Modeling Realistic Adversarial Attacks against Network Intrusion Detection Systems

GIOVANNI APRUZZESE, Institute of Information Systems, University of Liechtenstein
MAURO ANDREOLINI and LUCA FERRETTI, Department of Physics, Informatics and Mathematics, University of Modena and Reggio Emilia
MIRCO MARCHETTI, Department of Engineering "Enzo Ferrari," University of Modena and Reggio Emilia
MICHELE COLAJANNI, Department of Informatics, Science and Engineering, University of Bologna

The incremental diffusion of machine learning algorithms in supporting cybersecurity is creating novel defensive opportunities but also new types of risks. Multiple researches have shown that machine learning methods are vulnerable to adversarial attacks that create tiny perturbations aimed at decreasing the effectiveness of detecting threats. We observe that existing literature assumes threat models that are inappropriate for realistic cybersecurity scenarios, because they consider opponents with complete knowledge about the cyber detector or that can freely interact with the target systems. By focusing on Network Intrusion Detection Systems based on machine learning, we identify and model the real capabilities and circumstances required by attackers to carry out feasible and successful adversarial attacks. We then apply our model to several adversarial attacks proposed in literature and highlight the limits and merits that can result in actual adversarial attacks. The contributions of this article can help hardening defensive systems by letting cyber defenders address the most critical and real issues and can benefit researchers by allowing them to devise novel forms of adversarial attacks based on realistic threat models.

CCS Concepts: • **Security and privacy → Network security**; • **Computing methodologies → Machine learning;**

Additional Key Words and Phrases: Cybersecurity, network intrusion detection, adversarial attacks, evasion, NIDS

ACM Reference format:
Giovanni Apruzzese, Mauro Andreolini, Luca Ferretti, Mirco Marchetti, and Michele Colajanni. 2022. Modeling Realistic Adversarial Attacks against Network Intrusion Detection Systems. *Digit. Threat. Res. Pract.* 3, 3, Article 31 (September 2022), 19 pages.
https://doi.org/10.1145/3469659

1 INTRODUCTION

Machine Learning (ML) is increasingly adopted in multiple domains. As a typical consequence of this world, it is becoming a preferred target of modern adversarial attacks [70, 92]. This emerging digital threat involves attackers that exploit the intrinsic vulnerabilities of machine learning methods by leveraging specific inputs

Authors' addresses: G. Apruzzese, Institute of Information Systems, University of Liechtenstein, Vaduz, Liechtenstein; email: giovanni.apruzzese@uni.li; M. Andreolini and L. Ferretti, Department of Physics, Informatics and Mathematics, University of Modena and Reggio Emilia, Modena, Italy; emails: {mauro.andreolini, luca.ferretti}@unimore.it; M. Marchetti, Department of Engineering "Enzo Ferrari", University of Modena and Reggio Emilia, Modena, Italy; email: mirco.marchetti@unimore.it; M. Colajanni, Department of Informatics, Science and Engineering, University of Bologna, Bologna, Italy; email: michele.colajanni@unimore.it.

2576-5337/2022/09-ART31 $15.00
https://doi.org/10.1145/3469659

that thwart their predictions. The state of the art focuses on computer vision and natural language processing applications [59]. However, several studies on cybersecurity are appearing [63], especially because this scenario is inherently affected by opponents that are rarer in other considered contexts. In cybersecurity, the main interest has been on malware and spam detectors, while fewer works consider **Network Intrusion Detection Systems (NIDS)** that are of interest for this article [13, 27, 76].

The motivations for our research starts from the observation that most papers proposing, evaluating, and considering NIDS in adversarial scenarios do not discuss the realistic level of the proposed attacks. A typical research work may assume any threat model and then proceed to analyze the effects of the attack with no or insufficient considerations about the feasibility of the considered scenario [39]. For example, some papers assume attackers that know everything about the target system [35]. Others suppose that an adversary can perform an arbitrarily large number of trials against the NIDS without getting noticed [72]. Although investigating the effectiveness of adversarial attacks against any ML system is an important goal for creating more robust detectors, cybersecurity scenarios should always deal with realistic issues and adversaries. Otherwise, defenders may spend resources against false hits or unrealistic problems, when more critical issues are taking place. The sheer amount of researches on adversarial attacks may even induce cyber defenders to think that any **Machine Learning-based NIDS (ML-NIDS)** is an unreliable defensive system, although this is not the case.

This article analyzes the realistic feasibility of adversarial attacks against ML-NIDS by identifying the capabilities and conditions that are necessary for carrying out such attacks against ML-NIDS. We identify five elements of the target system that can be leveraged to perform adversarial attacks. By introducing the concept of *power*, which models the attacker's knowledge and capabilities on each of these five elements, we outline the realistic circumstances that allow an attacker to thwart the target system. Our proposal complements taxonomies that do not delve into the feasibility and constraints of real ML-NIDS scenarios (e.g., References [37, 54]) and it can be applied to assess and evaluate adversarial attacks against any ML-NIDS.

We apply the proposed concepts to analyze the state of the art on adversarial attacks against ML-NIDS. We assess the feasibility levels of current researches, and analyze the details of four significant use cases. We can conclude that few papers assume scenarios that are representative of an authentic cyber defensive world. Hence, there is a gap between academic and real environments that must be filled for a mutual interest. The most recent efforts are taking into account more interesting and realistic scenarios, which is a positive trend.

Researchers on adversarial ML can benefit of this article by formulating original adversarial attacks that assume realistic threat models. Moreover, security experts working on ML-NIDS can harden their defensive systems by considering the smaller and more realistic subset of feasible attacker's scenarios.

The article is structured as follows. Section 2 presents the basic concepts of this article. Section 3 motivates and compares our article against related work. Section 4 proposes our original analysis for modeling realistic adversarial attacks against ML-NIDS. Section 5 applies the proposed analysis to existing proposals of adversarial attacks, and describes three case studies. Section 6 concludes the article with some final remarks and possible future work.

2 BACKGROUND

We consider adversarial attacks against network intrusion detection systems that rely on machine learning methods for detecting attacks. Let us summarize these concepts.

