Adversarial Attacks against Intrusion Detection Systems: Taxonomy, Solutions and Open Issues

Igino Corona and Giorgio Giacinto and Fabio Roli
Dept. of Electrical and Electronic Engineering, University of Cagliari
Piazza d' Armi, 09123, Cagliari, Italy
{igino.corona}@diee.unica.it

Abstract—**Intrusion Detection Systems (IDSs) are one of the key components for securing computing infrastructures. Their objective is to protect against attempts to violate defense mechanisms. Indeed, IDSs themselves are part of the computing infrastructure, and thus they may be attacked by the same *adversaries* they are designed to detect. This is a relevant aspect, especially in safety-critical environments, such as hospitals, aircrafts, nuclear power plants, etc. To the best of our knowledge, this survey is the first work to present an overview on adversarial attacks against IDSs. In particular, this paper will provide the following original contributions: (a) a general taxonomy of attack tactics against IDSs; (b) an extensive description of how such attacks can be implemented by exploiting IDS weaknesses at different abstraction levels (c) for each attack implementation, a critical investigation of proposed solutions and open points. Finally, this paper will highlight the most promising research directions for the design of adversary-aware, harder-to-defeat IDS solutions. To this end, we leverage on our research experience in the field of intrusion detection, as well as on a thorough investigation of the relevant related works published so far.**

I. Introduction

Research in computer security must always take into account the presence of an *adversary*. In the last decade, attackers have become more organized, skilled and professional. Nowadays, the most serious Internet threats are posed by criminal organizations or nation-states seeking economic, military or political advantages. Criminal organizations manage wealthy underground markets, where a wide variety of illicit services and exploitation tools are commercialized [51], [155]. In particular, exploitation tools are routinely devised to defeat state-of-the-art security mechanisms [145], [122], [109], [26]. The main goal is to exploit security vulnerabilities, *without* actually being noticed by victims. In this way, criminals may stealthily retain control of target computers for further abuse [99].

In this scenario, Intrusion Detection Systems (IDSs) are nowadays recognized as necessary tools for the security of computer systems. IDSs aim at identifying violations of security policies, also known as intrusions, or intrusion attempts. In response to such events, IDSs can also perform automatic counteractions to protect computer systems and information[1] [41].

[1]Many commercial IDS products are also known as Intrusion *Prevention (or Protection)* Systems, e.g. [65], [30], [146], [63], to highlight their counteraction capability. In fact, such devices perfectly fit in our general definition of IDS.

As soon as they are deployed, IDS themselves become targets of attacks aimed at severely undermining their capabilities, and even turn them into *unconventional* attack tools. To the best of our knowledge, this survey is the first work proposing a broad overview of this relevant and complex problem, whose overall picture is still unclear. During the past years, there has been a significant number of research papers on related topics, each paper addressing specific issues, e.g., robustness of specific IDS classes [59], [137], detection approaches [100], [75] and/or attack tactics [103], [158]. In these works, many different terms are used to refer to attack tactics having the same goal or exploiting conceptually similar vulnerabilities, resulting in a confusing picture of the security scenario. In addition, relevant surveys on Intrusion Detection that categorize different IDS solutions, and propose the related taxonomies from the point of view of their detection mechanisms, usually neglect to analyze them from an adversarial point of view [39], [91], [25]. As a consequence, an overall picture of attacks against IDS is still missing. It is easy to see that such an outlook is crucial for the improvement of IDS solutions, especially for safety-critical computer infrastructures, such as those employed in hospitals, aircrafts, nuclear power plants [82]. In this paper we provide the following contributions:

- a general taxonomy of attack tactics against Intrusion Detection Systems;
- an extensive description of how such attacks can be implemented by exploiting IDS weaknesses at different abstraction levels, namely, measurement, classification and response;
- for each attack implementation, a critical analysis of some solutions that have been proposed so far, and the related open issues;
- the identification of some key research issues for the design of harder-to-defeat IDS solutions.

To this end, we resorted to our research experience in the field of intrusion detection, as well as to a thorough investigation of the related works published so far. In particular, we mostly focused on high-quality work published in top-tier venues in the computer security field[2].

Our contributions are structured in this paper in the follow-

[2]A reputable ranking of computer security conferences has been drawn by Guofei Gu, Texas A&M University: http://faculty.cs.tamu.edu/guofei/sec_conf_stat.htm.

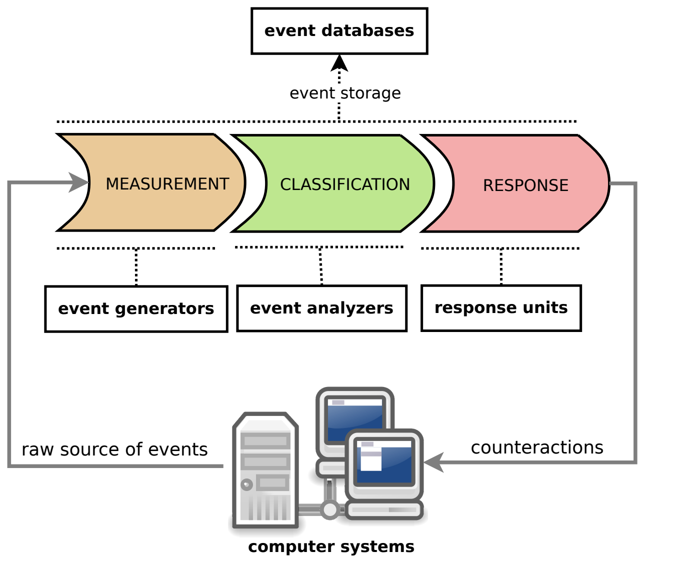


Fig. 1. General phases employed by an IDS, and the related components.

ing way. In Section I-A we present the general architecture of IDSs, and subdivide the intrusion detection task into three distinct phases, namely, measurement, classification and response. In Section I-B we provide a taxonomy of attacks against IDSs, that will be used as reference throughout the paper. Then, in Sections II, III, IV, we describe how such attacks can be implemented by leveraging on measurement, classification and response vulnerabilities, respectively. In these Sections we also outline the proposed approaches to address the reported attacks and outline the related open issues. Section V summarizes the most relevant aspects of IDS design and operation when addressing attack tactics against IDSs. Finally, in Section VI conclusions are drawn, and promising research directions are outlined.

Terminology: In the following, if not otherwise specified, the term *intrusion* will be used to refer to a *violation of security policies* associated to systems being monitored by an IDS. The term *attack* will be used to refer to *any* violation of security policies, including those associated to the IDS itself. Security policies may change according to the formulation of the specific security problem. However, without loss of generality, we might refer to violations of availability, integrity, and confidentiality of information managed by computer systems.

A. General Architecture of an Intrusion Detection System

Many IDS architectures have been proposed so far, differing each other in the approaches employed to gather and analyze data about computer systems [91]. Nevertheless, most of them rely on a relatively general architectural framework [149] based on four components, namely, *event generators*, *event analyzers*, *response units*, and *event databases* (see Figure 1). In this work, we refer to this general architectural framework, focusing on the three main functional phases of an IDS:

1) **Measurement** - *event generators* A vector of measurements, also known as *features*, about network and/or host events is used to characterize an event *pattern*. Such features are designed to accurately distinguish between intrusions and legitimate activities. Among the features used for intrusion detection are, for example, the number of unsuccessful logins, the number of half-open TCP connections, etc.
2) **Classification** - *event analyzers* Predefined models (e.g. rules), describing intrusions and/or legitimate patterns, are employed to classify (typically in real time) an event pattern as being either a legitimate or intrusive activity[3].
3) **Response** - *response units* If an intrusion pattern is detected, an alert is raised, and a suitable action can be taken to keep the computer systems in a safe state. For example, a firewall rule may be added to block an intrusion attempt.

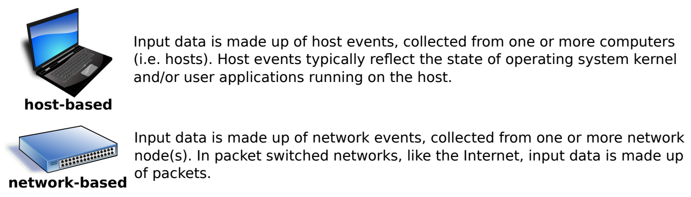
Such steps can also be viewed as the main "entry points" for the adversary to attack an IDS. We will refer to this general architectural framework to separately analyze each processing step, and to clearly point out and categorize different IDS vulnerabilities in Sections II, III, IV, respectively.

B. Taxonomy of Attacks against IDSs

As mentioned in Section I, a systematic, high-level categorization of attack tactics against IDSs is still lacking. We thus identify the following six main attack goals:

- **Evasion** an intrusion pattern is carefully modified so that the IDS won't be able to detect it (i.e., no alert will be raised). Evasion attacks are well known in the literature [159], [100], [49], [76], [87], [56], [75].
- **Overstimulation** a high number of patterns are crafted to generate false IDS alerts that overwhelm security operators and/or alert analyzers. While overstimulation attacks are less popular than evasion attacks, they have been studied (and implemented) in a number of works [112], [170], [101], [23].
- **Poisoning** well-crafted patterns are inserted into the set of data employed for designing the detection function of IDSs based on learning-by-example paradigms. The goal is to mislead the learning algorithm and negatively affect IDS performance in the classification phase (e.g., lower detection accuracy). Poisoning attacks represent a rather new and complex threat, which is gaining a lot of interest in the machine learning and security community [24], [27], [7], [102], [8], [81], [142].
- **Denial Of Service (DoS)** the detection activity of an IDS is inhibited, e.g., disabling IDS sensors [156], by generating well crafted patterns to overload an IDS sensor [59], or by slowing down a pattern-matching algorithm [36]. In the case of IDS working in *inline* mode (i.e., network packets are not routed until they are inspected, see e.g. [28]), this attack can also introduce significant packet drops/transfer delay (DoS against the monitored machines). In addition, this attack can also be used to support IDS evasion.
- **Response Hijacking** a pattern is crafted to generate an incorrect alert description and mislead the IDS response

[3]Indeed, intrusion detection may be a *multiple* classification task, that distinguishes between different intrusive events. However, we will refer to a two-class classification problem without loss of generality.



Input data is made up of host events, collected from one or more computers (i.e. hosts). Host events typically reflect the state of operating system kernel and/or user applications running on the host.

Input data is made up of network events, collected from one or more network node(s). In packet switched networks, like the Internet, input data is made up of packets.

Fig. 2. IDS solutions that collect either network or host data are usually referred to as network-based, and host-based IDS, respectively.

mechanism (either automatic or performed by a security operator), e.g., so that legitimate network connections are erroneously blocked [93].

- **Reverse Engineering** an adversary may gather information about the internal processing of an IDS (e.g., the employed features, or the detection algorithm) [100]. Such information can be used to conceive effective attacks aimed at attaining any of the above goals. For instance, an attacker can stimulate the IDS with well-known attack patterns to collect the related IDS responses for reconstructing the IDS detection technique [10].

These attacks may exploit different IDS vulnerabilities. Nevertheless, we observe that most of these vulnerabilities can be clearly associated with one of the processing phases described in Section I-A. Thus, we propose to categorize IDS vulnerabilities according to the IDS processing phase they are related to. The attack goal, together with (the category of) the exploited vulnerability can be used as a framework for the categorization of attacks against IDSs.

II. Measurement Phase

While IDS solutions may be extremely diverse in the set of the employed measurements, IDS sensors can be roughly subdivided according to the main source of data they rely on for detection purposes, i.e., the network level, and the host level (see Figure 2). For this reason, they are usually referred to as network-based IDS (NIDS), and host-based IDS (HIDS), respectively. Network sensors typically acquire data from one or more network nodes, e.g., network interfaces [126], [113], or network devices [169]. Host sensors typically acquire data from each monitored host, e.g., data about operating system (OS) kernel [89], file system [18], and applications [46]. Depending on the specific security problem, IDS categorization may be further split [91], but, for the sake of our investigation, we may refer to these two main abstraction levels without loss of generality.

The choice of the event sources is usually the result of a *tradeoff* between the complexity of the proposed IDS solution, and (a) the ease of implementation, (b) the deployment issues, and (c) the performance constraints. Depending on the target tradeoff, IDS solutions may either engage network or host measurements, or *both*. As we will discuss in the following, the latter option can lead to more reliable IDSs, but it is indeed more complex.

We identify four general attack tactics against the measurement phase. They leverage on weaknesses related to the chosen set of measurements, the input data, the event reconstruction phase, or the integrity/availability of IDS sensors. In the following Sections we present these attack tactics, and investigate how they can be addressed. Our analysis is independent from the data source, but, when necessary, we will differentiate between attacks targeting network measurements, and attacks targeting host measurements. In fact, network and host measurements allow to observe computer system events from different (complementary) perspectives which exhibit different pros and cons. That is, they may be attacked in different ways by an adversary.

A. *Set of Measurements*

The adversary may exploit limits in the discriminant capability of the chosen set of measurements to evade detection. In fact, for a given set of measurements, some intrusion instances may be unobservable, or weakly distinguishable from legitimate ones. Even if the employed measurements allow for accurately detecting *known* intrusion instances, they may be very poor for detecting novel or even *small* variations of known intrusion instances. This weakness is routinely exploited by modern malware (i.e., malicious, unwanted software) designed to defeat measurements performed by antivirus products, and evade them, by employing sophisticated techniques such as code metamorphism, cryptography, and virtualization [109].

Robustness against evasion can be achieved by choosing measurements that highlight *invariant* aspects of intrusive activities. In other words, they should take into account how intrusive activities can *evolve*, and capture characteristics that are invariant with respect to such an evolution. To this end, deep knowledge of the specific security problem is always necessary.

On the other hand, the *real* discriminant capability of the employed set of measurements can be evaluated by emulating the behavior of a skilled adversary, e.g., by studying new intrusion instances which attempt to mimic patterns produced by legitimate activities. We observe that such a knowledge can potentially be encoded by a security expert to *automatically* generate variations of known intrusion patterns, in a way similar to that proposed in [159] for evaluating detection models. This may be an interesting aspect to investigate in future works.

B. *Input Data*

An attacker may introduce errors in the raw (input) data collected by IDS sensors and thus lead to flawed measurements. For instance, a host sensor can leverage an OS *sys-call* to get the list of active processes (e.g., `get_active_processes` sys-call) running on the host. In this case, as showed in Fig. 3, an attacker may modify the data structures employed by the OS kernel (Direct Kernel Object Modification, DKOM), so that such a sys-call does not show one or more malicious processes [17]. To the best of our knowledge, this attack tactic has been investigated at the host level only. Nevertheless, in our opinion, the same attack tactic could also affect input data at the network level, e.g., if data is acquired from a network router, and the adversary is able to exploit an internal vulnerability of that network device.

It is easy to see that there is no general solution to this problem: depending on the input data, a tailored solution must be designed. For instance, kernel integrity checking [21] can be used to detect attack techniques that tamper the OS kernel data structures, such as DKOM. In addition, the integrity of OS and user applications can be enforced by preventing root abuse of file-system privileges [167], or preventing non-trusted kernel components from altering data employed by the kernel to manage its own execution, through a hypervisor-based solution [147].

C. Event Reconstruction

Depending on the measurement needed to perform attack detection, IDS sensors may require to intercept data from the monitored system(s), e.g., data processed by a user application. However, in some cases, it may be easier to collect data at a lower abstraction level, and *emulate* the internal processing of the monitored systems to extract the target measurements. Unfortunately, this approach allow attackers to cause a flawed event representation by exploiting any emulation error. For instance, as in modern malware detectors, host sensors may implement both file-type inference ,and format-specific parsing heuristics to extract information from a file (e.g., a PDF document), without the need of intercepting file data extracted by the associated application (e.g., Adobe Reader). In this way they can easily emulate the behavior of different applications/OS when recognizing file types, and interpreting their content. In this case, an adversary can easily create new malware instances that defeat such measurement heuristics [72].

In general, the higher the *semantic gap* between the collected data, and the measurements of interest, the higher the emulation complexity, and the likelihood of event reconstruction errors. It is easy to see that this aspect is crucial, especially for *network sensors employed to detect intrusions against user applications/OS* (semantic gap between network packets and host-side interpretation of such packets). In fact, there are some well-known attack tactics against network measurements [59], that are reported in the following.

Tunneling: Intrusive traffic is enclosed in a "tunnel", i.e., a type of traffic that is not inspected or that it is not observable by the network sensor. For instance, malicious traffic can be encrypted [79], e.g. using Transport Layer Security (TLS) [68], or routed on non-standard TCP ports, e.g., the HTTP traffic (port 80) could be routed on port 22. In the first case, the network sensor may ignore the traffic if it lacks the proper decryption routine, and the cryptographic key. In the second case, the sensor may ignore (or fail to process) the traffic flowing on TCP port 22, since this port is normally used by the Secure SHell (SSH) protocol [67].

