *Dynamic Programming*, 1st ed. Princeton, NJ, USA: Princeton University Press, 1957.

[139] C. E. Shannon, "A mathematical theory of communication," *SIGMOBILE Mob. Comput. Commun. Rev.*, vol. 5, no. 1, pp. 3–55, Jan. 2001.

[140] G. Owen, *Game theory*, 3rd ed. Emerald Group Publishing Limited, 1995.

[141] S.-J. Chen and S.-M. Chen, "Fuzzy risk analysis based on similarity measures of generalized fuzzy numbers," *IEEE Transactions on Fuzzy Systems*, vol. 11, no. 1, pp. 45–56, Feb 2003.

[142] F. G. Mármol and G. M. Pérez, "Security threats scenarios in trust and reputation models for distributed systems," *Computers & Security*, vol. 28, no. 7, pp. 545 – 556, 2009.

[143] G. Gonzalez-Granadillo, J. Garcia-Alfaro, E. Alvarez, M. El-Barbori, and H. Debar, "Selecting optimal countermeasures for attacks against critical systems using the attack volume model and the RORI index," *Comput. Electr. Eng.*, vol. 47, no. C, pp. 13–34, Oct. 2015.

[144] G. G. Granadillo, H. Débar, G. Jacob, C. Gaber, and M. Achemlal, "Individual countermeasure selection based on the return on response investment index," in *Proceedings of the 6th International Conference on Mathematical Methods, Models and Architectures for Computer Network Security: Computer Network Security*, ser. MMM-ACNS'12. Berlin, Heidelberg: Springer-Verlag, 2012, pp. 156–170.

[145] G. G. Granadillo, G. Jacob, H. Debar, and L. Coppolino, "Combination approach to select optimal countermeasures based on the rori index," in *Innovative Computing Technology (INTECH), 2012 Second International Conference on*, Sept 2012, pp. 38–45.

[146] R. S. Sandhu, E. J. Coyne, H. L. Feinstein, and C. E. Youman, "Role-based access control models," *Computer*, vol. 29, no. 2, pp. 38–47, Feb. 1996.

[147] A. A. E. Kalam, R. E. Baida, P. Balbiani, S. Benferhat, F. Cuppens, Y. Deswarte, A. Miege, C. Saurel, and G. Trouessin, "Organization based access control," in *Policies for Distributed Systems and Networks, 2003. Proceedings. POLICY 2003. IEEE 4th International Workshop on*, June 2003, pp. 120–131.

[148] "Federation of american scientists, special operations forces intelligence and electronic warfare operations, appendix d: Target analysis process, 1991," http://www.fas.org/irp/doddir/army/fm34-36/appd.htm.

[149] G. G. Granadillo, M. Belhaouane, H. Debar, and G. Jacob, "Rori-based countermeasure selection using the orbac formalism," *International journal of information security*, vol. 13, no. 1, pp. 63–79, 2014.

[150] L. P. Swiler, T. L. Paez, and R. L. Mayes, "Epistemic uncertainty quantification tutorial," in *Proceedings of the 27th International Modal Analysis Conference*, 2009.

[151] E. Miehling, M. Rasouli, and D. Teneketzis, "Optimal defense policies for partially observable spreading processes on bayesian attack graphs," in *Proceedings of the Second ACM Workshop on Moving Target Defense*, ser. MTD '15. New York, NY, USA: ACM, 2015, pp. 67–76.

[152] "Department of homeland security. moving target defense." https://www.dhs.gov/science-and-technology/csd-mtd, accessed: 2016-12-01.

[153] C.-L. Hwang and K. Yoon, *Methods for Multiple Attribute Decision Making*. Berlin, Heidelberg: Springer Berlin Heidelberg, 1981, pp. 58–191.

[154] R. Marler and J. Arora, "Survey of multi-objective optimization methods for engineering," *Structural and Multidisciplinary Optimization*, vol. 26, no. 6, pp. 369–395, 2004.

[155] I. Kotenko, O. Polubelova, I. Saenko, and E. Doynikova, "The ontology of metrics for security evaluation and decision support in siem systems," in *Availability, Reliability and Security (ARES), 2013 Eighth International Conference on*. IEEE, 2013, pp. 638–645.

[156] I. V. Kotenko and E. Doynikova, "Evaluation of computer network security based on attack graphs and security event processing." *JoWUA*, vol. 5, no. 3, pp. 14–29, 2014.

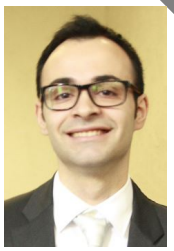
[157] E. Doynikova and I. Kotenko, "Countermeasure selection based on the attack and service dependency graphs for security incident management," in *International Conference on Risks and Security of Internet and Systems*. Springer, 2015, pp. 107–124.

[158] I. Kotenko and E. Doynikova, "Countermeasure selection in siem systems based on the integrated complex of security metrics," in *2015 23rd Euromicro International Conference on Parallel, Distributed, and Network-Based Processing*. IEEE, 2015, pp. 567–574.