2.1 Network Intrusion Detection Systems Based on Machine Learning

The detection of malicious events is a prominent issue in the cybersecurity landscape. As manual inspection is impossible when millions of events per day occur, human defenders are supported by **intrusion detection systems (IDS)** that analyze data from different sources and, when specific conditions are met, generate alerts for the triage phase [57]. We consider the specific category of Network-IDS, which aim at identifying intrusions

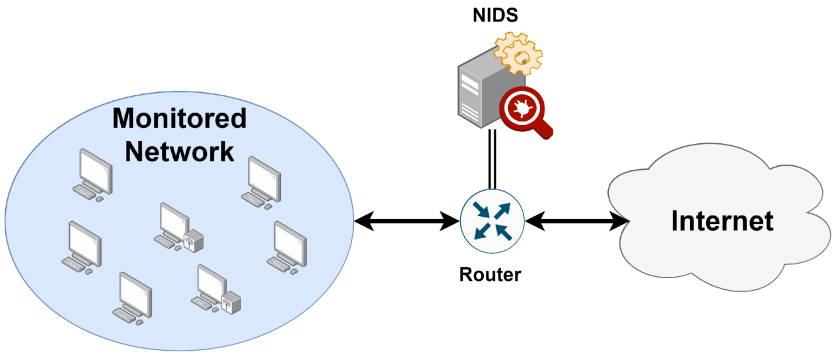

Fig. 1. Typical deployment of a Network Intrusion Detection System (NIDS).

at the network traffic level. Several types of NIDS exists, but common differences involve the data type analyzed, and the method used to perform the detection.

The first generation of NIDS used to analyze network packets by inspecting their payload. While this approach may be more accurate, it cannot be applied when data is encrypted, and requires high amounts of computational resources to process each network packet. The exponential growth of traffic that often is encrypted raised the interest toward NIDS inspecting metadata, such as network flows [57]. In the past decade [90], many NIDS leverage the analysis of network metrics that summarize entire communication sequences between two endpoints, such as the duration of the session or the amount of exchanged bytes. This information is computationally feasible to store and analyze, and does not present privacy concerns.

With regards to the detection method, initial NIDS identified known threats on the basis of human-written *signatures* [28], but recent solutions adopt (also) data-driven methods [60] leveraging anomaly detection typically based on machine learning techniques [31, 88]. These approaches allow a NIDS to detect also unknown malicious events that expert attackers may adopt to evade signature-based methods.

This article focuses on ML-NIDS. Machine Learning generates models that are able to learn specific patterns by providing them with training data. These patterns are then used to make predictions on new and unseen sets of data [70]. The main advantage of these detection schemes is their capability of automatically learning from training data without human intervention, thus simplifying the resource-intensive management procedures required by traditional misuse-based approaches. Furthermore, they are also able of detecting novel attack variants, for which no known signature exists and that would be undetectable by NIDS based exclusively on rules. There exists a wide literature on ML-NIDS [17, 21, 31], which can work either on payload (e.g., Reference [106]) or on netflow data (e.g., Reference [20]). Proposals include supervised and unsupervised techniques, and also an increasing number of recent approaches based on deep neural networks [63]. Although the detection performance depends on the ML algorithm, the superiority of deep learning methods in this domain is still questionable [12].

A typical deployment scenario is represented in Figure 1, where the border router forwards all incoming traffic to the NIDS, which proceeds to analyze it. This system is one of the most protected elements of the entire network. Cybersecurity threat models exclude that an attacker can access the NIDS, because, in a similar instance, the entire defensive infrastructure can be considered compromised: Any cybersecurity measure can be easily bypassed if the attacker acquires direct or physical access to the defensive system [86].

In Figure 2 we report the typical workflow of a ML-NIDS. The network traffic undergoes some pre-processing operations that extract the relevant features from the data, transforming it into samples accepted by the ML model. These samples are then forwarded to the (trained) ML model that will analyze them and determine whether they are legitimate or not.

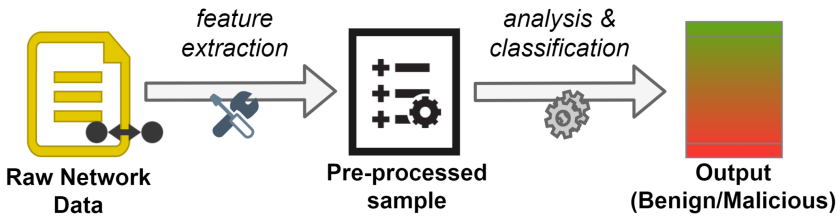

Fig. 2. Workflow of a ML-NIDS.

There exist several solutions of ML-NIDS. An organization may adopt and maintain a ML-NIDS on premises, for example by leveraging open-source software [82]. Alternatively, it may rely on third-party products [17, 23, 85, 86].

In the former case, the lifecycle of the ML model is managed by the organization, which has access to all the ML components, including the model's internal configuration parameters, the feature set, and the dataset used for training purposes. This dataset can be built, for example, by using the regular network traffic generated in the monitored enterprise as "benign" samples, while traces of malicious traffic can be obtained either synthetically or through external security feeds [30]. The dataset can also be periodically updated to better resemble the modifications of the network environment, or to include malicious samples of novel attack variants. The updated version will then be used to re-train the model accordingly [55].

If the ML-NIDS belongs to a third party vendor, then the ML model is trained on datasets that are likely unavailable to the organization. Hence, all the training and management operations will be performed by the third-party vendor. In a similar scenario, the organization has little or no information about the ML-model adopted by the NIDS. It is used as a black-box, where the network traffic generated by the organization is inspected by the model, whose results are then provided to the organization, usually in the form of alerts. An organization may be able to add some rules, for example to denote the most relevant server machines, but the internal configuration and the parameters of the ML model are unknown and not observable [23, 86].

ML-NIDS are gaining popularity [12, 31], with the typical consequence that real attackers are turning their attention to the vulnerabilities of some ML components. This article considers the so called adversarial attacks against ML-NIDS [63, 94], and makes no assumption on the specific ML-algorithm or on the analyzed data type of the ML-NIDS.

2.2 Adversarial Attacks against Machine Learning

Adversarial attacks involve the application of perturbations to some data with the goal of fooling the ML model, thus resulting in an incorrect detection output that favors the attacker. These perturbations should be imperceptible to a human observer. A similar objective is easily verifiable in computer vision, because the modification of few pixels causes unexpected results (e.g., Reference [91]). It is more difficult to reveal perturbations when the manipulation is performed on network traffic data, because each domain has its own characteristics. Attacks that are effective in a scenario may be unsuccessful in others [35, 84].

An adversarial attack can affect the training phase of the machine learning algorithm: In this case, it is denoted as *poisoning* attack, where the aim is to influence the detection through manipulations of the training dataset. Manipulations may involve either the injection of new data samples or the modification of existing samples (e.g., by flipping the labels of some data points [48]).

Alternative attacks may occur at *inference*-time when the model is already operational. The focus is to thwart detection by leveraging the sensitivity of the (trained) model to its learned decision boundaries. The most renown adversarial attacks in cyber detection are denoted as *evasion* attacks, where the goal is misclassifying malicious samples as legitimate [54].

Literature presents three main cases of adversarial attacks, depending on the characteristics of the threat model. The first case involves an opponent that knows everything about the target system. By leveraging such information it is possible to understand the decision boundaries of the ML model and to craft specific samples that thwart the detection mechanism. These attacks are also known as *white-box* attacks [63].