Desynchronization: The traffic view of network sensors is forced to be "out of phase" with respect to the monitored host(s) for an entire session [59]. This effect can be obtained by either sending packets that will be accepted by the sensor but that won't be processed by the monitored host(s) or vice-versa [123]. For instance, if a router is between the sensor and the monitored host(s), such an attack can be performed through the manipulation of the Time To Live (TTL) field of IP packets [136].

Encoding Variations: Intrusive traffic is encoded so that its semantic on the sensor is different from the semantic on the target. A typical example is the manipulation of the request URI of a HTTP message [66], as shown in Figure 4. The request URI `/MSADC/root.exe` has the same "meaning" of `/MSADC/../MSADC/root.exe`, because `/MSADC/../` is equivalent to the root directory `/`. But, since the sensor may not apply the same URI conversion employed by the victim web server, attack detection based on string matching between the known attack signature `/MSADC/root.exe?/c+dir` and the new attack instance fails. Thus, the attack is successful and no alerts are raised.

While this attack has been presented in the context of network-based IDSs, we observe that it can be performed in different contexts. In fact, the same attack can also be performed against host sensors, e.g., if the input data (the request URI) is gathered from web server logs, as in [88].

Segmentation: Intrusive traffic is subdivided into multiple parts, sent in an order such that the reconstruction made by the network sensor differs from that made by the destination hosts. The success of this technique is based on the fact that different Operating Systems (OSs) may process duplicate or overlapping fragments (at different levels of the TCP/IP protocol stack) with different reassembly policies [136], [107], whereas network sensors often employ a *unique* reassembly policy that is assumed to be coherent with all OSs of the monitored host(s).

*1) **Strengthening Event Reconstruction**:* We recognize a number of ways for strengthening event reconstruction, and they are described in the following.

Tight Integration with Monitored Systems: Integrating host sensors with the monitored applications or OS kernel is a general approach that can deal with event reconstruction vulnerabilities. For example, as suggested in [72] in the context of malware detection, tight integration can naturally address semantic gaps between data processed by the host sensor, and data processed by the applications. When tight integration is not feasible, sensor data processing should be thoroughly tailored to the monitored system, to limit possible flaws related to data reconstruction.

Sensors Placement: Ideally, network sensors should be able to capture all network traffic on the monitored machines to cope with desynchronization attacks. To this end, a client-based *distributed* approach, i.e., a network sensor for each monitored host, can be a solution [9]. Such an approach may also cope with flooding attacks, since the network traffic is distributed among sensors, and there is no single point of failure. In practice, it is sufficient to place a network sensor for each *collision domain*, i.e., a network segment where the host(s) share a physical link, so that the sensor can observe the traffic related to any host in this segment. A systematic approach to network sensors placements has been proposed in [127]. The shortcoming of this solution is that the correct sensor placement requires the knowledge of detailed attack definitions, that is not always available (indeed, it is not available for *never-before-seen* attacks).

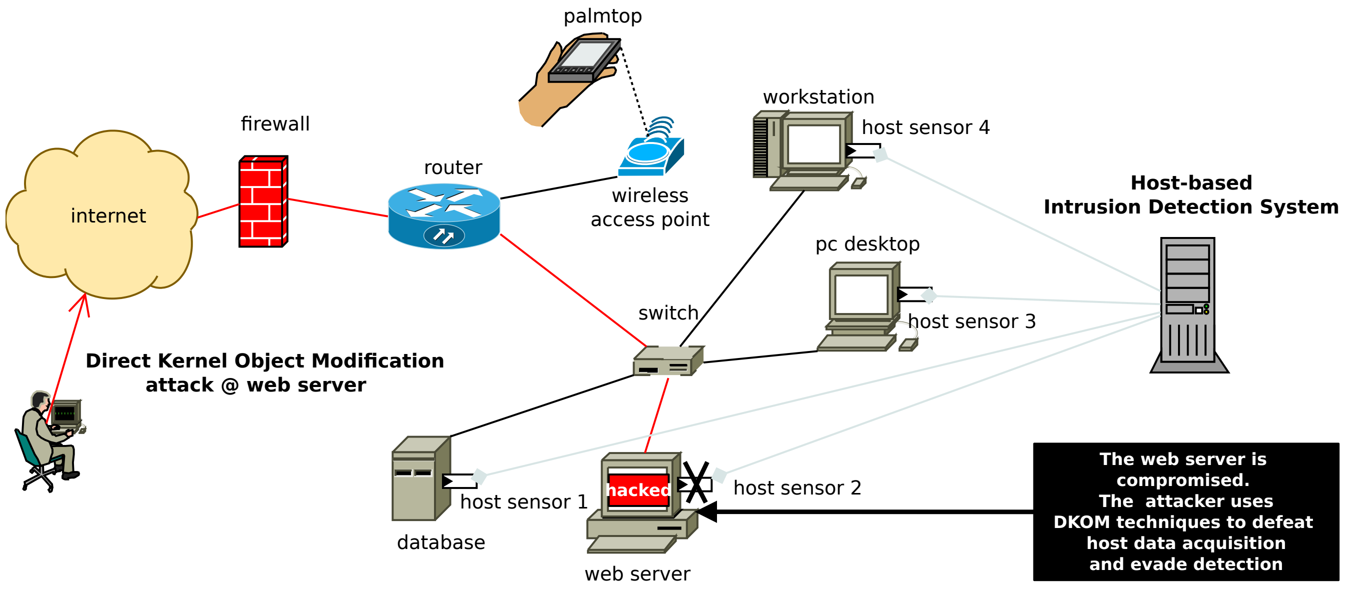

palmtop
workstation
host sensor 4
firewall
router
wireless access point
internet
Host-based Intrusion Detection System
pc desktop
switch
host sensor 3
Direct Kernel Object Modification attack @ web server
hacked
host sensor 1
host sensor 2
database
web server
The web server is compromised. The attacker uses DKOM techniques to defeat host data acquisition and evade detection

Fig. 3. Once the web server is compromised, a skilled attacker employs Direct Kernel Object Modification techniques to evade the host sensor.

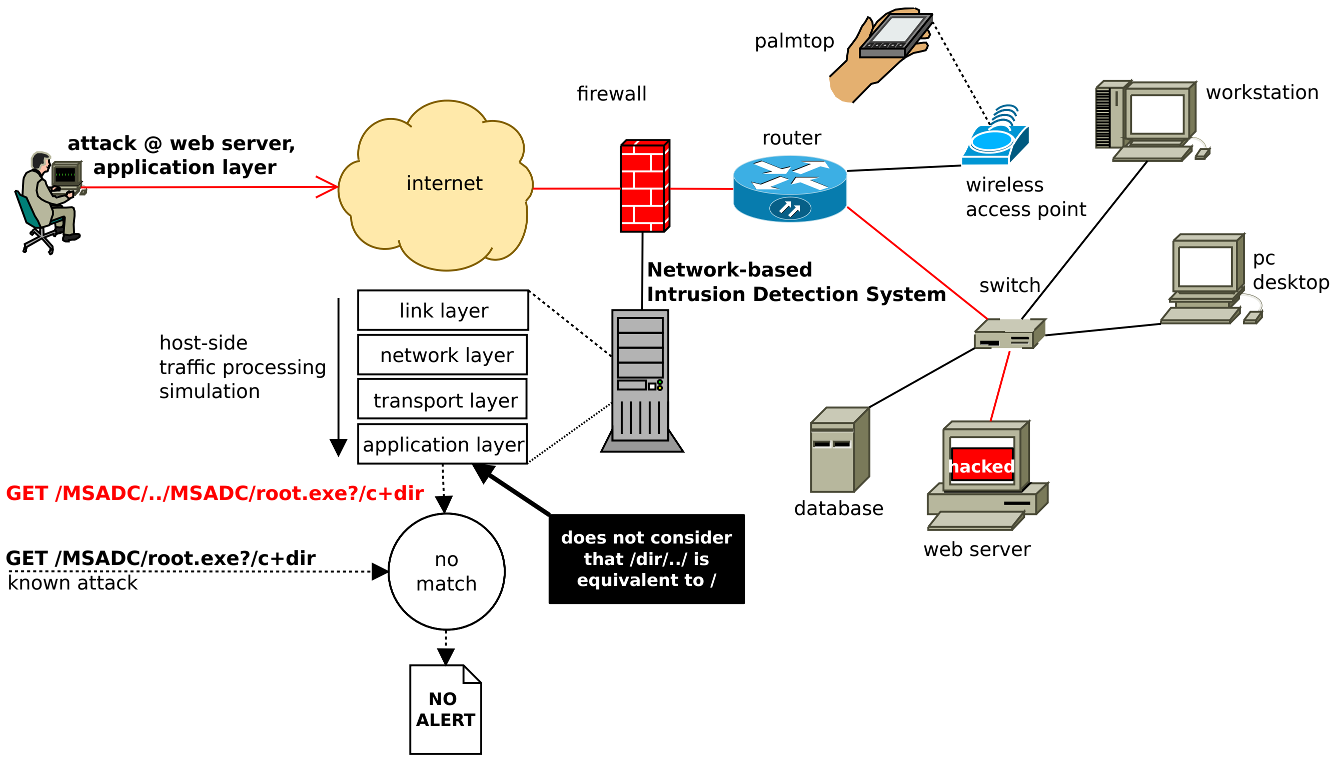

palmtop
firewall
workstation
router
attack @ web server, application layer
internet
wireless access point
pc desktop
Network-based Intrusion Detection System
switch
link layer
host-side traffic processing simulation
network layer
transport layer
application layer
GET /MSADC/../MSADC/root.exe?/c+dir
database
hacked
web server
does not consider that /dir/../ is equivalent to /
GET /MSADC/root.exe?/c+dir
known attack
no match
NO ALERT

Fig. 4. A skilled attacker evades NIDS detection exploiting a flawed reconstruction of the traffic at the application level (i.e. through encoding variations.) The attack has been devised both to compromise the web server, and to evade detection.

Compartmentalization: Compartmentalization is a well-known concept used for safety and security in different scenarios [133]. This concept can be used to strengthen measurements at the network level by clustering machines according to their operating system, or according to the applications that exchange data through the network. Each cluster can be seen as a different compartment, and network sensors may be thoroughly tuned for the compartment they are meant to monitor. In this way, it is easier to keep up with software updates, and reduce the traffic interpretation gap between NIDS and operating systems/applications running at the host-side.

Bifurcating Analysis: For each possible interpretation of the traffic, a distinct analysis thread may be executed [114], [19]. In this case, it is necessary a tradeoff between the maximum number of threads (that affects the amount of memory needed, and the computational cost), and the completeness of the method.

Traffic Normalization: Network traffic can be processed to eliminate interpretation ambiguities, by reducing the amount of connection state information that network sensors must manage [114], [58], [166], [38]. Traffic normalization can effectively deal with segmentation and desynchronization attacks at all levels of the TCP/IP stack, and nowadays almost all network-based IDS products implement some kind of traffic normalization (see e.g. [63], [65], [30], [146], [150]). On the other hand, traffic normalization must face some tradeoffs related to end-to-end semantics preservation, performance, and the number of states held [58]. This is a challenging task that explains why security appliances are still vulnerable to event reconstruction errors [150].

Data Enhancement: Network data can be transformed in such a way that relevant information, i.e., information that can be reliably exploited to discriminate intrusions from legitimate activities, is enhanced. For instance, Central Processing Unit (CPU) emulation [121], [120] or sandboxing techniques [4] can be used to evaluate network traffic portions that may contain (malicious) executable code. Other data enhancement techniques include network traffic clustering [118], and low level classification [104], [163].

Active Mapping: A *mapper* can actively probe the network to build a profile of the network topology, and collect the traffic processing policies of hosts in the probed network. Such a profile is then used to disambiguate the interpretation of the network traffic on a per-host basis [136]. Thus, active mapping does not require any modification of the network traffic, and avoids the semantic and performance problems of traffic normalization. Active mapping is also a synonym of *target-based* stream reassembly [107], [119].

Dynamic Protocol Analysis: The content of network traffic can be analyzed to identify the employed protocols, without relying on TCP/UDP ports [43]. Afterwards, a suitable protocol-specific IDS module can be instantiated. In this way, it is possible to cope with tunneling techniques based on non-standard ports, and to detect stealthy —unauthorized— host applications [43].

Data Correlation: Data related to different abstraction levels can be integrated, e.g., host measurements can be integrated with network measurements. In this way, even if host measurements are evaded or tampered by an adversary, we may still detect suspicious patterns in network traffic, and vice-versa. For instance, malware behavior can be characterized through network traffic analysis [83], [122], [118]. On the other hand, network-level data may be integrated with data related to applications, and information on the operating systems running on the monitored hosts [44]. This approach can easily address a number of attack tactics at all levels of the TCP/IP stack such as desynchronization, segmentation and tunneling.

Moreover, an interesting approach is represented by data reconciliation [35]. Data reconciliation is a well-known technique in the field of process control. Basically, this technique exploits redundancy in the set of measurements to get a more reliable evaluation of a certain variable. For example, we may measure the same variable (e.g., request URI) from both a network node and a host sensor (e.g., a sensor integrated within the web server), and then check whether the two values match or not. If the two values do not match, then the most likely cause can be either (1) a successful attack against the target host affected the host measurement, or (2) a flaw in the network traffic reconstruction module of the network sensor (this may be due to an evasion attack against the network sensor). We may also correlate information among different host sensors, e.g., sensors placed in different hosts that communicate each other, or different network sensors, e.g., sensors sharing a common traffic flow.

Evaluation through Attack Automation: The robustness of the event reconstruction phase can be evaluated by testing the IDS behavior using some tools that can automate and combine the attacks described above. For instance, the robustness of IDS in network event reconstruction can be evaluated by resorting to tools/libraries such as `fragroute` [143], `LibWhisker` [50], `idsprobe` [73], `pytbull` [37] and `evader` [151].

To the best of our knowledge, there are no tools specifically tailored to exploit errors in host event reconstruction. This lack is quite reasonable, because, as mentioned in Section II, host level data comes from heterogeneous sources, and, as a consequence, a general event reconstruction technique cannot be defined. However, we recognize that, if we consider particular cases, automating attacks against host event reconstruction is indeed feasible (see e.g. [72]).

D. *Integrity and Availability Attacks*

To inhibit the core tasks performed by IDS sensors, different techniques can be employed according to the class the sensor belongs to. For example, a network sensor is typically exposed to *availability* attacks: it can be overloaded by generating an amount of network traffic which exceeds its bandwidth (flooding), so that it cannot process subsequent packets [59]. It turns out that any dropped packet may be part of a missed attack (evasion). In order to significantly extend network bandwidth, and thus reduce the impact of flooding attacks, network-based IDSs are typically implemented as a standalone devices, employing dedicated hardware and software (see e.g. [63], [65], [30]). Robustness against flooding and event reconstruction attacks may be the result of a tradeoff with some performance constraints. For example, it may be possible to spot fragmentation attacks without performing any packet reassembly, at the cost of additional false alarms [5]. Moreover, network data may be reduced to a tractable amount, e.g., through statistical subsampling [171], [106] or by heuristic-based filtering techniques [115], [120]. In particular, heuristics can be used to proactively drop packets that are less likely to affect the accuracy of the IDS (selective packet dropping) [110].

More powerful adversarial attacks can be devised against host sensors, since they are often implemented as software programs that run on the monitored host(s) [137]. That is, while host sensors potentially allow for tight integration with host applications/OS kernel, they are typically more exposed to attacks that violate both their *integrity* and *availability*. Once a host is compromised, e.g., because the attacker has acquired administration rights, sensors within such a computer may be compromised as well, or disabled [156].

The principal solution proposed so far to protect the integrity of host sensors is based on virtualization, i.e., the monitored machine runs in a virtual environment (guest virtual machine), and a higher privileged *hypervisor* enforces memory protections, and intercepts the events within the virtual machine [137].

One approach that leverage on virtualization is known as Virtual Machine Introspection (VMI) [54]. Host sensors run in a virtual environment that is separate (trusted) from the monitored hosts, and, by means of a Virtual Machine Monitor (VMM), they acquire data related to the state of the monitored machine. The VMM is able to collect low level information such as the bytes written on physical memory or on disk. The sensors interact with an Operating System (OS) library, whose task is to translate low level information coming from the VMM into high (OS) level information (e.g., list of running processes, file contents, etc.) about the monitored machine. VMI is an effective approach, but it must face some important issues:

- **virtualization vulnerabilities**; an attacker may exploit flaws in the design or implementation of the VMM in order to prevent the correct acquisition of data from the monitored machine [165].
- **OS library vulnerabilities**; since such a library relies on prior knowledge of data structures employed by the OS kernel, the library could fail to correctly infer its state [6] if the attacker is able to modify such data structures in the target machine (e.g., by using DKOM techniques). However, recent work proposed new introspection techniques that do not require the employment of OS-specific libraries. Such proposals are mainly based on the communication between the guest machine and another (secure) virtual machine. The secure virtual machine can perform introspection either (a) by learning the behavior of small inspection programs (i.e., host sensors) within the guest machine given their dynamic execution traces [42], or (b) by invoking guest functions (function-call injection) and checking their secure execution (localized shepherding) [20], or (c) by using the same kernel of the guest machine and executing kernel data retrieved from the guest machine [52]. While such solutions are able to reduce the semantic gap between collected data and measurement of interest, their introspection capabilities are still limited and future work is needed to thoroughly evaluate their reliability from a security standpoint.
- **performance**; the overhead added by switches between virtual machines for each invocation of the VMM may be relevant. Hence VMI is suitable for applications that tolerate high-overhead [124], [53].