[159] ——, "Dynamical calculation of security metrics for countermeasure selection in computer networks," in *2016 24th Euromicro International*

DRAFT

Conference on Parallel, Distributed, and Network-Based Processing (PDP). IEEE, 2016, pp. 558–565.
[160] I. Kotenko and A. Chechulin, "Attack modeling and security evaluation in siem systems," *International Transactions on Systems Science and Applications*, vol. 8, pp. 129–147, 2012.
[161] ——, "Common framework for attack modeling and security evaluation in siem systems," in *Green Computing and Communications (GreenCom), 2012 IEEE international conference on*. IEEE, 2012, pp. 94–101.
[162] E. Novikova and I. Kotenko, "Analytical visualization techniques for security information and event management," in *2013 21st Euromicro International Conference on Parallel, Distributed, and Network-Based Processing*. IEEE, 2013, pp. 519–525.
[163] I. Kotenko and A. Chechulin, "A cyber attack modeling and impact assessment framework," in *Cyber Conflict (CyCon), 2013 5th International Conference on*. IEEE, 2013, pp. 1–24.
[164] ——, "Computer attack modeling and security evaluation based on attack graphs," in *Intelligent Data Acquisition and Advanced Computing Systems (IDAACS), 2013 IEEE 7th International Conference on*, vol. 2. IEEE, 2013, pp. 614–619.
[165] ——, "Fast network attack modeling and security evaluation based on attack graphs," *Journal of Cyber Security and Mobility*, vol. 3, no. 1, pp. 27–46, 2014.
[166] I. Kotenko, A. Shorov, A. Chechulin, and E. Novikova, "Dynamical attack simulation for security information and event management," in *Information Fusion and Geographic Information Systems (IF AND GIS 2013)*. Springer, 2014, pp. 219–234.
[167] M. S. Ahmed, E. Al-Shaer, and L. Khan, "A novel quantitative approach for measuring network security," in *INFOCOM 2008. The 27th Conference on Computer Communications. IEEE*, April 2008.
[168] A. Shameli-Sendi, H. Louafi, W. He, and M. Cheriet, "Dynamic optimal countermeasure selection for intrusion response system," *IEEE Transactions on Dependable and Secure Computing*, 2016.
[169] R. T. Marler and J. S. Arora, "Survey of multi-objective optimization methods for engineering," *Structural and multidisciplinary optimization*, vol. 26, no. 6, pp. 369–395, 2004.
[170] E. A. Hansen and S. Zilberstein, "LAO*: A heuristic search algorithm that finds solutions with loops," *Artificial Intelligence*, vol. 129, no. 1, pp. 35 – 62, 2001.
[171] A. V. Goldberg and C. Harrelson, "Computing the shortest path: A search meets graph theory," in *Proceedings of the Sixteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, ser. SODA '05. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics, 2005, pp. 156–165.
[172] IBM, *Cognitive security (white paper)*, 2016.
[173] D. Waltermire, P. Cichonski, and K. Scarfone, "CPE, common platform enumeration: Applicability language specification," *NIST Interagency/Internal Report (NISTIR) - 7698*, 2011.
[174] A. Shiravi, H. Shiravi, M. Tavallaee, and A. A. Ghorbani, "Toward developing a systematic approach to generate benchmark datasets for intrusion detection," *Computers & Security*, vol. 31, no. 3, pp. 357 – 374, 2012.



Pantaleone Nespoli is a Ph.D candidate at the Department of Information and Communications Engineering at the University of Murcia, Spain. He holds a MSc in Computer Engineering from the University of Napoli Federico II. His research interests include Information & Communication Systems Security, and more specifically Network Security, Intrusion Detection and Response Systems, and Security Information and Event Management.

Dimitrios Papamartzivanos is a Ph.D. candidate at the Info-Sec-Lab, department of Information & Communication Systems Engineering, University of the Aegean, Greece. He holds an MSc in Information & Communication Systems Security in University of the Aegean, Greece. His research concentrate on Information & Communication Systems Security, and more specifically on the field of Intrusion Detection Systems assisted by Machine Learning technologies. His research deals both with misuse and anomaly detection systems. Additionally, his research interests include VoIP and IoT security and intrusion response systems. More info at: www.icsd.aegean.gr/dpapamartz

Félix Gómez Mármol is a researcher in the Department of Information and Communications Engineering at the University of Murcia, Spain. His research interests include cybersecurity, internet of things, machine learning and bio-inspired algorithms. He received an M.Sc. and Ph.D. in computer engineering from the University of Murcia, Spain. More info at: http://webs.um.es/felixgm.

Georgios Kambourakis received the Diploma in Applied Informatics from the Athens University of Economics and Business (AUEB) and the Ph.D. in Information and Communication Systems Engineering from the dept. of Information and Communications Systems Engineering of the University of the Aegean. Currently, Dr. Kambourakis is an Associate Professor at the dept. of Information and Communication Systems Engineering, University of the Aegean, Greece, and the director of Info-Sec-Lab. His research interests are in the fields of mobile and wireless networks security and privacy, VoIP security, Public Key Infrastructure, DNS security, and security education and he has more than 110 refereed publications in the above areas. More info at: www.icsd.aegean.gr/gkamb.

DRAFT