The second type of instances, denoted as *black-box* adversarial attacks, assumes that the attacker has no information about the detection mechanism, but they are able to query the machine learning model by issuing some samples and inspecting the resulting classification. The attacker can exploit this input/output association to acquire information about the model adopted for detection. Such information can be leveraged in two ways.

- The attacker can repeatedly modify the malicious samples until they are misclassified. The goal is to determine the boundaries used by the model to distinguish benign from malicious samples.
- Alternatively, the attacker can create a surrogate model of the detection system, and then foster the *transferability* property of ML to devise adversarial samples that fool the surrogate classifier and, in turn, also the real detector [69].

Literature refers to ML models that are exploited through similar attacking strategies as "oracles" [70].

In the third case, denoted as *gray-box* attack, the adversary is more constrained and has limited knowledge about the classifier [38]. For example, they may only know a subset of the features adopted by the ML model; or they may know which algorithm is being used, but without any information on its learned configuration settings.

Some ML algorithms can be more resilient than others. For instance, gradient-based attacks (e.g., Reference [16]) are more effective against neural networks than against tree-based methods. However, any ML algorithm is intrinsically vulnerable to adversarial attacks [42]. Thus, in this article we do not focus on any specific ML method, and our analyses and conclusions can be applied to any ML-NIDS.

3 RELATED WORK

Three trends motivate this article: the increasing reliance of cyber defenses on machine learning and artificial intelligence tools, the consequential rise of novel threats based on adversarial attacks, and the lack of comprehensive guidelines detailing how to fight this emerging menace. We describe each of these points and compare our article with related researches on adversarial machine learning in cybersecurity.

The adoption of techniques belonging to the machine learning paradigm (and to the broader artificial intelligence concept) is growing. Modern advanced systems require the execution of a large number of tasks that cannot be entirely managed by human personnel [7]. Hence, the promising automation capabilities of artificial intelligence methods are greatly appreciated. As an example, autonomous agents are being developed for military applications [50], as well as for the deployment of 5G services [22], and also for cyber security [12, 102]. The increasing diffusion of machine learning leads to questioning their efficacy and trustworthiness when deployed in critical environments [21, 33, 97], as it has been observed even by the European Commission [25].

Among the issues affecting ML methods, the scientific community is giving importance to the effectiveness of these algorithms when they are subject to adversarial attacks [70]. The amount of research papers on adversarial machine learning have rapidly increased as evidenced by Figure 3 (taken from Reference [1]). In this broad landscape, the majority of studies are on adversarial attacks against image and text classification (e.g., References [5, 59, 105, 107, 109]), and only a small percentage looks at the problem from a cybersecurity perspective, which is the focus of this article. Moreover, most papers consider a multitude of scenarios where the target model suffers significant performance degradation (for example, a proficient model can be fooled with over 90% confidence [63]) that may result in overestimating the actual problem posed by adversarial attacks in real environments. Our work aims to answer the question about which adversarial attacks proposed in literature are a real threat to modern cyber systems.

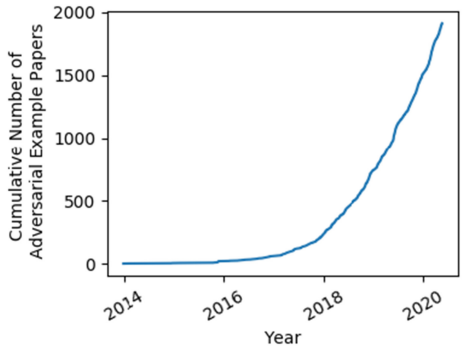

Fig. 3. Growth of adversarial machine learning papers over the years (source: Reference [1]).

Let us compare our article with related researches on adversarial attacks against NIDS. Among the first studies that summarize this threat we mention the seminal work in Reference [26], which did not consider attacks against ML-NIDS. The authors in Reference [31] evidence the problem of botnet detection in adversarial scenarios, but the considered attacks were oriented toward more general applications of machine learning (such as Reference [46]), and they do not discuss any use-cases of adversarial perturbations against NIDS. Two significant surveys on adversarial machine learning were published by Biggio et al. [19] and by Papernot et al. [70]. These papers contain a comprehensive review of the state of the art, although they consider attacks in multiple domains and are do not address the intrinsic issues affecting NIDS scenarios. A similar difference characterizes also some recent reviews [76, 77]. The exhaustive work of Reference [79] proposes several taxonomies for adversarial attacks at different stages, but it does not delve into the characteristics of cyber detection contexts. Recent works addressing this threat from a cybersecurity perspective can be found in References [13, 27, 63, 78], which summarize adversarial ML for cyber detection, but they do not consider the realistic level of the discussed papers. The review in Reference [39] proposes the original concept of adversarial risk that is used to measure the likelihood that an adversarial attack is successful, but this concept is based on the vulnerabilities of a machine learning approach, and does not take into account real world constraints, which is the focus of our contribution. Some practical applications of adversarial attacks not regarding cybersecurity are presented in Reference [69] and in Reference [108], which adopts a game-theoretical perspective. Unlike all these papers, our study provides an original modeling and analysis of adversarial attacks against ML-NIDS that focus on the realistic feasibility of the proposed threats. Our work is oriented to both ML researchers and security specialists that are interested in evaluating their ML-NIDS cyber defence systems in real scenarios that involve constrained attackers.

The recent analysis of Kumar et al. [51] presents the perspective of real enterprises about adversarial attacks and concludes that modern organizations are aware of these problems, but do not consider this threat as a top-priority, because there are no defensive mechanisms that are truly effective in real environments. This conclusion evidences that most literature on adversarial attacks considers unfeasible scenarios. Hence we find useful to outline the main characteristics that must be considered to reproduce realistic adversarial settings, which can be used to devise sensible attacks and to evaluate defensive strategies that are applicable in real contexts.

4 MODELING OF REALISTIC ADVERSARIAL ATTACKS

We analyze adversarial attacks against ML-NIDS by combining and extending the taxonomies of Laskov et al. [54] and Huang et al. [37]. To model the realistic capabilities of an attacker, we introduce the concept of "power" that denotes how much control the attacker has on the target detection system.

We identify five elements on which the attacker has power as follows:

- *Training Data.* It represents the ability to access the dataset used to train the ML-NIDS. It can come in the form of read, write, or no access at all.

- *Feature Set.* It refers to the knowledge of the features analyzed by the ML-NIDS to perform its detection. It can come in the form of none, partial or full knowledge.
- *Detection Model.* It describes the knowledge of the (trained) ML model integrated into the NIDS that is used to perform the detection. This knowledge may be none, partial or full.
- *Oracle.* This element denotes the possibility of obtaining feedback from the output produced by the ML-NIDS to an attacker's input. This feedback can be limited, unlimited or absent.
- *Manipulation Depth.* It describes the nature of the adversarial manipulation, that may modify the analyzed traffic (problem space) traffic level or one or more features (feature space.