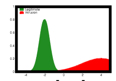
Hence, the relevance of these issues should be always compared to the benefits of the VMI technique for host data acquisition. Another approach is to leave host sensors running on the monitored machine and protect them through hardware virtualization techniques (in-VM monitoring) [137]. In such a case, host sensors run and access data on a hypervisor protected address space. This approach is very interesting, because it guarantees at least the same protection of VMI, while exhibiting better performances.

III. Classification Phase

The goal of this phase is to accurately classify each event pattern as being either intrusive or legitimate. In addition, when a pattern is classified as being intrusive, the system should provide a *reliable*, and *human-readable* interpretation of each intrusion pattern, e.g., targeted vulnerability, posed threats, plausible attacker's goal. As depicted in Figure 5, classification is usually performed by matching in real-time a test pattern against one or more models describing intrusive patterns (misuse detection) and/or legitimate patterns (anomaly detection).

A purely manual process in the definition of classification models may be too expensive and too slow to cope with current threats, characterized by complex and rapidly evolving environmental settings [22], [158]. Thus, in the past years there has been an increase in the adoption of machine learning algorithms to support an effective adaptability of detection models. Given a *statistically representative* set of patterns (training examples) associated to intrusion and/or legitimate activities, machine learning algorithms can automatically build (learn) accurate detection models [92], [104], [25], [84], [144]. Statistical representativity translates into three main practical problems:

- **Privacy** Event patterns can potentially contain sensitive information about computer system users. This aspect may lead to *privacy* concerns when collecting data about (legitimate) user activities. A discussion on this problem goes outside the scope of this paper, and we refer the interested reader to [48].
- **Real-world intrusions** An up-to-date set of real-world intrusions should be collected. This task strictly depends on the specific security problem, and it is challenging because of the rapid evolution of intrusions.
- **Ground Truth** The class of each training pattern should be thoroughly validated, to address the unavoidable presence of *noise*. This task usually requires deep human



A model of the normal (legitimate) activity is defined. An alert is raised if current events identify anomalous activity, i.e. activity that deviates from the model of normal activities.

anomaly detection

An alert is raised if current events identify known intrusive activities (attacks). Attacks are usually described through a set of "fingerprints", called signatures.

misuse detection

Fig. 5. Detection models based on either intrusive or legitimate patterns are usually named as misuse-based or anomaly-based, respectively.

expertise (i.e., prior knowledge about a specific security problem), and it may be challenging, especially with large amounts of training data. This challenge is the key reason why poisoning attacks are indeed feasible (see Section III-D).

The two main detection approaches, namely misuse-based, and anomaly-based, exhibit different pros and cons. Misuse-based detection approaches provide a human-readable description of intrusive events, because patterns are labelled as being intrusive if they match one of the attack models stored in the system, and attack models are usually written by security experts [126], [113]. But misuse-based IDSs are not effective against novel attacks, e.g., attacks for which there are no available signatures. On the other hand, anomaly-based IDSs label a pattern as being potentially intrusive if it does not match any of the models of legitimate activity. Thus they can just provide a description of the anomaly, which does not necessarily reflect a description of intrusions, also because the detected attack can be novel.

In general, classification models can be designed as a *mixture* of misuse- and anomaly-based approaches in order to exploit their respective pros [138]. For instance, when representative examples of both legitimate and malicious patterns are available, machine learning techniques can be used to build a *discriminative* detection model. Such a model can embed information related to *both* intrusive and legitimate patterns, to discriminate between them. In some sense, such a detection model describes neither intrusion, nor legitimate patterns, but differences between them. This allows to enhance misuse detection accuracy, and even detect never-before-seen attacks.

In the following, we provide a general overview of attacks against the classification phase, and, whenever necessary, we highlight how such attacks can be implemented. While we will refer to either misuse or anomaly detection approaches, our analysis can be extended to any other detection model, without loss of generality.

A. Difference between Alert Space and Intrusion Space

Let us define as *intrusion space* (I) the set of all intrusive patterns, and as *alert space* (A) the set of patterns which raise an alert according to the employed classification algorithm (see Figure 6). It is easy to see that any pattern within the set $I \cup A - I \cap A$ can be used to attack the classification algorithm. In particular, patterns within subset $I - A$ (missed alarms) or $A - I$ (false alarms) can be used for evasion or overstimulation attacks, respectively. As described in Table I,

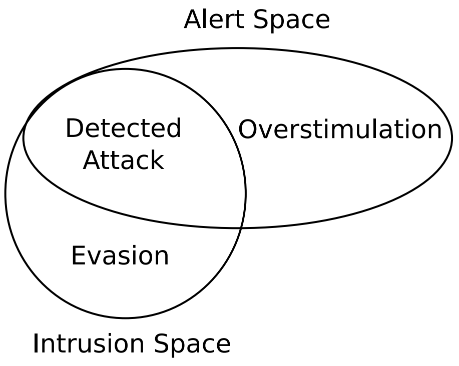


Fig. 6. Alert space and Intrusion space: the goal is to find a classification algorithm such that the alert space tends to overlap with the intrusion space.

these patterns can be generated in different ways, depending on the employed detection approach.

In order to address both overstimulation and evasion attacks, we identify a number of techniques, described in the following.

Exploiting Contextual Information: The robustness against overstimulation attacks can be augmented by exploiting contextual information about hosts and services being monitored [141]. This approach is the basis of the so-called *alert verification* mechanisms [85]. For example, it can be inspected whether the response of a victim host to an attack is anomalous [14]; if vulnerable applications, operating systems or services are running; whether certain packets reached a certain destination or not; whether new services (e.g., new applications, open ports, etc.) have been recently activated; whether a denial of service is present, or whether critical files properties (e.g., permissions, owner, size, content) have been recently changed [85]. In addition, contextual information can be obtained by *correlating* other alerts produced either by the IDS itself, or by other network appliances (e.g., other IDSs, firewalls, routers, etc.) [40], [86], [3]. The use of high-level concepts to exploit contextual information allows to significantly improve alert descriptions [141] and reduce the false alarm rate [85]. Hence, contextual information could potentially be used to address, and even *detect* overstimulation attacks.

Proactive Approaches against Mimicry Attacks: As described in Table I, in order to evade anomaly detection the adversary may craft intrusions to mimic legitimate patterns (mimicry attacks). A proactive approach to counteract such attacks is to inject fake legitimate traffic into the traffic generated by legitimate users [16]. In this way, mimicry attacks can be detected by the IDS as traffic patterns that emulate the profile of such fake legitimate traffic. Hence, this approach works under the hypothesis that the attacker is not able to distinguish fake from actual user activity.

Cost-Sensitive Classification: The evaluation of the cost of the damage caused by an attack forms the basis of cost-sensitive intrusion detection [93]. Damage costs should reflect (a) the value of the attack's target and (b) how the target is affected if the attack is successful. This may help focusing on the most relevant security-related events when alerts are

TABLE I
SUMMARY OF THE KEY ATTACKS AGAINST MISUSE- AND ANOMALY-BASED IDS.

Attack Goal	Detection Model	
	Misuse-based	**Anomaly-based**
Evasion	the adversary may modify an attack so that its pattern does not match any signature [159], [100], [87], [56]	the adversary may modify an attack to mimic normal traffic patterns (*mimicry* attack) [49], [76], [75]
Overstimulation	the adversary may generate event patterns matching one or more signatures, but that do not represent any threat for the monitored systems [112], [170], [101], [23]	the adversary may submit anomalous patterns which do not reflect any threat to monitored systems *(This attack is indeed possible, but to the best of our knowledge, it has not yet been reported)*.

analyzed, and attack response are generated. We note that overstimulation attacks, as well as generic false alarms, can be effectively addressed if the IDS is able to categorize them as alarms with *low* damage costs. In fact, some IDS products implement cost-sensitive classification (risk rating), and assign a confidence to each alert (signature fidelity) as part of the IDS policy management, e.g. [31]. To this end, a thorough study of relevant assets is essential [93].

Classifier Ensembles: The intrusion detection problem can be subdivided into multiple, complementary subproblems, e.g., tailored for a *specific* class of intrusions. Prior information (i.e., domain expertise) about each subproblem can be exploited to choose a limited and suitable set of measurements. This approach can decrease the complexity of the intrusion detection task, allowing for a more accurate modeling of event patterns [94]. It can also support an easier interpretation of the *meaning* of event patterns (e.g., improving alert descriptions) and the implementation of alert verification mechanisms. Finally, this approach is flexible, because new specialized classifiers can be easily added to detect a wider range of attacks.

Each subproblem can also be approached using the *classifier ensemble* paradigm, i.e., different classification techniques are employed in parallel to perform the same classification task. The outputs of these classifiers are then combined to support a final decision about the most likely class the event pattern belongs to. It has been shown theoretically [12] and experimentally [117] that a classifier ensemble may be harder to evade than a single classifier. For example, each classifier may employ:

- a different *bootstrap* replica of the original set of examples, obtained through random sampling with replacement. This technique is also known as *bagging* [134].
- a different feature set, i.e. different pattern representations, for the same event [117]. This approach is also known as *feature bagging* [90] or *random subspace method* [160].
- a different model, i.e., each classifier employs a different machine learning algorithm to build its model(s).

Then, an event may be classified as intrusive only if the *majority* of pattern classifiers agrees. Intuitively, evasion attacks can be more difficult to devise since the adversary should modify an attack pattern to evade the majority of classifiers in the ensemble, rather than a single classifier. Other decision fusion techniques can be employed to produce the final output according to the problem formulation, and to constraints on the final performances of the system.

Automatic Evaluation: There are some techniques that have been proposed to automatically evaluate the robustness of classification algorithms against evasion and overstimulation attacks. The robustness of a classification algorithm against evasion can be evaluated by using attack mutation techniques, which automatically generate variations of known attacks [101], [159], [130], [129], [49], [77], [98], [87]. Similarly, robustness against overstimulation can be evaluated through automatic generation of overstimulation patterns based on the knowledge of the target detection model. Surprisingly, we found that overstimulation attacks have been investigated [170], [101] and implemented (see e.g. `stick` [33], and `snot` [140]) in the context of misuse detection only. On the other hand, automatic overstimulation against anomaly detection is indeed feasible[4], and it would be a very interesting area of research.

B. Pattern Matching

In some cases, the adversary may be able to exploit vulnerabilities related to the implementation of pattern matching algorithms. In particular, rule matching algorithms may be dramatically slowed down by means of algorithmic complexity attacks [36], [139]. The adversary may overload the IDS (thus causing a Denial of Service attack) through the generation of well-crafted traffic that cause the rule matching algorithm to exhibit its worst-case performances. This way, further attacks against the monitored systems may go undetected. Algorithmic complexity attacks can be considered as a particular case of flooding attacks (see Section II) that target the IDS at the classification level.

A possible defense against algorithmic complexity attacks relies on the use of rule matching algorithms whose worst-case performances are bounded, or whose worst-case behavior is not predictable [36]. On the other hand, rule-matching algorithms may be strengthened by keeping track of intermediate matching results [139], and significantly speeded up by compressing the size of the discrete state automata corresponding to the overall set of rules [111].

[4]We believe that overstimulation may be even easier to produce, since an anomalous pattern does not necessarily represent an attack.

C. *Description of Intrusive Events*

Even if the adversary is not able to evade the IDS, he may modify an attack instance so that an incorrect or too generic alert description is triggered[5] [23]. In this way, the security analyst may be unable to understand either the targeted vulnerability, or the real threat posed by the attack, or the actual attacker's goal. In addition, an incorrect alert description may cause an erroneous response by the IDS or by the security analyst (see Section IV). The adversary may deliberately craft an attack in order to generate a misleading alert, and thus take some control over the IDS/security analyst's response. For instance, let us assume that, through the insertion of a firewall rule, the IDS "safely" drops all packets from the IP addresses whose previous traffic caused the IDS to raise one or more alert(s). The underlying assumption is that the IP detected by the IDS is the actual IP address of the attacker's machine. Unfortunately, the adversary may use a spoofed source IP address X [45] pertaining to a legitimate host in the protected network. This way, the attacker may exploit the IDS response (Response Hijacking attack, see Section I-B) to block all packets from the legitimate host (i.e., the victim), thus having the IDS to cause a Denial of Service of the victim host[6].

Classification Confidence: Alert descriptions may be augmented with a *confidence* value. In this way, the IDS response can take into account the (estimated) reliability of such descriptions, to reduce the likelihood that legitimate activities are affected by the IDS response [168], [108].

Automated Attack Inference: Anomaly detection is a fundamental approach, because it allows detecting novel attacks. Unfortunately, anomalies cannot be easily translated into an adequate description of intrusions (e.g., attacker's goal, exploited vulnerability, etc.). To overcome this problem, techniques that automatically associate anomalies to *known* attack definitions should be devised. As a consequence, it would be easier to spot variants of known (or even never-before-seen) intrusions, given that they produce similar anomalies. The association of anomalous events to known attacks can be carried out according to heuristics thoroughly defined by human experts [125], or can be automatically built through machine learning algorithms [15], given the association between a sample of attacks (with known attack descriptions), and their related anomalies.

Model of the Adversary: One of the most exciting (and challenging) issues in adversarial pattern classification is to model the profile of attackers, to determine their goals, and better describe intrusive events. For instance, an attacker model has been proposed to understand the impact of adversarial actions on measuring unwanted Internet traffic [2], and to compare the security properties of different routing protocols in mobile ad-hoc networks [32]. An attacker model is also (implicitly) assumed by a recent approach called Active Intrusion Prevention (AIP) [55]. The IDS keeps track of all the requested data that could be used to jeopardize the protected systems (suspicious requests). Then, "true" attacks are identified as any request that exploits such data. Thus, AIP actually models the behavior of an adversary as a two-step process: (1) information gathering, and (2) attack realization.

[5]In fact, this is usually a side effect of evasion attacks, see e.g. [100], [73].

[6]The identification of spoofing attacks may not be an easy task [154], [1].

D. *Poisoning attacks*

As mentioned in Section III, machine learning (ML) algorithms are increasingly adopted to support an effective adaptability of detection models. A number of machine learning models, including Support Vector Machines (SVM) [117], Hidden Markov Models (HMM) [34], N-grams [162], Decision Trees [96], Artificial Neural Networks [47] have been successfully applied for solving different intrusion detection tasks.

However, the automatic adaptability of machine learning techniques is threatened by adversarial attacks. In particular, some recent papers clearly showed that IDSs based on ML algorithms can be misled through the insertion of well-crafted noise within the training examples. This attack can be exploited to introduce errors in the classification model and significantly reduce its accuracy [116], [105], [27], [131], [81]. This adversarial setting is also known to produce *poisoning* attacks [102] or *causative* attacks [8].

A solution may be the use of machine learning algorithms or learning frameworks robust to malicious noise in the sample data. We are able to identify two complementary defense strategies, described as follows.

Training Data Manipulation: Training data can be manipulated to reduce the influence of poisoning attacks over the learning algorithm. To the best of our knowledge, the *Reject On Negative Impact* (RONI) [103] technique is one of the first approaches adopting this defense strategy. Its basic operation is as follows. The original training data is processed to generate different data sets by including or excluding a test sample. Then, a detector is trained on each data set. A test sample is rejected (i.e., classified as a poisoning attack) if its presence in the training data significantly worsen the detector's performance (i.e., it has a negative impact on the detection performances). While the RONI technique has been proposed in the context of spam filtering, the underlying approach can be used in other scenarios. However, it is worth noting that RONI is best suited when incremental learning algorithms are employed, i.e., algorithms whose learning function is estimated by adding a sample at a time. On the other hand, its use with learning algorithms that perform batch processing of the training data may be computationally expensive.

Another solution can be based on the correlation of multiple instances of the detector, each instance being obtained by training the detection algorithm with a randomly sampled portion of the original training data. Each dataset obtained by randomly sampling the original training data may be further filtered by selecting only (or sampling with higher probability) patterns that are evaluated as "representative" of their class (i.e., intrusive, or legitimate). The underlining hypothesis of this approach is that patterns related to poisoning attacks are different from representative patterns of legitimate and intrusive events, since the goal of poisoning attacks is to prevent the accurate discrimination between the two classes. For each class, representative patterns can be identified through statistical significance tests [164] or kernel density estimators [11]. All the above approaches are general, in the sense that they can potentially cope with poisoning attacks *regardless* the specific learning algorithm employed.