We describe each element from the perspective of an attacker that aims at thwarting a ML-NIDS. The conclusions are summarized in Section 4.6.

We assume the typical network intrusion detection scenario, where an attacker has already established a foothold into one (or more) device of the monitored network perimeter, and intends to maintain (and expand) their control and cause damage to the organization, while remaining undetected [14, 31, 63, 101].

4.1 Training Data

Power on the training data comes in two different forms: *read* access, which may allow an attacker to reproduce the training phase and obtain a similar detector to the one adopted by the organization; or *write* access, which can be leveraged to perform poisoning attacks by either injecting new data, or modifying existing samples.

Obtaining power on the training data depends on the ML-NIDS solution adopted by the target organization: The detector can either be developed and managed entirely in-house or can be acquired from a third-party vendor.

In the former case, the attacker may be theoretically able to retrieve the training data by identifying the device hosting the dataset, and then inspecting or exfiltrating the actual data. In practice, a similar machine is well protected through restrictive access policies [23] and/or segregated in a dedicated network segment that is inaccessible by the hosts of the compromised network. In the latter case, the ML component is trained on datasets of the vendor that are not accessible by the attacker unless they also compromise the vendor's network, which is unlikely. However, we stress that if attackers manage to seize control on the training dataset adopted by a third-party provider, then they could potentially launch adversarial attacks against any enterprise that is adopting the specific solution of the (compromised) provider. For this reason, providers of ML-based solutions for cybersecurity must protect their assets as much as possible.

A different circumstance occurs when the ML-NIDS is periodically retrained on new data. In this case, an attacker that has acquired a foothold into the organization may generate some malicious traffic that may affect the detection, thus granting some (indirect) write access to the training data. This attack strategy is unfeasible without direct read access to the training data. Attackers cannot be sure that the produced samples are truly used to retrain the ML-NIDS unless they can inspect the composition of the training dataset.

We can conclude that acquiring any form of reliable access (either *read* or *write*) to the data used to train (or re-train) the ML-model is not very realistic [32, 55, 86]. Some power on the training data is obtainable only if the NIDS and corresponding training set are entirely managed by the target organization.

4.2 Detection Model

With power on the detection model, an attacker can determine the internal configuration of the ML method, alongside the decision boundaries learnt after its training phase. White-box attacks denote scenarios where such knowledge is available to the adversary.

The detection model represents the core component of the ML-NIDS, which is deployed on machines whose access requires high administrative privileges and which can be contacted only by few selected devices [47]. Hence, it is unrealistic to assume that an infected host can allow the attacker to access the ML-NIDS containing its detection model [86, 104].

An attacker could perform reconnaissance and lateral movements [15] to understand the network topology and eventually obtaining access to the NIDS machine. However, even in this unlikely case, to acquire power on the detection model the attacker must be able to inspect the underlying source-code. If the ML-NIDS is a commercial product, then such code may not be observable; whereas if the ML-NIDS is developed in-house, then the (human readable) source-code may be located on a different machine [23, 86].

We exclude the possibility of directly modifying the target ML-NIDS: In a similar circumstance, the entire defensive system would be completely overthrown and at complete disposal of the attacker. Any rational attacker who already managed to take control over the detection system will be in a position to achieve all of its goals in a more reliable and direct way rather than through subtle adversarial attacks.

In summary, it is unlikely that an adversary may get power on the internal configuration of the ML model (which is the case of white-box attacks).

4.3 Feature Set

By knowing the features that represent a given sample, the attacker is able to determine which operations are required to generate an adversarial example. For instance, if the ML-NIDS analyzes the duration of network communications, then an attacker can alter the length of the sessions between the controlled hosts. Obtaining power on the actual set of features used by the ML model faces similar challenges of gaining access to the trained detection model [23, 86, 104]. A similar scenario occurs because the actual features representing each sample and used to perform the detection are specified only at inference time.

However, attackers may leverage their domain expertise to guess which features are likely to be analyzed by the detector [23, 34]. Although each ML-NIDS is unique, they all analyze network traffic either in the form of raw network packets or as derived network metadata. Hence a ML-NIDS may employ feature sets with many similarities and overlaps [14, 53, 82, 101]. Therefore, it is reasonable to assume that expert attackers will leverage such intelligence to estimate with high probability some features utilized by the actual detector, which will impact the performance. The adversarial samples will then revolve around perturbations of these features.

It is important to establish the relationship between power on the training data (see Section 4.1), and power on the feature set. The former does not necessarily lead to power on the latter. Manipulating the training data with perturbations of existing samples causes the modification of their features with consequences on the training phase. However, our definition of "power on the feature set" in Section 4 denotes the *knowledge* about the feature set. As an example, if attackers have write access, then they can inject (directly or indirectly) new samples to the training data, but they would not know the actual features considered by the ML detector. Hence, write access to the training dataset does not lead to power on the feature set.

The situation may differ when an attacker has a read access on the training dataset, and the features are distinguishable. In such a case, an attacker may acquire some knowledge and power on the feature set. However, it is possible that the actual features used by the model are computed at inference time [13]. For example, an organization may store the training dataset in the form of raw packet captures, but the detector may be trained on network flows that are generated right before the training (or testing) phase. In such a scenario, even if attackers have power on the training data, they would not know the complete feature set used for the detection process although they can guess it and determine a portion of the actual feature set.

In summary, it is realistic to assume an attacker that has some power on the feature-set used by the ML-NIDS; having complete knowledge on this element is a tough challenge. Finally, it is unlikely that an attacker is completely oblivious of the composition of the feature-set.

4.4 Oracle

An attacker that has oracle power is able to reverse-engineer the target ML model by submitting some inputs and observing the corresponding output (see Section 2.2). In the specific case of ML-NIDS, to obtain oracle power the

attacker faces two obstacles: not triggering other detection mechanisms and extracting meaningful information from the input/output feedback.

Obstacle 1: Avoiding Detection. Modern organizations protect their networks through multiple defensive layers [73]. Hence, an attacker must operate in such a way to avoid being detected by these additional detection schemes. To this purpose, the attacker should aim at minimizing the amount of queries issued to the target NIDS [41, 52], since each query requires to create and send some additional anomalous traffic to the target network. Furthermore, these queries should be performed in a low-and-slow approach, because excessive queries in a short time frame may easily trigger alerts by detection systems that leverage simple statistical approaches to model the normal network traffic [64, 73]. Similar methodologies require an extended amount of time, up to days or weeks. These operations increase the probability that the attacker is detected. At the very least, they increase the length and cost of the offensive campaign [45]. In summary, an attacker can interact with the NIDS by sending some queries, but any rational attacker will try its best to limit the number of interactions.

Obstacle 2: Acquiring Feedback. The output of the detection may not be directly observable by an attacker. When a NIDS identifies a malicious event, it generates *alerts*, which are only notified to security administrator [57] through the NIDS console. In other words, even if attackers perturb some input samples, they cannot reliably receive any feedback from the NIDS. To witness the results of their own actions, attackers need to wait until the human operator, after triaging the alerts raised by the malicious samples, applies policies that specifically address the samples issued by the attacker. Thus, in these circumstances the adversary must rely on the defender's reactions, which is an unreliable attacking approach that may require long timespans. We identify three situations where the attacker can directly observe some information about the classification output generated by the NIDS after a given query.

- The first scenario requires the NIDS to be integrated in some reactive defensive mechanism that is able to automatically stop the detected malicious traffic. In this case an attacker can obtain the input/output pairing, for example by investigating which network communication of the controlled machines are (or not) received by the external hosts. Although the use of similar defensive systems is common in modern environments, we observe that the attacker must be aware that these mechanisms operate with the NIDS and that they are configured to block the specific traffic generated by the attacker. Acquiring such information requires significant intelligence expertise.
- The second scenario requires the attacker to gain access to the NIDS logs or its console, but this is unlikely, because such data are accessible only by the NIDS administrator.
- The third scenario requires the organization to rely on a commercial NIDS. In a similar case, the attackers could acquire the same NIDS and deploy it in a controlled environment where they have complete freedom: They may still be unable to determine the inner configurations of the ML components, but they would not be subject to any limitations of the query. A similar scenario is rather unrealistic as the attacker needs to know the NIDS product used by the target organization, which may require extended intelligence operations. They also must acquire the product by the third party vendor, which increases the cost of the offensive campaign; finally, they have to reproduce a network environment that resembles that of the target organization. All these requirements make this scenario feasible only for skilled and highly motivated adversaries.

We observe that attackers willing to acquire oracle power must overcome both obstacles (that is, avoiding detection, and receiving feedback). Attackers that can perform an unlimited amount of queries without being detected do not have any oracle power if they cannot reliably determine the input-output association. Similarly, attackers that can obtain the input-output association but may be detected during the process.

We can conclude that using the NIDS as an oracle is a tough challenge. In real scenarios, motivated attackers may get some feedback but with many limitations (e.g., few queries, or uncertain ML predictions), while a

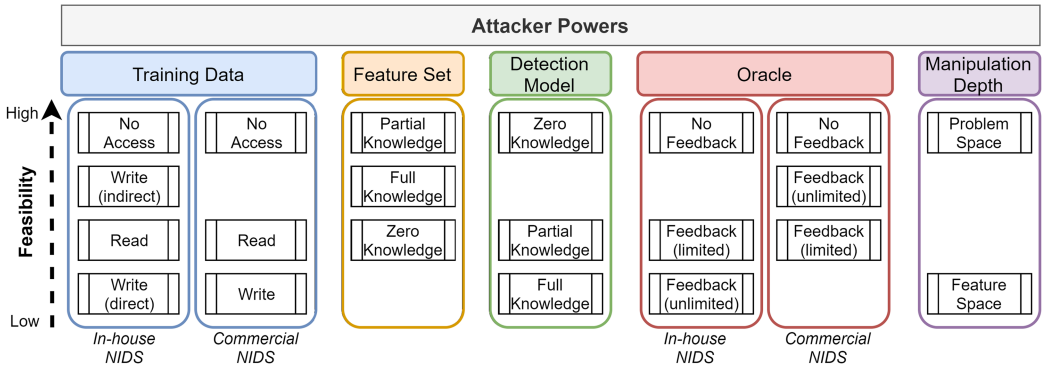

Fig. 4. Feasibility of each power available to the attacker.

complete oracle power is feasible only if the organization employs a commercial NIDS, or if the NIDS itself is compromised.

4.5 Manipulation Depth

The last element on which the attacker has power involves the data manipulation capabilities: the (adversarial) perturbations can be introduced either at the raw traffic level, or after the network data is being transformed into its higher-level feature representation. Indeed, an emerging topic in adversarial ML literature is the differentiation between *feature*-space and *problem*-space attacks [39, 43, 74]. The former implies that the adversarial perturbation is applied directly to the sample that is provided as an input to the ML model. Problem-space attacks involve obtaining perturbations by performing all the operations at the lower data level. As a practical example, consider a ML-NIDS that works on network flows: A feature space attack may involve increasing the value of one feature of the flow sample (e.g., the duration). A problem-space attack requires modifying the malware logic so as to produce network packets that, when exported to network flows, result in samples with increased duration (e.g., by adding some communication delays). In summary, feature-space attacks are a higher-level abstraction of the attacker's workflow focusing only on its intended results, whereas problem-space attacks involve the reproduction of the entire malicious procedure.

For these reasons, attacks at the problem-space are more representative of a realistic scenario. The only way in which an adversary could perform a real feature-space attack is by manipulating the conversion of the raw traffic data into its feature representation, which requires full power on the ML component.

4.6 Final Considerations

By taking into account the attacker power on all five elements, we summarize the considerations in Figure 4 representing the powers on the target NIDS. For each power, the figure shows the feasibility of its different forms as in Section 4. These forms are vertically ordered on the basis of their realistic feasibility: The most likely are placed in higher positions, and the least likely in lower positions. For the Training Data and Oracle powers, we distinguish the commercial NIDS on the right, and the in-house solutions on the left.

We remark that any attack is possible, but some require a higher amount of resources to be carried out, which may discourage their actuation. The aim of Figure 4 is to help determining which circumstances are more likely to occur in a real adversarial attacks against ML-NIDS.

In the most realistic attacks the attacker manipulates the real network traffic (where the perturbations occur at the *problem space*, rather than at the *feature space*) with just a *partial knowledge* of the features, *no feedback* from the NIDS, and *no access* to the training data. These powers do not require an attacker to previously compromise

the system nor to acquire knowledge about some proprietary and possibly well protected information about the NIDS and the target organization. Hence, similar attacks can be deployed even by attackers with an average skill and at low costs.

Adversarial attacks relying on the oracle power are feasible only if the target organization leverages a commercial NIDS that the attacker can also buy to freely experiment with. However, even in this situation the feasibility might be limited by non-standard configurations employed in the target organization that the attacker will likely not know and replicate on their testing environment.