Robust Learning Algorithms: Early works on poisoning attacks focused on their influence on "Probably Approximately Correct" (PAC) learning algorithms [78], a class of algorithms introduced in [157]. Novel learning algorithms that are robust against poisoning attacks have been recently proposed. Interesting solutions are based on boosting [135], and on the framework of robust statistics [131]. Boosting can smooth the influence of each single sample on the learned model. In this way, the influence of poisoning attacks can be reduced, since their percentage in the training data is likely to be relatively small, as the adversary may have only a limited control over training data collection. On the other hand, the framework of robust statistic provides a set of statistical measurements that are intrinsically robust to outlying observations in training data. For instance, statistical estimators such as the mean and variance can be substituted by their robust versions: median and median absolute deviation (MAD) [131]. This approach can be useful to develop new learning algorithms as well as robust versions of known algorithms.

The above approaches are effective only if the percentage of poisoning patterns is small with respect to representative patterns either related to intrusions or legitimate activities[7]. However, the validity of such an assumption depends on the source of training data. In anomaly-based IDS, the source of information is the set of legitimate users, i.e., users complying with security policies, while in misuse-based systems the source of information is the knowledge of past attacks. In the former case, the higher the amount of legitimate traffic, the higher the amount of malicious traffic the adversary has to generate in order to reach a target percentage of malicious noise.

This may not be true for misuse-based systems. Let us refer to the case of honeypots. Honeypots can be used to identify suspicious network traffic, such as traffic generated by computer worms. The basic assumption that underlies the success of honeypots is that the adversary attacks them as they were normal computers. Skilled adversaries may be able to recognize them [62], [69], and inject an unbounded percentage of fake, poisoning attack patterns to mislead machine learning algorithms based on malicious traffic collected by honeypots [95], [116], [105], [57]. This can be a significant vulnerability when learning in the presence of adversaries [158].

On the other hand, the actual amount of poisoning attacks in the training data depends on the cost of devising/performing such attacks, and the expected advantages for the adversary.

IV. Response

Automated responses are needed to *protect* the monitored assets, as well as to reduce the effort spent by security analysts in deploying timely and effective countermeasures [161]. Protection against *automated* attacks requires counteractions at computer speed, e.g. the automatic set up of firewalls or routers rules [70], [71], to keep malicious traffic from reaching the monitored hosts. It is worth noting that while research activities have been mainly focused on developing effective detection solutions, research efforts in intrusion response are still isolated [148]. Indeed, effective responses require a good and reliable description of intrusive events, which in itself is a challenging problem (see Section III). Besides that, we identify some relevant *adversarial* problems in the automation of response mechanisms, outlined in the following Sections.

A. *Response Effectiveness*

A key issue when providing for intrusion responses is whether and how much they are effective against the adversary. Indeed, if they are ineffective, they are worthless, and potentially dangerous. By definition, this information is not available for new intrusion techniques, e.g., those detected through anomaly-based systems. This is perhaps the most challenging issue for the evaluation of the effectiveness of intrusion responses. Besides, this is also a challenging issue for well-known intrusions. The reason for such a difficulty relies on the need for deep human expertise regarding known security vulnerabilities and possible countermeasure(s). In addition, to the best of our knowledge, there are no standard methods to perform such an evaluation.

Nevertheless, in the following we identify some interesting research lines that can be helpful to correctly devise automatic responses and assess their effectiveness.

Game Theory: Intrusion detection can be formulated as a two-players game: the adversary aims to violate the security policies, and the defender (IDS) aims to keep computer assets in a safe state. Thus, the application of game theory to intrusion detection seems a natural way to provide for automated responses [13], [74], [172].

The correct formulation of the intrusion detection problem within a game-theoretical framework is perhaps one of the most difficult tasks. In fact, IDS responses should be thoroughly tuned according to the detected intrusion attempts. To this end, the IDS must correctly map vulnerabilities being exploited with possible countermeasure(s). The definition of this mapping should be driven by security experts, as it requires a detailed knowledge of different security problems. Moreover, according to the game-theoretical framework, this mapping may take into account (a) the definition of an attacker model, (b) the costs and benefits of the adversary when attacking a particular asset, (c) the actual traceability of the adversary when an attack is detected, (d) the value of each protected asset for the defender and its vulnerability [128], [97].

Response Frameworks: Different frameworks and infrastructures for intrusion response have been proposed [132], [80], [60], [153], and the interested reader can find an overview of the major approaches in [148]. Understanding the pros and cons of each framework is helpful to select the most suitable solution for the security scenario at hand. For instance, response frameworks can be chosen according to characteristics such as the latency, the level of automation, and the ease of implementation.

[7]Otherwise, the discriminant features of poisoning attacks has to be learned in order to remove them, or reduce their impact over the learning algorithm.

Cost-sensitive models: Effective intrusion responses can be devised according to some cost-sensitive model, i.e., a model which takes into account the costs associated to a successful intrusion versus the execution of a certain counteraction. In this way, responses can be focused on the most relevant assets, and thus they can be more effective against an adversary. Cost-sensitive models may also take into account alert confidence to reduce the damage costs due to false alarms or erroneous alert descriptions. For instance, automatic responses may be triggered only when the detection is characterized by a high level of confidence. As mentioned in Section III, such an approach should take into account a thorough evaluation and characterization of assets that should be protected [93], [152].

Response Time: It is easy to see that the overall time needed by the IDS to detect and react to an intrusion attempt (i.e., the IDS response time) should be smaller than the time needed by the adversary to violate the monitored systems. If this is not the case, the IDS response is not adequate to cope with the intrusion[8]. Moreover, in many real world cases IDS are deployed *inline*, and network traffic could be blocked within the time frame needed by the IDS to assess its legitimacy [10]. Thus, a very low response time (latency) is a "must" to guarantee high levels of quality of service. Finally, a worst-case bound should be guaranteed for the response time in order to avoid the exposure of the IDS to DoS attacks, such as flooding (see Section II) or algorithmic complexity attacks (see Section III).

B. *Response Feedback*

The adversary may stimulate the IDS with a set of well-known attacks and look for the corresponding (re)actions, using a reverse engineering approach to infer the response rules employed by the IDS [100]. For instance, if the IDS is programmed to block the traffic deemed to be malicious, the adversary may be able to understand the set of attacks detected by the IDS by just looking at the characteristics of the blocked traffic [10], [64]. That is, response mechanisms may give a relevant *feedback* to the adversary. By means of a query-response strategy, the adversary may potentially infer the set of models or algorithms employed to perform the intrusion detection task. This information, as mentioned in Section III, can be used to evade or overstimulate the IDS, or even hijack the IDS response.

In fact, most of the current commercial IDS products, as well as most of the IDS solutions proposed by academic research groups, deal with the response problem by merely adopting defensive solutions, e.g., by enforcing rules for blocking/dropping malicious traffic, or by bounding the bandwidth assigned to suspect connections [148], [64], [29]. Nevertheless, we think that IDSs responses could be tuned to be *stealthy* and *proactive*. For instance, the IDS could *substitute* in real time a malicious request with a known-as-legitimate request, e.g., stripping the malicious content. In this case, for the adversary it may be difficult to distinguish ineffective attacks from those detected (and stealthily modified in real time) by the IDS. Moreover, if an intrusion is detected, the IDS may return well-crafted data that can include fake information to mislead the attacker, and profile his/her behavior. Thus, we believe that proactive approaches to intrusion response would make IDS solutions "smarter" and effective against adversaries.

[8]In the meantime, the attacker may even corrupt/disable host sensors (see Section II).

C. *Response Evaluation*

Response evaluation is a fundamental aspect in IDS design, but it is still an open problem. It would require the simulation of known and hypothetical intrusion techniques against the protected assets, *including* attacks against the IDS itself, and the evaluation of (a) the increased cost for the adversary when the IDS solution is deployed, (b) the impact on the normal operation of the monitored systems. To the best of our knowledge, there is no previous work that addressed this problem as a whole. Nevertheless, some cost-sensitive evaluation metrics for intrusion response have been proposed [152]. We think that these metrics could be used as a starting point for a comprehensive, adversary-aware evaluation of intrusion responses.

V. Discussion

IDS solutions operate in an environment characterized by a skilled, adaptive adversary, who may severely undermine IDS capabilities, and even turn them into unconventional attack tools. This aspect is crucial, especially in safety-critical computer systems. In Section I-B we outlined six main attack categories against IDSs, namely, evasion, overstimulation, poisoning, denial of service, response hijacking, and reverse engineering. Such attack categories can exploit a wide variety of vulnerabilities which can be clearly associated to different IDS processing phases, namely, measurement (Section II), classification (Section III), or response (Section IV). In the following, for each phase, we summarize its main vulnerabilities, how attackers can exploit them, and some solutions that can reduce the impact of their threat. To get a better understanding of the impact of each attack, we highlight the main results in the relevant research literature in terms of IDS attack simulation. We also present the principal results of the solutions that can deal with these attacks, to get insights into their effectiveness.

Measurement (Figure 7, Tabs. II, III, IV): Even if the selected set of measurements allow for accurately distinguishing *known* intrusion instances from legitimate activities, it may be very inadequate for detecting novel or even *small* variations of known intrusion instances. In order to achieve robustness against evasion, the set of measurements should be capable of capturing only the *invariant* aspects of intrusive activities, discarding all the measures loosely related to the specificity of the attack. In other terms, deep knowledge of the specific security problem is always needed to select evasion-proof measurements.

Regardless the discriminant capability of measurements, attackers may induce an erroneous generation of event patterns, either (1) by introducing errors in the raw (input) data collected by IDS sensors, or (2) by exploiting event reconstruction

TABLE II
RELEVANCE OF INPUT DATA CORRUPTION (MEASUREMENT PHASE), AND EFFECTIVENESS OF THE RELATED SOLUTIONS.

Input Data Corruption *Problem relevance*
[21] show that host sensors are vulnerable to DKOM techniques (43% of evasion success). *Solutions and their effectiveness*
[21] show that kernel integrity checking can detect *all* known DKOM techniques.

TABLE III
RELEVANCE OF THE "SEMANTIC GAP" PROBLEM (MEASUREMENT PHASE), AND EFFECTIVENESS OF THE RELATED SOLUTIONS. DR=DETECTION RATE, FP=FALSE POSITIVE RATE.

Semantic gap between collected data and measurements of interest *Problem relevance*
[72] show that parsing heuristics employed by host sensors for file processing can be evaded with up to 97% of success. [59] describe *tunneling*, *desynchronization*, *segmentation*, *encoding variations* attacks against network sensors, however, to the best of our knowledge, there is still no systematic evaluation of the impact of flawed network event reconstruction on different IDS solutions. *Solutions and their effectiveness*
As highlighted by [72] *tight integration* is the key solution, but its implementation depends on the specific intrusion detection problem. [127] propose a method for *automatic network sensor placement* that can spot all attacks whose formal definition is available. [136] show that *active mapping* can detect all fragmentation attacks (devised through `fragroute`): the mapping process requires about 37 seconds per host and network traffic reconstruction is 15% faster under attack. [38] demonstrate how *protocol normalization* can successfully defeat desynchronization/segmentation attacks for `HTTP`, `FTP`, `SMTP` protocols introducing about 16% overhead. *Data Enhancement*: [120] show that CPU emulation can accurately (DR=94%, FP=0%) spot polymorphic shellcode within network traffic; [118] demonstrate that polymorphic instances of HTTP-based malware can be accurately detected (DR=85.9% and FP=0%) by analyzing network traffic; In a real-world deployment, [43] leveraged on *dynamic protocol analysis* to find 710 new servers, out of 73,818 known servers, whose {`HTTP`/`FTP`/`SMTP`/`IRC`} traffic would otherwise be missed.

flaws. We observe that, while an erroneous generation of event patterns has mostly been exploited to conceive *evasion* attacks, it can also be exploited to conceive *overstimulation*, *poisoning*, *response hijacking* and *reverse engineering* attacks. This is an important issue, that highlights the threat posed by such kind of errors: any adversarial manipulation of event patterns can propagate through subsequent processing phases, and lead to classification/response errors, or allow attackers to gain information about the internal processing of an IDS.

While attack tactics that introduce errors in input data have been documented at the host-level only, we speculate that network sensors may be affected as well by this kind of attacks. Hence, the feasibility of these attacks at the network-level should be further investigated. In general, depending on the IDS input data, a tailored solution for assessing/enhancing the reliability of input data should be designed. For instance, kernel integrity checking [21] can be used to detect attack techniques that tamper the OS kernel data structures, such as DKOM. In addition, the integrity of OS and user applications can be enforced by preventing root abuse of file-system privileges [167], or by preventing non-trusted kernel components from altering data employed by the kernel to manage its own execution, through a hypervisor-based solution [147].

In general, the larger the *semantic gap* between the collected data and the related measurements of interest, the larger the emulation complexity and the likelihood of event reconstruction errors. The semantic gap can be reduced through tight integration between IDS sensors and monitored systems (e.g., applications, OS kernel, other services). Unfortunately, tight integration may be difficult to implement, especially with a large number of hosts and services (e.g., large server farms). In this case, it is important to design event reconstruction mechanisms aimed at keeping input data interpretation as close as possible to that performed by the monitored systems, even in the presence of unexpected values. To this end, we presented a number of general techniques that can strengthen event reconstruction from network data, namely, *sensor placement*, *compartmentalization*, *traffic normalization*, *data enhancement*, *bifurcating analysis*, *active mapping*, *dynamic protocol analysis*. In particular, the correlation of data at different abstraction levels (e.g., network and host level) can further strengthen measurements. Redundancy in the set of measurements can support a more reliable evaluation of computer system events (data reconciliation techniques) and possibly spot evasion/overstimulation attacks that attempt to leverage on measurement flaws.

Finally, IDS sensors can be inhibited through flooding attacks, e.g., overloading a network sensor with an amount of traffic that exceeds its bandwidth, or even disabled, e.g., once the monitored host has been compromised (if sensors are running in the same host they are protecting). Protection against flooding attacks can be achieved by extending sensors' bandwidth. This is one of the reasons why the vast majority of commercial IDS solutions are implemented as standalone devices, with dedicated hardware and software. Robustness against flooding and event reconstruction attacks may be the result of a tradeoff with some performance constraints. For instance, in order to save bandwidth, heuristics can be used to proactively drop packets that are less likely to affect IDS accuracy (selective packet dropping) [110].

On the other hand, the integrity of IDS sensors can be strengthened by isolating them from the systems being monitored. General solutions are actually based on virtualization techniques, where the monitored host runs in a virtual environment and a higher privileged *hypervisor* enforces memory protections and intercepts the events within the virtual machine. In general, we observe that there is a tradeoff between isolation and semantic gap for IDS sensors. The more the IDS sensors are isolated, the more the difficulty of bridging the semantic gap between the collected data and the measurements of interest. We think that a very interesting tradeoff is employed by in-VM monitoring solutions, where sensors are left within the guest machine, but they are executed on a hypervisor protected address space, and access data from this location [137]. This way, the semantic gap can be significantly reduced (tight integration), and isolation can be guaranteed by enforcing memory protections on IDS sensors. We believe that future research work should be focused on further improving this architecture, e.g., by including protection mechanisms

TABLE IV
RELEVANCE OF DOS (AVAILABILITY/INTEGRITY) ATTACKS AGAINST THE MEASUREMENT PHASE, AND EFFECTIVENESS OF THE RELATED SOLUTIONS. DR=DETECTION RATE, FP=FALSE POSITIVE RATE.

Denial of Service against Measurement *Problem relevance*
[110] show that under high traffic load (900 Mbit/sec), the detection rate of `Snort` can drop to 55%. [137] and [61] highlight how current malware can take complete control of the protected hosts, thus compromising host sensors reliability. *Solutions and their effectiveness*
[5] show that, in order to save memory and packet processing time, fragmentation attacks can be detected without packet reassembly (up to DR=99.8% and FP=0.5%) [110] demonstrate that *selective packet dropping* can significantly improve detection rate under high traffic load (96.3% versus 55%). [137] propose *in-VM monitoring* to protect host sensors from DoS attacks: very small overhead is observed compared to traditional VMI approaches on introspection-heavy security applications (13.7% vs 690.5%).

TABLE V
RELEVANCE OF EVASION/OVERSTIMULATION ATTACKS AGAINST THE CLASSIFICATION PHASE, AND EFFECTIVENESS OF THE RELATED SOLUTIONS. DR=DETECTION RATE, FP=FALSE POSITIVE RATE.

Difference between Alert and Intrusion Spaces *Problem relevance*
[159] show that misuse-based IDSs can be evaded devising variations of known attacks: from 60% (`Snort` IDS) to 90% (`IBM ISS Realsecure`) evasion success. [56] demonstrate that polymorphic worms can completely evade the `Hamsa` and `Polygraph` misuse-based detectors. [101] show that 76.3% of `Snort v.1.8.6` signatures can be used to automatically build successful overstimulation attacks. [75] show that the `Stide` and `pH` anomaly detectors can be evaded (DR reduction of 44%) through mimicry attacks. [49] show that the `PAYL` anomaly detector can be completely evaded through mimicry attacks. *Solutions and their effectiveness*
[85] show that *alert verification* can narrow down false alerts from 99.64% to 0% (simulation using Snort v.2.0.2), and correctly label as unsuccessful 161,166 attacks against `Linux` hosts (98.3%), and 78,785 attacks against `Windows` hosts (99.4%). [14] show that *alert verification* can yield a reduction of false positives between 50% and 100% (test on `Snort` and `Poseidon` IDSs) without introducing false negatives. [16] employs a *proactive approach* (injection of fake legitimate traffic) that yields 100% DR on malware performing mimicry attacks. [117] show that an *ensemble* composed by 11 classifiers can significantly outperform the single best classifier (DR=99.2% and FP=0.49% against DR=97.6% and FP=11.25%) and successfully detect mimicry attacks.