Attacks that require power on the training data or on the detection model of a NIDS that is built and maintained in-house by the target organization, or a full knowledge of the feature set, are unrealistic, since these powers can only be obtained by an attacker that already managed to compromise the systems belonging to the target network that store these information. The same considerations also apply to attacks that require the use of the detection system as an oracle. In principle, a similar result can be achieved by an attacker that submits malicious traffic samples to a network intrusion prevention system (i.e., a detection system that is also configured to block malicious network communications). However this allows only to achieve partial knowledge, increases the cost and duration of the attack campaign and also increases the likelihood that these ancillary attack activities might trigger some other alert.

The most unrealistic attacks are those requiring full power over the training data or on the detection model of a commercial IDS, since the attacker should be able to compromise the vendor of the commercial defensive solution. Even attacks that require power on the detection model of a commercial NIDS are quite unrealistic, because they require an attacker to arbitrarily modify the ML-based detector executed by the commercial NIDS appliance acquired by the target organization. Finally, attacks that can only be performed at the feature space fall in this category, since the attacker would need to compromise the component of the detector that extracts features from the raw traffic.

We conclude with an important observation: *poisoning* attacks at the training-phase can only be performed if the attacker has power (in the form of write access) on the training data. Otherwise, the attacker is limited to attacks at inference-time.

Poisoning attacks can be extremely powerful, but they can be avoided by negating attacker power on the training set. However, devising threat models for attacks at inference-time is more complex, because the attacker can leverage a wider array of powers. Hence, attacks at inference time may be less disruptive, but they are more difficult to prevent.

5 REALISTIC EVALUATION OF EXISTING ATTACKS

We now evaluate the adversarial attacks against NIDS proposed in the literature by applying the proposed modeling guidelines, with the goal of analyzing the maturity and realism of existing examples. We initially describe some operations required to simulate realistic adversarial attacks. Then we provide an overview of the state of the art (Section 5.1) and finally describe the details of some cases (Section 5.2).

To reproduce realistic adversarial attacks against NIDS it is necessary to ensure that the perturbations maintain the malicious logic of the original sample. Indeed, to evade a ML detector it would suffice to generate a completely new malware variant that is not represented by any malicious sample contained in the training dataset. However, doing so would defeat the purpose of adversarial attacks, which involves the application of small and imperceptible modifications [70, 91]. Hence, the modified samples need to only slightly differ from their original variants, and they must also maintain their underlying malicious logic without triggering other detection mechanisms [14, 52, 75, 95, 101]. While such conditions are verifiable for attacks performed in the problem-space, attacks performed at the feature-space require additional verification steps [34].

Furthermore, when simulating attacks at the feature-space it is also needed to check that all inter-dependencies between features are maintained, and that the feature values of the perturbed samples do not result in impossible

Table 1. Characteristics of Existing Adversarial Attacks against ML-NIDS

Paper	Year	Power					Constraints Verified?	Type	Dataset
		Training Data	*Feature set*	*Detection model*	*Oracle*	*Manipulation depth*			
Kloft et al. [48]	2010	W	—	Full	—	feature	✗	poisoning	KDD99
Biggio et al. [18]	2013	—	Full	Partial	—	feature	✗	inference	custom
Sethi et al. [81]	2018	—	Full	—	—	feature	✓	inference	KDD99
Apruzzese et al. [11, 14]	2018	—	Partial	—	—	feature	✓	inference	CTU13, Botnet2014 IDS2017, CICIDS2018
Warzynski et al. [100]	2018	—	Full	Full	—	feature	✗	inference	KDD99
Lin et al. [58]	2018	—	—	—	∞	feature	✓	inference	KDD99
Li et al. [55]	2018	R, W	Partial	—	—	feature	✗	poisoning	KDD99, Kyoto[89]
Wang et al. [99]	2018	—	Full	Full	—	feature	✗	inference	KDD99
Yang et al. [104]	2018	R	Partial	—	∞	feature	✓	inference	KDD99
Marino et al. [61]	2018	—	Full	Full	—	feature	✗	inference	KDD99
Apruzzese et al. [13]	2019	R, W	Partial	—	—	feature	✗	poisoning	CTU13
Hashemi et al. [35, 36]	2019	—	Full	Full	—	feature	✗	inference	IDS2017
Usama et al. [95]	2019	—	—	—	∞	feature	✓	inference	KDD99
Khamis et al. [3]	2019	—	Full	Full	—	feature	✗	inference	UNSW-NB15
Clemens et al. [24]	2019	—	Full	Full	—	feature	✗	inference	Kitsune
Ibitoye et al. [40]	2019	—	Full	Full	—	feature	✗	inference	BoT-IoT[49]
Aiken et al. [4]	2019	—	Partial	—	—	problem	✓	inference	IDS2017
Yan et al. [103]	2019	—	—	—	∞	feature	✓	inference	KDD99, IDS2017
Martins et al. [62]	2019	—	Full	Full	—	feature	✗	inference	KDD99, IDS2017
Usama et al. [96]	2019	—	—	—	∞	problem	✓	inference	Tor-nonTor[53]
Peng et al. [72]	2019	—	—	—	∞	feature	✓	inference	KDD99, IDS2017
Kuppa et al. [52]	2019	—	Partial	—	100s	feature	✓	inference	CICIDS2018
Wu et al. [101]	2019	—	Partial	—	10s	problem	✓	inference	CTU13
Ayub et al. [16]	2020	—	Full	Full	—	feature	✗	inference	IDS2017, TRAbID2017[98]
Novo et al. [66]	2020	—	Full	Full	—	feature	✓	inference	MTA[2]
Apruzzese et al. [9]	2020	—	Partial	—	—	feature	✓	inference	CTU13
Chernikova et al. [23]	2020	—	Partial	—	∞	feature	✓	inference	CTU13
Sadeghzadeh et al. [80]	2020	—	Full	Full	—	problem	✓	inference	ICSX2016
Chernikova et al. [23]	2020	—	Full	Full	—	feature	✓	inference	CTU13
Piplai et al. [75]	2020	—	Full	Full	—	feature	✗	inference	BigDataCup2019[44]
Alhajjar et al. [6]	2020	—	Full	Full	—	feature	✗	inference	KDD99, UNSW-NB15
Apruzzese et al. [10]	2020	—	Partial	—	20s	feature	✗	inference	CTU13, Botnet2014
Han et al. [34]	2020	—	Partial	—	100s	problem	✓	inference	Kitsune, IDS2017
Pawlicki et al. [71]	2020	—	Full	Full	—	feature	✗	inference	IDS2017
Shu et al. [87]	2020	—	Full	Partial	50s	feature	✗	inference	IDS2017
Papadopoulos et al. [68]	2021	R, W	Full	—	—	feature	✗	poisoning	Bot-IoT
Pacheco et al. [67]	2021	—	Full	Full	—	feature	✗	inference	UNSW-NB15
Anthi et al. [8]	2021	R	Full	—	—	feature	✗	inference	ICSX2016
Papadopoulos et al. [68]	2021	—	Full	Full	—	feature	✗	inference	Bot-IoT

numbers: for example, the size of a TCP packet cannot exceed 64 KBytes, while some network flow collectors have fixed thresholds for the maximum flow duration [15, 72].

To evaluate realistic adversarial attacks against NIDS, it is necessary to ensure that all these properties are preserved [35, 58].

5.1 Overview

We perform an extensive literature survey on adversarial attacks against ML-NIDS. To the best of our knowledge, the overall results reported in Table 1 represent the state of the art until March 2021.

For each paper, we report its publication date, and the power of the considered attacker for each element described in Section 4.

- *Training Data*: The attacker can have Read, Write or no access (R, W, or "—", respectively).
- *Feature Set* and *Detection Model*: The attacker can have Full, Partial or no (denoted with "—") knowledge.
- *Oracle*: Papers where the NIDS is used as an oracle always assume that the attacker has full feedback on the the input-output association. We use "∞" to denote papers that do not set any boundary on the amount of queries available to the attacker; for proposals that consider a constrained attacker, we report the order of magnitude of the required interactions (e.g., 10s denotes tens of queries); we use "—" for papers where the attacker does not use the NIDS as an oracle.
- We also report if the manipulation occurs at the *feature* or *problem* space.

Moreover, we specify whether the authors consider perturbations that preserve the maliciousness and integrity of the original sample (✓) or not (✗). Finally, in the two rightmost columns we report the type of attack (inference or poisoning), and the data sets used for the experimental testbed. If a paper considers different attack scenarios, then it will have more entries in the table. As an example, let us consider the scenario proposed in Reference [55]: This paper presents a poisoning attack where the adversary is assumed to have both Read and Write access to the training data, and has partial knowledge of the features analyzed by the ML-NIDS; however, they have no knowledge on the internal configuration of the detector, and cannot use it as an oracle. Finally, the proposed attack involves manipulations occurring in the feature space, and the authors do not verify that the adversarial samples preserve their realistic integrity.

From this table, we can express several considerations. The initial studies considered an outdated and widely deprecated dataset (KDD99 [83]). More recent works include additional datasets in their experiments, such as the CTU-13 [30], the Kitsune [24], the UNSW-NB15 [65], the ICSX2016 [29], or the very recent IDS2017 and CICIDS2018 [83]. This is an important improvement, because evaluating adversarial attacks on multiple datasets increases the impact and value of the experiments. Furthermore, by considering more recent data, these results allow the researchers to understand the effectiveness of their threats in modern defensive scenarios.

We also observe that just a minority of proposals considers poisoning attempts [13, 48, 55], while the majority of them focuses on evasion attacks at inference time. This is a relevant trend, because, as discussed in Section 4.1, poisoning attacks are difficult to perform in real scenarios due to the difficulty faced by attackers to acquire some power on the training data.

Assuming that the ML-NIDS can be used as an oracle that answers to an unlimited amount of queries is possible only if an attacker can acquire a perfect replica the detector. However, only some recent papers (e.g., References [34, 52, 101]) consider attackers with a limited number of queries. This is an important step toward realistic security scenarios.

A large amount of proposals verify whether the modified samples preserve or not their integrity and their malicious logic. Few papers regard attacks at problem space (e.g., References [34, 80, 96]). This choice evidences the possibility of novel research opportunities that can evaluate more complex but also more realistic adversarial samples.

5.2 Case Studies

We consider three papers [4, 55, 101] from Table 1, because they represent meaningful examples for adversarial attacks at different degrees of feasibility. Then, for comparison purposes we analyze a famous case of a real adversarial attack against a real detector [56].

The work in Reference [4] involves evasion attacks at inference time against NIDS. The adversary has very little power on the target system: They have no access to the training set and cannot interact with the detector in any way (neither to inspect its internal configuration, nor to leverage it as an oracle). The only assumption is that the attacker knows some of the features used by the detection mechanism. Thus, the strategy consists in modifying the payloads of the raw network traffic (resulting in a problem-space attack), so as to induce perturbations of these features that induce the model to misclassify the malicious network traffic. In their experiments,

Aiken et al. [4] show the performance drop of state-of-the-art classifiers against these attacks: the baseline 99.9% detection rate decreases to an unacceptable 70%. There are several elements that make the scenario described in this paper as very realistic. First, the attacker is not assumed to have access to some of the most protected devices in the target system (that is, the detector, and the server hosting the training data); the only power available to the attacker is the knowledge of a subset of the features adopted by the classification mechanism, which is a feasible assumption (see Section 4.3). Furthermore, the operations performed to apply the perturbations are easily achievable for any attacker that has established a foothold in a network environment. Finally, the target detector is trained on a recent dataset (the IDS2017), representing modern network environments. We conclude that the attack represented in this paper portrays a complete and realistic adversarial scenario.

The authors of Reference [101] consider attacks at inference time against botnet detectors. The adversary plans to use an autonomous deep reinforcement learning agent to evade a classifier based on network flows. The paper assumes an attacker that is able to inspect the feedback of the detector to a given sample (they can use it as an oracle) with no limitation to the amount of queries that can be performed. The attacker also guesses that the detector analyzes network flows, and they are aware of the possible features employed. However, they do not have any power on the training data, and have no knowledge of the ML-model integrated in the NIDS. The samples are generated at the problem space, because the agent modifies the raw (malicious) network packets by adding redundant data in the payload, therefore altering the network flows that are forwarded to the NIDS. With these settings, the authors of Reference [101] show that the proposed agent is able to generate adversarial samples that evade detection in almost 80% of the cases, by requiring only dozens of queries. Based on these assumptions, we consider this paper to represent a realistic but less feasible scenario than the one in Reference [4], due to the additional presence of the oracle power. As explained in Section 4.4, receiving direct feedback from the NIDS requires that the attacker either (i) acquires the same NIDS and uses it in a dedicated environment (if the NIDS used by the target is produced by a third-party vendor) or (ii) that the NIDS also integrates an IPS that can be leveraged to determine which communications are blocked. In the former case, the attacker needs to invest resources to acquire and deploy the NIDS, which is feasible but increases the cost of the campaign. In the latter case, the attacker must ensure that the performed queries do not trigger an excessive amount of alarms: we consider the few dozens queries required in Reference [101] to be an acceptable number that is unlikely to induce security personnel to manually investigate the infected devices and triage them.