TABLE VI
RELEVANCE OF ALERT DESCRIPTION ERRORS, AND EFFECTIVENESS OF THE RELATED SOLUTIONS. DR=DETECTION RATE, FP=FALSE POSITIVE RATE.

Erroneous Alert Descriptions *Problem relevance*
As showed by [101], evasion attacks can generate an incorrect or too generic alert description. Even if complete evasion is not accomplished, the security analyst may be unable to understand either the targeted vulnerability, or the real threat posed by the attack, or the actual attacker's goal. *Solutions and their effectiveness*
[108] show that alert descriptions can be improved through the estimation of *classification confidence*, and if there is *not* a high number of sensors characterized by very poor detection capabilities, 80% or 100% (all) classification errors can be eliminated. [15] devise a automatic *alert inference* mechanisms that is able to achieve 95% of accuracy in the automatic assignment of attack descriptions to anomalous events (100% accuracy for some attack classes) . [55] employ a *model of the adversary* that is able to process alerts to accurately distinguish real intrusive events (DR=96.5%) from probing events or legitimate activities.

against input data corruption.

Classification (Figure 8, Tabs. V, VI, VII, VIII): Even if event patterns (measurements) allow to *perfectly* distinguish between intrusions and legitimate activities, attackers may exploit vulnerabilities in the classification function. Any difference between alert and intrusion spaces can be exploited to evade or overstimulate the IDS. Classification errors can be reduced through the employment of classifier ensembles, by complementing misuse with anomaly detection, and by employing proactive approaches such as the injection of fake legitimate traffic to mislead mimicry attacks. In order to fight overstimulation attacks (and narrow down false alarms), contextual information about systems being monitored as well as other intrusion detectors can be exploited to perform alert verification, and alert correlation, respectively. Moreover, overstimulation attacks can be potentially addressed either if the IDS is able to categorize these events as attacks with low damage costs, through cost sensitive classification, or if the IDS employs a model of the adversary that allows classifying these events as attacks against the IDS itself. The actual classification accuracy can be evaluated through the development of automatic variations of known intrusions, or by devising automatic overstimulation attacks. In particular, we observe that overstimulation attacks have been investigated and implemented in the context of misuse detection only. On the other hand, automatic overstimulation against anomaly detection or discriminative classification models (such as those built through supervised machine learning) is indeed feasible, and it would be an interesting line of research.

Even if the adversary is not able to evade the IDS, he may modify an attack instance so as to generate an incorrect or too generic alert description. This way, the security analyst may be unable to understand either the targeted vulnerability, or the real threat posed by the attack, or the actual attacker's goal. An incorrect alert description may also cause an erroneous response by the IDS or by the security analyst. This problem can be approached by estimating the confidence in the accuracy of alert descriptions, and through an accurate model of the adversary. In addition, alert descriptions in anomaly-based systems may be improved through automatic attack inference mechanisms.

Similarly to flooding attacks targeted at the measurement phase, an adversary may also perform DoS attacks targeted at the classification phase, e.g., by means of algorithmic complexity attacks. A defense against algorithmic complexity attacks can be devised by employing rule matching algorithms that either guarantee bounds on worst-case performance, or whose worst-case behavior is not predictable [36]. Rule-matching algorithms may also be strengthened by keeping track of intermediate matching results [139], and by significantly speeding it up by compressing the size of the discrete state automata corresponding to the overall set of rules [111].

Finally, if automatic learning mechanisms are employed for model inference, they should be explicitly devised to be robust against poisoning attacks. Two main complementary

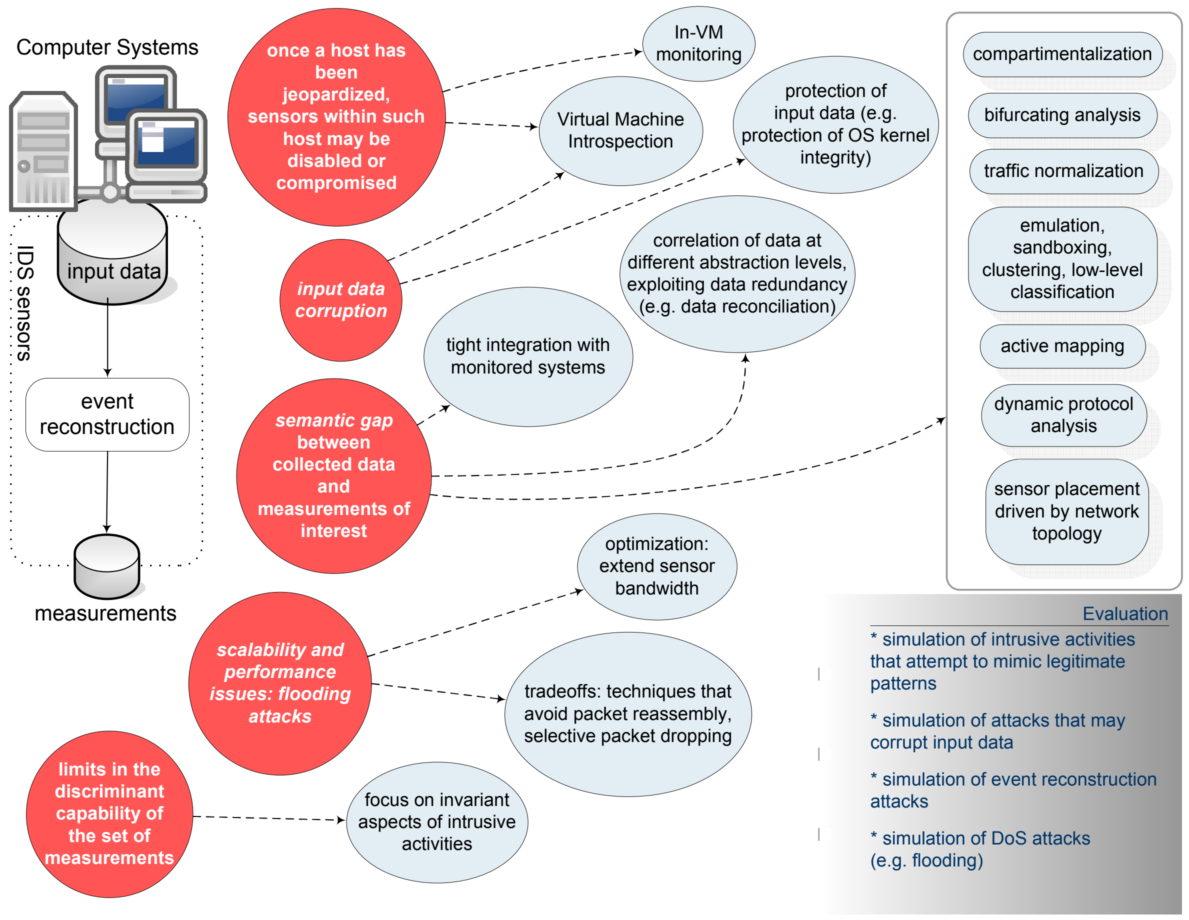

Computer Systems
IDS sensors
input data
event reconstruction
measurements
once a host has been jeopardized, sensors within such host may be disabled or compromised
input data corruption
semantic gap between collected data and measurements of interest
scalability and performance issues: flooding attacks
limits in the discriminant capability of the set of measurements
In-VM monitoring
Virtual Machine Introspection
protection of input data (e.g. protection of OS kernel integrity)
correlation of data at different abstraction levels, exploiting data redundancy (e.g. data reconciliation)
tight integration with monitored systems
optimization: extend sensor bandwidth
tradeoffs: techniques that avoid packet reassembly, selective packet dropping
focus on invariant aspects of intrusive activities
compartimentalization
bifurcating analysis
traffic normalization
emulation, sandboxing, clustering, low-level classification
active mapping
dynamic protocol analysis
sensor placement driven by network topology
Evaluation
* simulation of intrusive activities that attempt to mimic legitimate patterns
* simulation of attacks that may corrupt input data
* simulation of event reconstruction attacks
* simulation of DoS attacks (e.g. flooding)

Fig. 7. Summary of IDS measurement vulnerabilities. Dotted arrows indicate some proposed solutions.

TABLE VII
RELEVANCE DOS ATTACKS AGAINST THE CLASSIFICATION PHASE, AND EFFECTIVENESS OF THE RELATED SOLUTIONS. DR=DETECTION RATE, FP=FALSE POSITIVE RATE.

DoS attacks against Classification *Problem relevance*
[139] devise algorithmic complexity attacks that make pattern matching up to 1.5 million times slower on `Snort v.2.4.3`, and that are able to completely *evade* detection. *Solutions and their effectiveness*
By keeping track of intermediate matching results, [139] is able to bound worst-case performance of `Snort v.2.4.3` (max 8 times slower), and keep 100% DR when algorithmic complexity attacks occur. [111] show that by compressing the size of the discrete finite automata (DFA) used for pattern matching, a 300 times speedup (1882.1 times less memory) on `Snort` can be yielded.

TABLE VIII
RELEVANCE OF POISONING ATTACKS, AND EFFECTIVENESS OF THE RELATED SOLUTIONS. DR=DETECTION RATE, FP=FALSE POSITIVE RATE.

Poisoning attacks against Classification *Problem relevance*
[116] show that the injection of one poisoning worm for each worm sample, i.e., a 50% noise level in the set of intrusive examples, allows to make the `Polygraph` (misuse-based IDS) useless. [11] show that only 1% of thoroughly crafted noise can render an anomaly-based systems useless (experiments performed on a simplified version of the `HMM-Web` anomaly detector). *Solutions and their effectiveness*
[103] propose the Reject On Negative Impact (RONI) technique: it is able to filter out 100% of poisoning attacks with 0% FP (poisoning fraction=5%) [11] show that *weighted bagging* can successfully handle 20% of poisoning attacks in the training set, without affecting IDS accuracy [131] show that robust learning algorithms can significantly improve detection under poisoning attack: DR=90% versus DR<50% (for FP=1% and poisoning fraction=10%).

approaches can be used: training data pre-processing, and development of machine learning algorithms robust to noise. In the former case, training data is processed in such a way that the influence of poisoning patterns over the learning algorithm is reduced. This approach is interesting since it may cope with poisoning attacks regardless the employed learning algorithm. On the other hand, learning algorithms can explicitly cope with poisoning attacks either through ad-hoc procedures (e.g., the RONI technique [103]), or by exploiting the framework of robust statistics (e.g., outlier-resistant statistical estimators).

Response (Figure 9, Tab. IX): Responses against intrusions should reveal as little information as possible about the employed intrusion detection algorithm, and should possibly employ proactive methods aimed at misleading the interpretation of responses by the adversary. Cost-sensitive models, game theory, well-suited response frameworks, and response

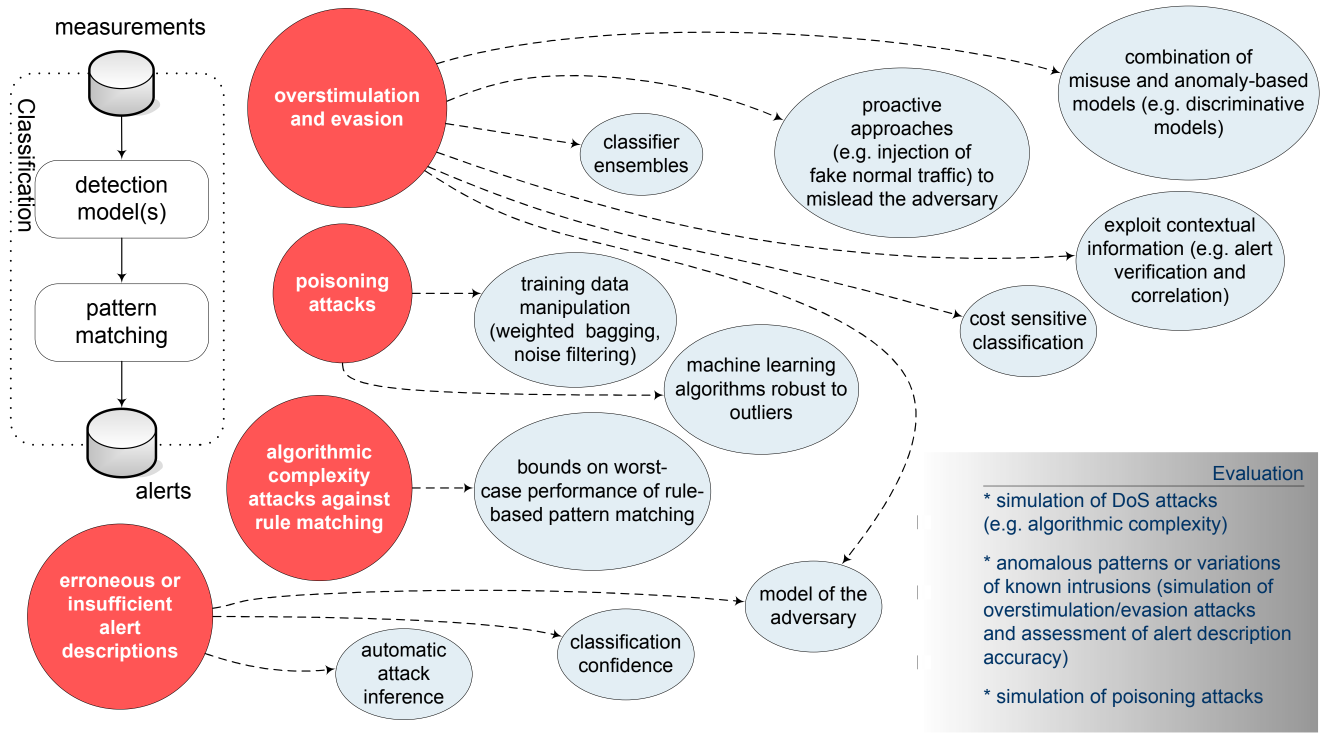

measurements
Classification
detection model(s)
pattern matching
alerts
overstimulation and evasion
poisoning attacks
algorithmic complexity attacks against rule matching
erroneous or insufficient alert descriptions
classifier ensembles
proactive approaches (e.g. injection of fake normal traffic) to mislead the adversary
combination of misuse and anomaly-based models (e.g. discriminative models)
exploit contextual information (e.g. alert verification and correlation)
cost sensitive classification
training data manipulation (weighted bagging, noise filtering)
machine learning algorithms robust to outliers
bounds on worst-case performance of rule-based pattern matching
model of the adversary
classification confidence
automatic attack inference
Evaluation
* simulation of DoS attacks (e.g. algorithmic complexity)
* anomalous patterns or variations of known intrusions (simulation of overstimulation/evasion attacks and assessment of alert description accuracy)
* simulation of poisoning attacks

Fig. 8. Summary of classification vulnerabilities. Dotted arrows indicate some proposed solutions.

time optimization can be exploited for the selection of effective responses against an adversary, and for the reduction of the costs due to false alarms and overstimulation attacks. Preliminary results show the feasibility of these solutions, but more research efforts are needed to allow them to be easily implemented in operating environments. Indeed, as mentioned in Section IV, research efforts in intrusion response are still limited, and we believe that there is significant room for improvement.

Finally, response evaluation is a fundamental issue in IDS design, but it is still an open problem. Indeed, a thorough evaluation would require the simulation of attack techniques against the IDS and its monitored assets, and the evaluation of (a) the increased cost for the adversary when the IDS solution is deployed, (b) the impact on the normal operation of monitored systems. To the best of our knowledge, there is no previous work that addressed this problem as a whole. Nevertheless, we think that cost-sensitive evaluation metrics proposed in [152] could be exploited as a starting point for a comprehensive evaluation of adversary-aware intrusion responses.

TABLE IX
RELEVANCE OF RESPONSE ATTACKS, AND EFFECTIVENESS OF THE RELATED SOLUTIONS.

Attacks against IDS Response
Problem relevance
[93] show that automatic responses are necessary, but it is quite difficult to judge their effectiveness. Erroneous responses yield damage costs that should be correctly evaluated. In addition, automatic responses can be exploited by the adversary to gain relevant feedback about the internal processing of the IDS.
Solutions and their effectiveness
[93] show that *cost-sensitive* classification and response can yield a reduction of about 10% of the cumulative cost of an IDS with respect to cost-insensitive models. [13] show that *game theory* can be exploited to support security administrators (effort reduced up to 59.64%) when dealing with intrusive activities that are not automatically stopped. [153] show that exploiting the knowledge of previous alerts and responses from multiple intrusion detectors allows to deploy *cost-effective responses* and reduce false negative rate (-44%) and false positives (-14%). [152] show that cost-sensitive responses can yield a significant reduction (up to 25%) over IDS costs.

VI. Conclusions and Promising Research Directions

Intrusion Detection Systems are nowadays recognized as fundamental tools for the security of computer systems. IDSs aim at identifying violations of security policies and perform automatic counteractions to protect computer systems and information. As soon as IDSs are deployed, they may become target of attacks that may severely undermine or mislead their capabilities. To the best of our knowledge, this paper is the first survey on adversarial attacks against IDSs, a relevant topic especially for safety-critical environments. In this paper we provided the following contributions: (1) we provided a general taxonomy of attack tactics against Intrusion Detection Systems; (2) we subdivided the IDS task into three different phases, namely, measurement, classification and response, to clearly outline different ways by which attack tactics can be implemented; (3) for each attack implementation, we critically analyzed proposed solutions and open issues.

Moreover, throughout the paper we identified a number of challenging issues that should be addressed by future research activities on intrusion detection. We focus our attention on a few of them:

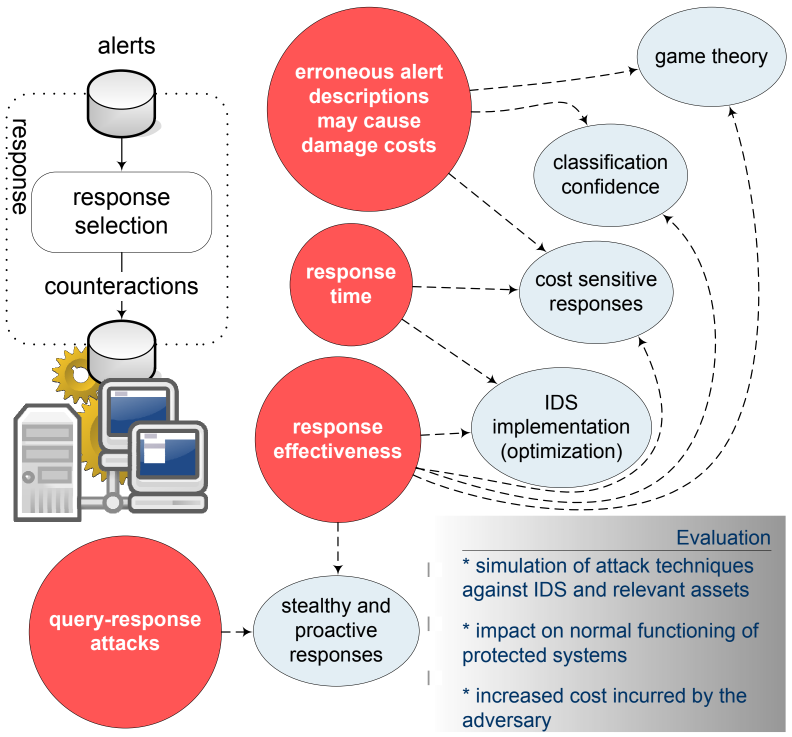

alerts
response
response selection
counteractions
erroneous alert descriptions may cause damage costs
response time
response effectiveness
query-response attacks
game theory
classification confidence
cost sensitive responses
IDS implementation (optimization)
stealthy and proactive responses
Evaluation
* simulation of attack techniques against IDS and relevant assets
* impact on normal functioning of protected systems
* increased cost incurred by the adversary

Fig. 9. Summary of the problems of automatic response mechanisms in intrusion detection. Dotted arrows indicate some proposed solutions.

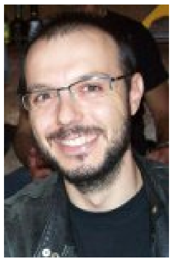
- Strengthening the measurement mechanisms by relying on both host and network sensors, and exploiting the concept of redundancy as performed by data reconciliation techniques in process control. In addition, in-VM monitoring showed to be a very promising way to strengthen measurements at the host level.
- Enhancement of the description of alerts in anomaly-based systems through automatic attack inference mechanisms. This may definitely cope with the lack of informative output in anomaly-based systems, that may allow for the detection of variants of known, or never-before-seen intrusions. Moreover, exploiting contextual information about the systems being monitored (e.g., for performing alert verification) seems the natural way to deal with overstimulation attacks, as well as false alarms in general.
- Responses against intrusions based on cost-sensitive models, game theory and proactive techniques should be further investigated. Human expertise will always play a central role, but these methods can be helpful to automate the response process and make it effective against an adversary.
- IDS solutions are expected to increasingly implement machine learning mechanisms, to deal with the complexity of the intrusion detection task. Consequently, techniques based on adversarial machine learning are worth being further investigated.

Acknowledgments

The authors would like to thank the anonymous reviewers, and Davide Maiorca, whose valuable insights helped us to improve the quality of this manuscript. This work was partly supported by a grant from Regione Autonoma della Sardegna awarded to I. Corona, PO Sardegna FSE 2007-2013, L.R.7/2007 “Promotion of the scientific research and technological innovation in Sardinia”, Project PRR-MAB-A2011-19112 “Tecniche Avanzate per la Rilevazione di Attacchi Informatici”, and Project “Advanced and secure sharing of multimedia data over social networks in the future Internet” (CUP F71J11000690002).

About The Authors

Igino Corona



Igino Corona is Post Doc Researcher in the Pattern Recognition and Applications Group, Dept. of Electrical and Electronic Engineering, University of Cagliari, Italy. Igino Corona received both M.Sc. in Electronic Engineering and PhD Degree in Computer Engineering from the University of Cagliari. His main research interests are about computer security, intrusion detection and pattern recognition.

In his MSc thesis, he discussed the design and the implementation of an anomaly-based, unsupervised Intrusion Detection System for the analysis of the HTTP traffic. The Clusit Association awarded that work as one of the best Italian research thesis on computer system security. In 2009, Igino Corona worked with the research group headed by Prof. Wenke Lee (Georgia Institute of Technology, Atlanta, USA) as a visiting PhD student. During such a period, Igino Corona and Roberto Perdisci (Assistant Professor, Department of Computer Science, University of Georgia, USA) developed Flux Buster, an advanced system which is able to detect fast flux service networks by means of passive analysis of DNS traffic in large networks. Igino Corona is also the author of SuStorID, an advanced intrusion detection system for web services based on machine learning, released in January 2012 under open source license.

Igino Corona is manager of the Computer Security Technical Committee of GIRPR (Italian Group of Pattern Recognition researchers) and one of the organizers of the International School on Computer Security & Privacy.

Giorgio Giacinto

Giorgio Giacinto is Associate Professor of Computer Engineering at the University of Cagliari, Italy. His main research interests are in the area of pattern recognition and its application. His main contributions are in the field of combination (a.k.a. fusion) of multiple classifiers, computer security, and multimedia retrieval. Giorgio Giacinto also contributes to researches in the fields of biometric personal authentication, video-surveillance, and remote sensing image classification. Giorgio Giacinto is author of more than ninety papers in international journals and conference proceedings, including six book chapters. He is currently serving as associate editor of the "Information Fusion" journal. Giorgio Giacinto is involved in several technical committees of international workshops and conferences on pattern recognition and applications, and regularly serves as a reviewer. He is a Senior Member of the ACM and the IEEE.

Fabio Roli

Fabio Roli received his M.S. degree, with honours, and Ph.D. degree in Electronic Engineering from the University of Genoa, Italy. He was a member of the research group on Image Processing and Understanding of the University of Genoa, Italy, from 1988 to 1994. He was adjunct professor at the University of Trento, Italy, in 1993 and 1994. In 1995, he joined the Dept. of Electrical and Electronic Engineering of the University of Cagliari, Italy, where he is now professor of computer engineering and head of the research group on pattern recognition and applications. Dr Roli's research activity is focused on the design of pattern recognition systems and their applications to biometric personal identification, multimedia text categorization, and computer security. On these topics, he has published more than two hundred papers at conferences and on journals. He was a very active organizer of international conferences and workshops, and established the popular workshop series on multiple classifier systems. He is a member of the governing boards of the International Association for Pattern Recognition and of the IEEE Systems, Man and Cybernetics Society. He is Fellow of the IEEE, and Fellow of the International Association for Pattern Recognition.

References

[1] H. Aljifri, M. Smets, and A. Pons, "Ip traceback using header compression," *Computers & Security*, vol. 22, no. 2, pp. 136–151, 2003.

[2] M. Allman, P. Barford, B. Krishnamurthy, and J. Wang, "Tracking the role of adversaries in measuring unwanted traffic," in *SRUTI'06: Proceedings of the 2nd conference on Steps to Reducing Unwanted Traffic on the Internet*. Berkeley, CA, USA: USENIX Association, 2006, pp. 6–6.

[3] M. Almgren, U. Lindqvist, and E. Jonsson, "A multi-sensor model to improve automated attack detection," in *Recent Advances in Intrusion Detection*, ser. Lecture Notes in Computer Science, R. Lippmann, E. Kirda, and A. Trachtenberg, Eds. Springer Berlin / Heidelberg, 2008, vol. 5230, pp. 291–310, 10.1007/978-3-540-87403-4_16.

[4] S. Andersson, A. Clark, and G. Mohay, "Detecting network-based obfuscated code injection attacks using sandboxing," in *AusCERT Asia Pacific Information Technology Security Conference (Gold Coast, Australia)*, 2005.

[5] G. Antichi, D. Ficara, S. Giordano, G. Procissi, and F. Vitucci, "Counting bloom filters for pattern matching and anti-evasion at the wire speed," *IEEE Network*, vol. 23, no. 1, pp. 30–35, 2009.

[6] S. Bahram, X. Jiang, Z. Wang, M. Grace, J. Li, D. Srinivasan, J. Rhee, and D. Xu, "Dksm: Subverting virtual machine introspection for fun and profit," in *Proceedings ofthe 29th IEEE International Symposium on Reliable Distributed Systems (SRDS 2010)*, New Delhi, India, October 2010.

[7] M. Barreno, P. L. Bartlett, F. J. Chi, A. D. Joseph, B. Nelson, B. I. Rubinstein, U. Saini, and J. D. Tygar, "Open problems in the security of learning," in *AISec '08: Proceedings of the 1st ACM workshop on Workshop on AISec*. New York, NY, USA: ACM, 2008, pp. 19–26.

[8] M. Barreno, B. Nelson, A. D. Joseph, and J. D. Tygar, "The security of machine learning," *Machine Learning*, vol. 81, no. 2, pp. 121–148, 2010.

[9] I. Basicevic, M. Popovic, and V. Kovacevic, "The use of distributed network-based ids systems in detection of evasion attacks," in *AICT/SAPIR/ELETE*. IEEE Computer Society, 2005, pp. 78–82.

[10] R. Bidou, "Ips shortcomings," in *Black Hat Briefings*, Caesars Palace, Las Vegas, USA, July 29-August 3 2006. [Online]. Available: https://www.blackhat.com/presentations/bh-usa-06/BH-US-06-Bidou.pdf

[11] B. Biggio, I. Corona, G. Fumera, G. Giacinto, and F. Roli, "Bagging classifiers for fighting poisoning attacks in adversarial classification tasks," in *Multiple Classifier Systems - 10th International Workshop, Naples, Italy. Proceedings*, ser. Lecture Notes in Computer Science, vol. 6713. Springer, 2011, pp. 350–359.

[12] B. Biggio, G. Fumera, and F. Roli, "Multiple classifier systems for adversarial classification tasks," in *Multiple Classifier Systems, 8th International Workshop, MCS 2009, Reykjavik, Iceland, June 10-12, 2009. Proceedings*, ser. Lecture Notes in Computer Science, vol. 5519. Springer, 2009, pp. 132–141.

[13] M. Bloem, T. Alpcan, and T. Basar, "Intrusion response as a resource allocation problem," in *Proc. 45th IEEE Conference on Decision and Control*, San Diego, CA, December 2006, pp. 6283–6288.

[14] D. Bolzoni, B. Crispo, and S. Etalle, "Atlantides: an architecture for alert verification in network intrusion detection systems," in *LISA'07: Proceedings of the 21st conference on Large Installation System Administration Conference*. Berkeley, CA, USA: USENIX Association, 2007, pp. 1–12.

[15] D. Bolzoni, S. Etalle, and P. H. Hartel, "Panacea: Automating attack classification for anomaly-based network intrusion detection systems," in *RAID '09: Proceedings of the 12th International Symposium on Recent Advances in Intrusion Detection*. Berlin, Heidelberg: Springer-Verlag, 2009, pp. 1–20.

[16] K. Borders, X. Zhao, and A. Prakash, "Siren: Catching evasive malware," in *IEEE Symposium on Security and Privacy*. IEEE Computer Society, 2006, pp. 78–85.

[17] J. Butler, J. Undercoffer, and J. Pinkston, "Hidden processes: the implication for intrusion detection," in *Information Assurance Workshop, 2003. IEEE Systems, Man and Cybernetics Society*, jun. 2003, pp. 116–121.

[18] K. R. Butler, S. McLaughlin, and P. D. McDaniel, "Rootkit-resistant disks," in *CCS '08: Proceedings of the 15th ACM conference on Computer and communications security*. New York, NY, USA: ACM, 2008, pp. 403–416.

[19] C. Byeong-Cheol, S. Dong-il, and S. Sung-Won, "Two-step rule estimation (tre) - intrusion detection method against evading nids," in *Advanced Communication Technology Conference*, vol. 1. IEEE Computer Society, 2004, pp. 504–507.

[20] M. Carbone, M. Conover, B. Montague, and W. Lee, "Secure and robust monitoring of virtual machines through guest-assisted introspection," in *Proceedings of Research in Attacks, Intrusions, and Defenses - 15th International Symposium, RAID 2012*, Amsterdam, The Netherlands, September 12-14 2012, pp. 22–41.

[21] M. Carbone, W. Cui, L. Lu, W. Lee, M. Peinado, and X. Jiang, "Mapping kernel objects to enable systematic integrity checking," in *CCS '09: Proceedings of the 16th ACM conference on Computer and communications security*. New York, NY, USA: ACM, 2009, pp. 555–565.

[22] L. Cavallaro, A. Lanzi, L. Mayer, and M. Monga, "Lisabeth: automated content-based signature generator for zero-day polymorphic worms," in *SESS '08: Proceedings of the fourth international workshop on Software engineering for secure systems*. New York, NY, USA: ACM, 2008, pp. 41–48.

[23] D. J. Chaboya, R. A. Raines, R. O. Baldwin, and B. E. Mullins, "Network intrusion detection: Automated and manual methods prone to attack and evasion," *IEEE Security and Privacy*, vol. 4, no. 6, pp. 36–43, 2006.

[24] P. K. Chan and R. P. Lippmann, "Machine learning for computer security," *Journal of Machine Learning Research*, vol. 7, pp. 2669–2672, 2006.

[25] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Computing Surveys*, vol. 41, no. 3, pp. 1–58, 2009.

[26] D. Chechik, "Siberia exploits kit fights back against av companies," M86 Security Labs, November 2010, article. [Online]. Available: http://labs.m86security.com/2010/11/siberia-exploits-kit-fights-back-against-av-companies

[27] S. P. Chung and A. K. Mok, "Advanced allergy attacks: Does a corpus really help?" in *Recent Advances in Intrusion Detection, 10th International Symposium*, ser. Lecture Notes in Computer Science, C. Krugel, R. Lippmann, and A. Clark, Eds., vol. 4637. Springer, 2007, pp. 236–255.

[28] Cisco, "Cisco intrusion prevention system sensor cli configuration guide for ips 5.1," Web, March 2013. [Online]. Available: http://www.cisco.com/en/US/docs/security/ips/5.1/configuration/guide/cli/cliInter.html#wp1033986

[29] ——, "Getting started with your cisco ips," White Paper, March 2013. [Online]. Available:

http://www.cisco.com/en/US/prod/collateral/vpndevc/ps5729/ps5713/ps4077/guide_c07-464689_ps6120_Products_White_Paper.html

[30] ——, "Intrusion prevention system (ips)," Web, March 2013. [Online]. Available: http://www.cisco.com/en/US/products/ps5729/Products_Sub_Category_Home.html

[31] ——, "Risk rating and threat rating: Simplify ips policy management," Web, March 2013. [Online]. Available: http://www.cisco.com/en/US/prod/collateral/vpndevc/ps5729/ps5713/ps4077/prod_white_paper0900aecd806e7299.html

[32] J. Cordasco and S. Wetzel, "An attacker model for manet routing security," in *The ACM Conference on Wireless Network Security (WISEC)*, D. A. Basin, S. Capkun, and W. Lee, Eds. ACM, 2009, pp. 87–94.

[33] G. Coretez, "Stick, tool for resource starvation attacks against ids," Web, October 2012. [Online]. Available: http://packetstormsecurity.org/files/24487/stick.htm.html

[34] I. Corona, D. Ariu, and G. Giacinto, "Hmm-web: a framework for the detection of attacks against web applications," in *Proceedings of the 2009 IEEE international conference on Communications*, ser. ICC'09. Piscataway, NJ, USA: IEEE Press, 2009, pp. 747–752.

[35] I. Corona, G. Giacinto, C. Mazzariello, F. Roli, and C. Sansone, "Information fusion for computer security: State of the art and open issues," *Information Fusion*, vol. 10, no. 4, pp. 274–284, 2009.

[36] S. A. Crosby and D. S. Wallach, "Denial of service via algorithmic complexity attacks," in *SSYM'03: Proceedings of the 12th conference on USENIX Security Symposium*. Berkeley, CA, USA: USENIX Association, 2003, pp. 3–3.

[37] S. Damaye, "Pytbull, ids testing framework," Web, October 2012. [Online]. Available: http://pytbull.sourceforge.net

[38] D. Davidson, R. Smith, N. Doyle, and S. Jha, "Protocol normalization using attribute grammars," in *ESORICS'09: Proceedings of the 14th European conference on Research in computer security*. Berlin, Heidelberg: Springer-Verlag, 2009, pp. 216–231.

[39] H. Debar, M. Dacier, and A. Wespi, "Towards a taxonomy of intrusion-detection systems," *Computer Networks*, vol. 31, no. 8, pp. 805–822, 1999.

[40] H. Debar and A. Wespi, "Aggregation and correlation of intrusion-detection alerts," in *Recent Advances in Intrusion Detection*, ser. Lecture Notes in Computer Science, W. Lee, L. M, and A. Wespi, Eds., vol. 2212. Springer, 2001, pp. 85–103.

[41] D. Denning, "An intrusion detection model," *IEEE Transactions on Software Engineering*, vol. SE-13, no. 2, pp. 222–232, feb. 1987.

[42] B. Dolan-Gavitt, T. Leek, M. Zhivich, J. Giffin, and W. Lee, "Virtuoso: Narrowing the semantic gap in virtual machine introspection," in *IEEE Symposium on Security and Privacy*, ser. SP '11. Washington, DC, USA: IEEE Computer Society, 2011, pp. 297–312.

[43] H. Dreger, A. Feldmann, M. Mai, V. Paxson, and R. Sommer, "Dynamic application-layer protocol analysis for network intrusion detection," in *USENIX-SS'06: Proceedings of the 15th conference on USENIX Security Symposium*. Berkeley, CA, USA: USENIX Association, 2006.

[44] P. V. Dreger H., Kreibich C. and R. Sommer, "Enhancing the accuracy of network-based intrusion detection with host-based context," in *Proc. Conference on Detection of Intrusions and Malware and Vulnerability Assessment (DIMVA)*, 2005.

[45] W. M. Eddy, "Defenses against tcp syn flooding attacks," *The Internet Protocol Journal*, vol. 9, pp. 2–17, 2006. [Online]. Available: http://www.cisco.com/web/about/ac123/ac147/archived_issues/ipj_9-4/syn_flooding_attacks.html

[46] V. Felmetsger, L. Cavedon, C. Kruegel, and G. Vigna, "Toward automated detection of logic vulnerabilities in web applications," in *Proceedings of the USENIX Security Symposium*, Washington, DC, August 2010.

[47] D. Fisch, A. Hofmann, and B. Sick, "On the versatility of radial basis function neural networks: A case study in the field of intrusion detection," *Information Sciences*, vol. 180, no. 12, pp. 2421–2439, 2010.

[48] U. Flegel, *Privacy-Respecting Intrusion Detection*, ser. Advances in Information Security. Springer Science Business Media, LLC, 2007, vol. 35.

[49] P. Fogla, M. Sharif, R. Perdisci, O. Kolesnikov, and W. Lee, "Polymorphic blending attacks," in *USENIX Security Symposium*, 2006, pp. 241–256.

[50] J. Forristal, "Libwhisker," Web, March 2013. [Online]. Available: http://swik.net/LibWhisker

[51] J. Franklin, A. Perrig, V. Paxson, and S. Savage, "An inquiry into the nature and causes of the wealth of internet miscreants," in *ACM Conference on Computer and Communications Security*, P. Ning, S. D. C. di Vimercati, and P. F. Syverson, Eds. ACM, 2007, pp. 375–388.

[52] Y. Fu and Z. Lin, "Space traveling across vm: Automatically bridging the semantic gap in virtual machine introspection via online kernel data redirection," in *IEEE Symposium on Security and Privacy*, 2012, pp. 586–600.

[53] F. Gadaleta, Y. Younan, B. Jacobs, W. Joosen, E. De Neve, and N. Beosier, "Instruction-level countermeasures against stack-based buffer overflow attacks," in *VDTS '09: Proceedings of the 1st EuroSys Workshop on Virtualization Technology for Dependable Systems*. New York, NY, USA: ACM, 2009, pp. 7–12.

[54] T. Garfinkel and M. Rosenblum, "A virtual machine introspection-based architecture for intrusion detection," in *Proc. 10th Symp. Network and Distributed System Security (NDSS 03)*, I. Society, Ed., 2003, pp. 191–206.

[55] I. Green, T. Raz, and M. Zviran, "Analysis of active intrusion prevention data for predicting hostile activity in computer networks," *Communications of the ACM*, vol. 50, no. 4, pp. 63–68, 2007.

[56] M. V. Gundy, D. Balzarotti, and G. Vigna, "Catch me, if you can: evading network signatures with web-based polymorphic worms," in *Proceedings of the first USENIX workshop on Offensive Technologies*, ser. WOOT '07. Berkeley, CA, USA: USENIX Association, 2007, pp. 7:1–7:9.

[57] M. V. Gundy, H. Chen, Z. Su, and G. Vigna, "Feature omission vulnerabilities: Thwarting signature generation for polymorphic worms," in *Annual Computer Security Applications Conference (ACSAC), December 10-14, 2007, Miami Beach, Florida, USA*. IEEE Computer Society, 2007, pp. 74–85.

[58] M. Handley, V. Paxson, and C. Kreibich, "Network intrusion detection: evasion, traffic normalization, and end-to-end protocol semantics," in *SSYM'01: Proceedings of the 10th conference on USENIX Security Symposium*. Berkeley, CA, USA: USENIX Association, 2001, pp. 9–9.

[59] B. Hernacki, J. Bennett, and J. Hoagland, "An overview of network evasion methods," *Information Security Technical Report*, vol. 10, no. 3, pp. 140–149, 2005.

[60] A. Hess, M. Jung, and G. Schäfer, "Combining multiple intrusion detection and response technologies in an active networking based architecture," in *DFN-Arbeitstagung über Kommunikationsnetze*, 2003, pp. 153–165.

[61] G. Hoglund and J. Butler, *Rootkits-Subverting the windows kernel*, Addison-Wesley, Ed. Addison-Wesley Professional, ISBN 978-0321294319, 2006.

[62] T. Holz and F. Raynal, "Detecting honeypots and other suspicious environments," in *Workshop on Information Assurance and Security*, West Point, NY, June 2005, pp. 29–36.

[63] HP-TippingPoint, "Intrusion prevention systems," Web, March 2013, data Sheet. [Online]. Available: http://h17007.www1.hp.com/ca/en/whatsnew/040511-1.aspx

[64] IBM, "Changing the intrusion prevention response on my proventia m integrated security appliance," Web, March 2013. [Online]. Available: http://www-935.ibm.com/services/jp/iss/pdf/document/proventia/proventia_mseries_userguide_2.3.pdf

[65] ——, "Proventia network intrusion prevention system," Web, March 2013, data Sheet. [Online]. Available: http://www-935.ibm.com/services/us/iss/pdf/proventia-network-intrusion-prevention-system-ss.pdf

[66] IETF, "Rfc 2068: Hypertext transfer protocol – http/1.1," Web, January 1997. [Online]. Available: http://www.ietf.org/rfc/rfc2068.txt

[67] ——, "Rfc 4251: The secure shell (ssh) protocol architecture," Web, January 2006. [Online]. Available: http://www.ietf.org/rfc/rfc4251.txt

[68] ——, "Rfc 5246: The transport layer security (tls) protocol, version 1.2," Web, August 2008. [Online]. Available: http://www.ietf.org/rfc/rfc5246.txt

[69] S. Innes and C. Valli, "Honeypots: How do you know when you are inside one?" in *Proceedings of the 4th Australian Digital Forensics Conference*, Edith Cowan University, Perth Western Australia, December 4th 2006.

[70] J. Ioannidis and S. M. Bellovin, "Implementing pushback: Router-based defense against DDoS attacks," in *Proc. Internet Society Symposium on Network and Distributed System Security*, San Diego, California, USA, 2002.

[71] M. Jahnke, J. Tolle, S. Lettgen, M. Bussmann, and U. Weddige, "A robust snmp based infrastructure for intrusion detection and response in tactical manets," in *Detection of Intrusions and Malware &amp; Vulnerability Assessment*, ser. Lecture Notes in Computer Science,

R. Bschkes and P. Laskov, Eds. Springer Berlin / Heidelberg, 2006, vol. 4064, pp. 164–180.

[72] S. Jana and V. Shmatikov, "Abusing file processing in malware detectors for fun and profit," in *IEEE Symposium on Security and Privacy*, San Francisco, California, USA, 21-23 May 2012, pp. 80–94.

[73] L. Juan, C. Kreibich, C.-H. Lin, and V. Paxson, "A tool for offline and live testing of evasion resilience in network intrusion detection systems," in *DIMVA '08: Proceedings of the 5th international conference on Detection of Intrusions and Malware, and Vulnerability Assessment*. Berlin, Heidelberg: Springer-Verlag, 2008, pp. 267–278.

[74] I. Kantzavelou and S. Katsikas, "Playing games with internal attackers repeatedly," in *Systems, Signals and Image Processing, 2009. IWSSIP 2009. 16th International Conference on*, Chalkida, Greece, 2009, pp. 1–6.

[75] H. Kayacik and A. Zincir-Heywood, "Mimicry attacks demystified: What can attackers do to evade detection?" in *Sixth Annual Conference on Privacy, Security and Trust, 2008*. Fredericton, New Brunswick, Canada: IEEE, October 1-3 2008, pp. 213–223.

[76] H. Kayacik, A. Zincir-Heywood, and M. Heywood, "Automatically evading ids using gp authored attacks," in *IEEE Symposium on Computational Intelligence in Security and Defense Applications, 2007*. IEEE, 2007, pp. 153–160.

[77] H. G. Kayacik, M. Heywood, and N. Zincir-Heywood, "On evolving buffer overflow attacks using genetic programming," in *GECCO '06: Proceedings of the 8th annual conference on Genetic and evolutionary computation*. New York, NY, USA: ACM, 2006, pp. 1667–1674.

[78] M. Kearns and M. Li, "Learning in the presence of malicious errors," *SIAM Journal on Computing (SICOMP)*, vol. 22, no. 4, pp. 807–837, 1993.

[79] M. Keil, "Encrypted tunnels enable users to circumvent security controls, palo alto networks," Web, June 2009, security Post. [Online]. Available: http://threatpost.com/en_us/blogs/encrypted-tunnels-enable-users-circumvent-security-controls-060109

[80] J. Kim, K. Kim, and J. Jang, "Policy-based intrusion detection and automated response mechanism," in *Information Networking: Wireless Communications Technologies and Network Applications*, ser. Lecture Notes in Computer Science, I. Chong, Ed. Springer Berlin / Heidelberg, 2002, vol. 2344, pp. 399–408, 10.1007/3-540-45801-8_39.

[81] M. Kloft and P. Laskov, "Online anomaly detection under adversarial impact," in *JMLR Workshop and Conference Proceedings, Volume 9: AISTATS*, Y. W. Teh and M. Titterington, Eds. MIT Press, 2010, pp. 405–412.

[82] J. C. Knight, "Safety critical systems: challenges and directions," in *Proceedings of the 24th International Conference on Software Engineering*, ser. ICSE '02. New York, NY, USA: ACM, 2002, pp. 547–550.

[83] C. Kolbitsch, P. M. Comparetti, C. Kruegel, E. Kirda, X. Zhou, and X. Wang, "Effective and efficient malware detection at the end host," in *SSYM'09: Proceedings of the 18th conference on USENIX security symposium*. Berkeley, CA, USA: USENIX Association, 2009, pp. 351–366.

[84] C. Kolbitsch, T. Holz, C. Kruegel, and E. Kirda, "Inspector gadget: Automated extraction of proprietary gadgets from malware binaries," in *IEEE Symposium on Security and Privacy*. IEEE Computer Society, 2010, pp. 29–44.

[85] C. Kruegel, W. Robertson, and G. Vigna, "Using alert verification to identify successful intrusion attempts," in *Practice in Information Processing and Communication (PIK 2004)*, vol. 27, no. 4, October 2004, pp. 219–227.

[86] C. Kruegel, F. Valeur, and G. Vigna, *Intrusion Detection and Correlation: Challenges and Solutions*, Springer, Ed. Springer-Verlag, ISBN: 978-0387233987, 2005, vol. 14.

[87] C. Kruegel, D. Balzarotti, W. Robertson, and G. Vigna, "Improving signature testing through dynamic data flow analysis." in *Annual Computer Security Applications Conference (ACSAC)*. Miami Beach, Florida, USA: IEEE Computer Society, December 10-14 2007, pp. 53–63.

[88] C. Kruegel, G. Vigna, and W. Robertson, "A multi-model approach to the detection of web-based attacks," *Computer Networks*, vol. 48, no. 5, pp. 717–738, 2005.

[89] A. Lanzi, D. Balzarotti, C. Kruegel, M. Christodorescu, and E. Kirda, "Accessminer: Using system-centric models for malware protection," in *ACM Conference on Computer and Communications Security (CCS)*, Chicago, USA, October 2010.

[90] A. Lazarevic and V. Kumar, "Feature bagging for outlier detection," in *Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Chicago, Illinois, USA, August 21-24, 2005*, 2005, pp. 157–166.

[91] A. Lazarevic, V. Kumar, and J. Srivastava, "Intrusion detection: A survey," in *Managing Cyber Threats*, ser. Massive Computing, V. Kumar, J. Srivastava, and A. Lazarevic, Eds. Springer US, 2005, vol. 5, pp. 19–78.

[92] W. Lee, "Applying data mining to intrusion detection: The quest for automation, efficiency, and credibility," *SIGKDD Explorations*, vol. 4, no. 2, pp. 35–42, 2002.

[93] W. Lee, W. Fan, M. Miller, S. J. Stolfo, and E. Zadok, "Toward cost-sensitive modeling for intrusion detection and response," *Journal of Computer Security*, vol. 10, pp. 5–22, July 2002.

[94] W. Lee and D. Xiang, "Information-theoretic measures for anomaly detection," in *IEEE Symposium on Security and Privacy*, Oakland, California, USA, May 14-16 2001, pp. 130–143.

[95] Z. Li, M. Sanghi, Y. Chen, M. Kao, and B. Chavez, "Hamsa: Fast signature generation for zero-day polymorphic worms with provable attack resilience," in *IEEE Symposium on Security and Privacy*. Washington, DC, USA: IEEE Computer Society, May 2006, p. 3247.

[96] D. Maiorca, G. Giacinto, and I. Corona, "A pattern recognition system for malicious pdf files detection," in *MLDM - International Conference on Machine Learning and Data Mining*, P. Perner, Ed., vol. 7376, Springer. Berlin: Springer, 16/07/2012 2012, pp. 510–524.

[97] M. H. Manshaei, Q. Zhu, T. Alpcan, T. Basar, and J.-P. Hubaux, "Game theory meets network security and privacy," Ecole Polytechnique Federale de Lausanne, Switzerland, Tech. Rep. 151965, September 2010.

[98] F. Massicotte, F. Gagnon, Y. Labiche, L. C. Briand, and M. Couture, "Automatic evaluation of intrusion detection systems," in *Annual Computer Security Applications Conference (ACSAC)*. Miami Beach, Florida, USA: IEEE Computer Society, 11-15 December 2006, pp. 361–370.

[99] McAfeeLabs, "Threats report second quarter," Web, March 2013, security Report. [Online]. Available: http://www.mcafee.com/us/local_content/reports/q22010_threats_report_en.pdf

[100] D. Mutz, C. Kruegel, W. Robertson, G. Vigna, and R. Kemmerer, "Reverse engineering of network signatures," in *Proceedings of the AusCERT Asia Pacific Information Technology Security Conference (Gold Coast, Australia), University of Queensland*, 2005.

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

[102] B. Nelson, M. Barreno, F. J. Chi, A. D. Joseph, B. I. P. Rubinstein, U. Saini, C. Sutton, J. D. Tygar, and K. Xia, *Misleading learners: Co-opting your spam filter*. Springer, 2009, ch. Machine Learning in Cyber Trust: Security, Privacy, and Reliability, pp. 17–51.

[103] ——, "Exploiting machine learning to subvert your spam filter," in *USENIX Workshop on Large-Scale Exploits and Emergent Threats, Proceedings*, F. Monrose, Ed. San Francisco, CA, USA: USENIX Association, April 15 2008.

[104] J. Newsome, B. Karp, and D. X. Song, "Polygraph: Automatically generating signatures for polymorphic worms," in *IEEE Symposium on Security and Privacy*. Oakland, CA, USA: IEEE Computer Society, 8-11 May 2005, pp. 226–241.

[105] ——, "Paragraph: Thwarting signature learning by training maliciously," in *RAID*, ser. Lecture Notes in Computer Science, D. Zamboni and C. Krugel, Eds., vol. 4219. Springer, 2006, pp. 81–105.

[106] Z. Ning and J. Gong, "A sampling method for intrusion detection system," in *Challenges for Next Generation Network Operations and Service Management*, ser. Lecture Notes in Computer Science, Y. Ma, D. Choi, and S. Ata, Eds. Springer Berlin Heidelberg, 2008, vol. 5297, pp. 419–428.

[107] J. Novak, "Target-based fragmentation reassembly," Sourcefire, Incorporated, 9770 Patuxent Woods Drive, Columbia, MD 21046, Tech. Rep., 2005. [Online]. Available: http://www.cs.luc.edu/~pld/courses/447/sum08/class3/novak.target_based_frag.pdf

[108] F. Oliviero, L. Peluso, and S. Romano, "Refacing: An autonomic approach to network security based on multidimensional trustworthiness," *Computer Networks*, vol. 52, no. 14, pp. 2745–2763, 2008.

[109] G. Ollmann, "Serial variant evasion tactics: Techniques used to automatically bypass antivirus technologies," Web, January 2009, security Whitepaper. [Online]. Available: http://www.damballa.com/downloads/r_pubs/WP_SerialVariantEvasionTactics.pdf

[110] A. Papadogiannakis, M. Polychronakis, and E. P. Markatos, "Improving the accuracy of network intrusion detection systems under load using selective packet discarding," in *EUROSEC '10: Proceedings of the*

Third European Workshop on System Security. New York, NY, USA: ACM, 2010, pp. 15–21.

[111] J. Patel, A. Liu, and E. Torng, "Bypassing space explosion in regular expression matching for network intrusion detection and prevention systems," in *Proceedings of the 19th Annual Network & Distributed System Security Symposium (NDSS)*, San Diego, California, February 2012.

[112] S. Patton, W. Yurcik, and D. Doss, "An achilles' heel in signature-based ids: Squealing false positives in snort," in *Proceedings of fourth International Symposium on Recent Advances in Intrusion Detection*, vol. 10, october 2001, p. 12.

[113] V. Paxson, "Bro: a system for detecting network intruders in real-time," *Computer Networks*, vol. 31, pp. 2435–2463, 1999.

[114] V. Paxson and M. Handley, "Defending against network ids evasion," in *Recent Advances in Intrusion Detection*, West Lafayette, Indiana, USA, September 7-9 1999.

[115] R. Perdisci, I. Corona, D. Dagon, and W. Lee, "Detecting malicious flux service networks through passive analysis of recursive dns traces," in *Twenty-Fifth Annual Computer Security Applications Conference (ACSAC)*, Honolulu, Hawaii, USA, 7-11 December 2009.

[116] R. Perdisci, D. Dagon, W. Lee, P. Fogla, and M. Sharif, "Misleading worm signature generators using deliberate noise injection," in *IEEE Symposium on Security and Privacy*. Washington, DC, USA: IEEE Computer Society, 2006.

[117] R. Perdisci, G. Gu, and W. Lee, "Using an ensemble of one-class svm classifiers to harden payload-based anomaly detection systems," in *Proceedings of the 6th IEEE International Conference on Data Mining (ICDM)*. Hong Kong, China: IEEE Computer Society, 18-22 December 2006, pp. 488–498.

[118] R. Perdisci, W. Lee, and N. Feamster, "Behavioral clustering of http-based malware and signature generation using malicious network traces," in *Proceedings of the 7th USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, San Jose, CA, USA, April 28-30 2010, pp. 391–404.

[119] M. Pinyathinun and C. Sathitwiriyawong, "Dynamic policy model for target based intrusion detection system," in *ICIS '09: Proceedings of the 2nd International Conference on Interaction Sciences*. New York, NY, USA: ACM, 2009, pp. 930–934.

[120] M. Polychronakis, K. Anagnostakis, and E. Markatos, "Comprehensive shellcode detection using runtime heuristics," in *Twenty-Sixth Annual Computer Security Applications Conference (ACSAC)*, Austin, Texas, USA, 6-10 December 2010.

[121] M. Polychronakis, K. G. Anagnostakis, and E. P. Markatos, "Network-level polymorphic shellcode detection using emulation," *Journal in Computer Virology*, vol. 2, no. 4, pp. 257–274, 2007.

[122] P. A. Porras, "Directions in network-based security monitoring," *IEEE Security & Privacy*, vol. 7, no. 1, pp. 82–85, 2009.

[123] T. Ptacek and T. Newsham, "Insertion, evasion, and denial of service: evading network intrusion detection," Secure Networks Inc., Tech. Rep., 1998. [Online]. Available: http://insecure.org/stf/secnet_ids/secnet_ids.html

[124] R. Riley, X. Jiang, and D. Xu, "Multi-aspect profiling of kernel rootkit behavior," in *EuroSys '09: Proceedings of the 4th ACM European conference on Computer systems*. New York, NY, USA: ACM, 2009, pp. 47–60.

[125] W. K. Robertson, G. Vigna, C. Krügel, and R. A. Kemmerer, "Using generalization and characterization techniques in the anomaly-based detection of web attacks," in *In Proceedings of the 13 th Symposium on Network and Distributed System Security*, San Diego, California, USA, 2006.

[126] M. Roesch, "Snort - lightweight intrusion detection for networks," in *LISA '99: Proceedings of the 13th USENIX conference on System administration*. Berkeley, CA, USA: USENIX Association, 1999, pp. 229–238. [Online]. Available: http://www.snort.org

[127] M. Rolando, M. Rossi, N. Sanarico, and D. Mandrioli, "A formal approach to sensor placement and configuration in a network intrusion detection system," in *SESS '06: Proceedings of the 2006 international workshop on Software engineering for secure systems*. New York, NY, USA: ACM, 2006, pp. 65–71.

[128] S. Roy, C. Ellis, S. Shiva, D. Dasgupta, V. Shandilya, and Q. Wu, "A survey of game theory as applied to network security," in *Hawaii International Conference on System Sciences*. Los Alamitos, CA, USA: IEEE Computer Society, 2010, pp. 1–10.

[129] S. Rubin, S. Jha, and B. Miller, "On the completeness of attack mutation algorithms," in *Computer Security Foundations Workshop, 2006. 19th IEEE*, Venice, Italy, 5-7 July 2006, pp. 14–56.

[130] S. Rubin, S. Jha, and B. P. Miller, "Automatic generation and analysis of nids attacks," in *ACSAC '04: Proceedings of the 20th Annual Computer Security Applications Conference*. Washington, DC, USA: IEEE Computer Society, 2004, pp. 28–38.

[131] B. I. Rubinstein, B. Nelson, L. Huang, A. D. Joseph, S.-h. Lau, S. Rao, N. Taft, and J. D. Tygar, "Antidote: understanding and defending against poisoning of anomaly detectors," in *IMC '09: Proceedings of the 9th ACM SIGCOMM conference on Internet measurement conference*. New York, NY, USA: ACM, 2009, pp. 1–14.

[132] D. Schnackenberg, K. Djahandari, and D. Sterne, "Infrastructure for intrusion detection and response," *DARPA Information Survivability Conference and Exposition*, vol. 2, p. 1003, 2000.

[133] B. Schneier, *Beyond Fear: Thinking Sensibly about Security in an Uncertain World*, Copernicus, Ed. New York, NY: Copernicus, ISBN: 978-0387026206, September 2003.

[134] S. Segui, L. Igual, and J. Vitria, "Weighted bagging for graph based one-class classifiers," in *Proceedings of the 9th International Workshop on Multiple Classifier Systems*, ser. Lecture Notes in Computer Science, vol. 5997. Springer-Verlag, 2010, pp. 1–10.

[135] R. Servedio, "Smooth boosting and learning with malicious noise," *The Journal of Machine Learning Research*, vol. 4, pp. 633–648, 2003.

[136] U. Shankar and V. Paxson, "Active mapping: Resisting nids evasion without altering traffic," in *IEEE Symposium on Security and Privacy*. Washington, DC, USA: IEEE Computer Society, 2003, p. 44.

[137] M. I. Sharif, W. Lee, W. Cui, and A. Lanzi, "Secure in-vm monitoring using hardware virtualization," in *ACM Conference on Computer and Communications Security*, Chicago, Illinois, USA, 2009, pp. 477–487.

[138] T. Shon and J. Moon, "A hybrid machine learning approach to network anomaly detection," *Information Sciences*, vol. 177, no. 18, pp. 3799–3821, 2007.

[139] R. Smith, C. Estan, and S. Jha, "Backtracking algorithmic complexity attacks against a nids," in *ACSAC '06: Proceedings of the 22nd Annual Computer Security Applications Conference*. Washington, DC, USA: IEEE Computer Society, 2006, pp. 89–98. [Online]. Available: http://www.acsac.org/2006/papers/54.pdf

[140] sniph00, "Snot, snort alert generator," Web, October 2012. [Online]. Available: ftp://ftp.st.ryukoku.ac.jp/pub/security/tool/snot/

[141] R. Sommer and V. Paxson, "Enhancing byte-level network intrusion detection signatures with context," in *CCS '03: Proceedings of the 10th ACM conference on Computer and communications security*. New York, NY, USA: ACM, 2003, pp. 262–271.

[142] ——, "Outside the closed world: On using machine learning for network intrusion detection," in *IEEE Symposium on Security and Privacy*, Berleley/Oakland, California, USA, 16-19 May 2010 2010, pp. 305–316.

[143] D. Song, "Fragroute," Web, March 2013, attack Tool. [Online]. Available: http://monkey.org/~dugsong/fragroute/

[144] J. Song, H. Takakura, Y. Okabe, and K. Nakao, "Toward a more practical unsupervised anomaly detection system," *Information Sciences*, vol. 231, pp. 4–14, 2013, data Mining for Information Security.

[145] SophosLabs, "Security threat report," Web, March 2013. [Online]. Available: http://www.sophos.com/security/technical-papers/modern_web_attacks.pdf

[146] Sourcefire, "Intrusion prevention system," Web, March 2013. [Online]. Available: http://www.sourcefire.com/solutions/etm/ips

[147] A. Srivastava and J. Giffin, "Efficient protection of kernel data structures via object partitioning," in *Annual Computer Security Applications Conference (ACSAC)*, Orlando, FL, USA, 3-7 December 2012.

[148] N. Stakhanova, S. Basu, and J. Wong, "Taxonomy of intrusion response systems," *International Journal of Information and Computer Security*, vol. 1, pp. 169–184, January 2007.

[149] S. Staniford-Chen, B. Tung, and D. Schnackenberg, "The common intrusion detection framework (cidf)," in *Information Survivability Workshop*, Orlando, Florida, USA, 1998. [Online]. Available: http://gost.isi.edu/cidf/

[150] Stonesoft, "Protection against advanced evasion techniques in stonesoft ips," Whitepaper, Tech. Rep., 2012. [Online]. Available: http://evader.stonesoft.com/assets/files/AET_Whitepaper2012.pdf

[151] ——, "Evader, network-based ids testing environment," Web, March 2013, evasion Tool. [Online]. Available: http://evader.stonesoft.com

[152] C. Strasburg, N. Stakhanova, S. Basu, and J. Wong, "A framework for cost sensitive assessment of intrusion response selection," in *Computer Software and Applications Conference, 2009. COMPSAC '09. 33rd Annual IEEE International*, vol. 1, 2009, pp. 355–360.

[153] I. Svecs, T. Sarkar, S. Basu, and J. S. Wong, "Xidr: A dynamic framework utilizing cross-layer intrusion detection for effective re-

sponse deployment," in *IEEE 34th Annual Computer Software and Applications Conference Workshops*, 2010, pp. 287–292.

[154] S. Templeton and K. Levitt, "Detecting spoofed packets," in *DARPA Information Survivability Conference and Exposition, 2003. Proceedings*, vol. 1, 2003, pp. 164–175.

[155] O. Thonnard, L. Bilge, G. O'Gorman, S. Kiernan, and M. Lee, "Industrial espionage and targeted attacks: Understanding the characteristics of an escalating threat," in *Research in Attacks, Intrusions, and Defenses - 15th International Symposium, RAID 2012*, Amsterdam, The Netherlands, September 12-14 2012, pp. 64–85.

[156] E. Tsyrklevich, "Attacking host intrusion prevention systems," in *Black Hat USA*, 2004. [Online]. Available: http://www.blackhat.com/presentations/bh-usa-04/bh-us-04-tsyrklevich.pdf

[157] L. G. Valiant, "A theory of the learnable," *Communications of the ACM*, vol. 27, no. 11, pp. 1134–1142, Nov. 1984.

[158] S. Venkataraman, A. Blum, and D. Song, "Limits of learning-based signature generation with adversaries," in *Proceedings of the Network and Distributed System Security Symposium, NDSS 2008, San Diego, California, USA, 10th February - 13th February 2008*, 2008.

[159] G. Vigna, W. Robertson, and D. Balzarotti, "Testing network-based intrusion detection signatures using mutant exploits," in *CCS '04: Proceedings of the 11th ACM conference on Computer and communications security*. New York, NY, USA: ACM, 2004, pp. 21–30.

[160] N. H. Vu, H. H. Ang, and V. Gopalkrishnan, "Mining outliers with ensemble of heterogeneous detectors on random subspaces," in *Database Systems for Advanced Applications, 15th International Conference, DASFAA 2010, Tsukuba, Japan, April 1-4, 2010, Proceedings, Part I*, 2010, pp. 368–383.

[161] H. Wang, G. Wang, Y. Lan, K. Wang, and D. Liu, "A new automatic intrusion response taxonomy and its application," in *Advanced Web and Network Technologies, and Applications*, ser. Lecture Notes in Computer Science, H. Shen, J. Li, M. Li, J. Ni, and W. Wang, Eds. Springer Berlin / Heidelberg, 2006, vol. 3842, pp. 999–1003, 10.1007/11610496_139.

[162] K. Wang, G. F. Cretu, and S. J. Stolfo, "Anomalous payload-based worm detection and signature generation," in *Recent Advances in Intrusion Detection, 8th International Symposium, RAID*, Seattle, WA, USA, September 7-9 2005, pp. 227–246.

[163] L. Wang, Z. Li, Y. Chen, Z. Fu, and X. Li, "Thwarting zero-day polymorphic worms with network-level length-based signature generation," *IEEE/ACM Transactions on Networking*, vol. 18, no. 1, pp. 53–66, 2010.

[164] S. L. Wang, K. Shafi, C. Lokan, and H. A. Abbass, "Adversarial learning: the impact of statistical sample selection techniques on neural ensembles," *Evolving Systems*, vol. 1, no. 3, pp. 181–197, 2010.

[165] Z. Wang and X. Jiang, "Hypersafe: A lightweight approach to provide lifetime hypervisor control-flow integrity," in *IEEE Symposium on Security and Privacy*. Washington, DC, USA: IEEE Computer Society, 2010, pp. 380–395.

[166] D. Watson, M. Smart, G. R. Malan, and F. Jahanian, "Protocol scrubbing: network security through transparent flow modification," *IEEE/ACM Transactions on Networking (TON)*, vol. 12, no. 2, pp. 261–273, 2004.

[167] G. Wurster and P. C. van Oorschot, "A control point for reducing root abuse of file-system privileges," in *CCS '10: Proceedings of the 17th ACM conference on Computer and communications security*. New York, NY, USA: ACM, 2010, pp. 224–236.

[168] D. Yu and D. Frincke, "Alert confidence fusion in intrusion detection systems with extended dempster-shafer theory," in *Proceedings of the 43rd annual Southeast regional conference - Volume 2*, ser. ACM-SE 43. New York, NY, USA: ACM, 2005, pp. 142–147.

[169] J. Yu, H. Lee, M.-S. Kim, and D. Park, "Traffic flooding attack detection with snmp mib using svm," *Computer Communications*, vol. 31, pp. 4212–4219, November 2008.

[170] W. Yurcik, "Controlling intrusion detection systems by generating false positives: Squealing proof-of-concept," in *Proceedings of the 27th Annual IEEE Conference on Local Computer Networks (LCN)*. Tampa, Florida, USA: IEEE Computer Society, 6-8 November 2002, pp. 134–135.

[171] K. Zhao, M. Zhang, K. Yang, and L. Hu, "Data collection for intrusion detection system based on stratified random sampling," in *IEEE International Conference on Networking, Sensing and Control*, april 2007, pp. 852–855.

[172] S. Zonouz, H. Khurana, W. Sanders, and T. Yardley, "Rre: A game-theoretic intrusion response and recovery engine," in *Dependable Systems Networks, 2009. DSN '09. IEEE/IFIP International Conference on*, 292009-july2 2009, pp. 439–448.