The third case study [55] describes a poisoning attack where the attackers are assumed to have power on training data. They know the entire composition of the data used to train the model, because they have full reading access to the training dataset. They also have write access. Although they cannot change the label (as in label flipping [93]) or modify the features of existing training samples, they are allowed to inject new samples in the training dataset. The adversary also knows the features used by the detector—which correspond to the features representing the samples of the training dataset. The attacker thus plans to add some adversarial samples in the training set to modify the decision boundaries of the detection model, thus ensuring that malicious samples remain undetected. By executing these operations, Li et al. [55] show that the accuracy of detectors could drop from over 95% to nearly 50%, making the target detector impractical. We consider this scenario as unrealistic. As explained in 4.1, obtaining both read and write access to the training data is a daunting task even for expert adversaries. Having complete knowledge of the features used by the detection model is not realistic: Even if the attacker has access to the training data and knows the features associated to each sample, it is likely that the feature-set used by the detection model presents some differences. For example, it may include some derived features computed right before the training operations [13], or it may even discard some features that lead to unfavorable performance. Finally, we remark that the targeted detectors are trained on two outdated datasets (KDD99 and Kyoto2006) that do not capture the characteristics of the network traffic generated by modern organizations and recent attacks [83].

Finally, we consider an interesting attack against a real detection system that is embedded into the phishing detector of the Google Chrome browser [56]. Although this work is not included in Table 1 because the target

system is not a NIDS, we find it useful to summarize its circumstances, because it can be inspirational for future researches. The considered scenario involves a ML-detection system that is deployed on a popular browser that is obtainable by any user. The (trained) detection model is embedded into the application, so it is impossible to affect the training phase. Furthermore, the underlying application code is not readable, hence it is impossible to determine which features are used to perform the analysis, nor to acquire any information about the actual detection model. However, the accessibility of the application, and therefore of its embedded detection system, allows a user to have complete *oracle* power, with an unlimited amount of queries and direct access to the feedback of the input/output pair. In other words, a user can craft a Web page, and then visit such page with the browser: If the browser alerts the user, then it means that the page is considered as malicious; otherwise, the page is considered as benign. An attacker with a similar power can create a wide array of pages to reverse engineer the classifier used to perform the detection. Consequently, the authors of [56] were able to gain important information about the detection system, such as which ML algorithm was used, the parameters, and the most significant classification features. This allowed the attackers to identify the criteria used to pinpoint whether a Web page was malicious or not and it allowed to determine how to evade a similar system. The authors were able to camouflage a Web page (in the *problem* space) so that its malicious score dropped from 0.99 to 0.45. The attack in Reference [56] is an example of a real use case where the attackers have no power on the training data, the feature set, and the detection model; they operate at the problem space, but they have complete oracle power. The vulnerability is represented by the deployment of the detection system at the client level: A similar attack would be more difficult if the ML model was deployed on a dedicated, controlled, and remote server. This is the typical case of a NIDS and motivates the difficulty of obtaining a complete oracle power in the reality.

6 CONCLUSIONS

Adversarial attacks represent a threat affecting the reliability of any cyber defense relying on artificial intelligence. The literature has shown the poor performance of ML-NIDS against adversarial perturbations, but few papers analyze this emerging menace by taking into account the realistic characteristics of modern environments. The consequence is that many threat models are practically unfeasible. By assuming a cybersecurity perspective, we identify and model the different elements of the target systems that can be leveraged by an attacker to carry out an adversarial attack against ML-NIDS. We discuss each of these elements by describing the realistic circumstances that allow attackers to seize its control. The proposed model is then applied to analyze several papers presenting adversarial attacks against ML-NIDS. On the base of the identified parameters, we can easily identify papers considering scenarios that are more likely (or unlikely) to result in real adversarial attacks. We can conclude that many papers assume adversarial threat models that are inapplicable to realistic ML-NIDS, but recent works consider some real obstacles that attackers need to overcome to bypass or perturb detection. This article can guide researchers in devising threat models that are more representative of real defensive settings, and motivates the need for additional and more realistic research on adversarial attacks against ML-NIDS. The identification of the main defensive vulnerabilities and the prioritization against known adversarial attacks allows security experts to harden ML-based defensive systems. Our article is specifically oriented to Network Intrusion Detection problems, but some analyses and conclusions can be applied to other cyber detection problems, such as phishing and malware detection. These contexts can represent interesting applications for extensions in future work.

REFERENCES

[1] 2020. A Complete List of All (arXiv) Adversarial Example Papers. Retrieved from https://nicholas.carlini.com/writing/2019/all-adversarial-example-papers.html.

[2] 2020. Malware Traffic Analysis Dataset. Retrieved from https://malware-traffic-analysis.net/.

[3] Rana Abou Khamis, M. Omair Shafiq, and Ashraf Matrawy. 2020. Investigating resistance of deep learning-based IDS against adversaries using min-max optimization. In *Proceedings of the IEEE International Conference Commun.* 1–7.

[4] J. Aiken and S. Scott-Hayward. 2019. Investigating adversarial attacks against network intrusion detection systems in SDNs. In *Proceedings of the IEEE Conference Network Function Virtualization and Software Defined Network*. 1–7.

[5] Naveed Akhtar and Ajmal Mian. 2018. Threat of adversarial attacks on deep learning in computer vision: A survey. *IEEE Access* 6 (2018), 14410–14430.

[6] Elie Alhajjar, Paul Maxwell, and Nathaniel D. Bastian. 2020. Adversarial machine learning in network intrusion detection systems. *Expert Systems with Applications* 186 (2021), 115782.

[7] Mauro Andreolini, Marcello Pietri, Stefania Tosi, and Andrea Balboni. 2014. Monitoring large cloud-based systems. In *Proceedings of the 4th International Conference on Cloud Computing and Services Science (CLOSER'14)*. SciTePress, 341–351.

[8] Eirini Anthi, Lowri Williams, Matilda Rhode, Pete Burnap, and Adam Wedgbury. 2021. Adversarial attacks on machine learning cybersecurity defences in industrial control systems. *J. Inf. Secur. Appl.* 58 (2021), 102717.

[9] Giovanni Apruzzese, Mauro Andreolini, Michele Colajanni, and Mirco Marchetti. 2020. Hardening random forest cyber detectors against adversarial attacks. *IEEE Trans. Emerg. Top. Comp. Intell.* 4, 4 (2020), 427–439.

[10] Giovanni Apruzzese, Mauro Andreolini, Mirco Marchetti, Andrea Venturi, and Michele Colajanni. 2020. Deep reinforcement adversarial learning against botnet evasion attacks. *IEEE Trans. Netw. Serv. Manag.* 17, 4 (2020), 1975–1987.

[11] Giovanni Apruzzese and Michele Colajanni. 2018. Evading botnet detectors based on flows and random forest with adversarial samples. In *Proceedings of the IEEE International Symposium Network Computing and Applications* 1–8.

[12] Giovanni Apruzzese, Michele Colajanni, Luca Ferretti, Alessandro Guido, and Mirco Marchetti. 2018. On the effectiveness of machine and deep learning for cybersecurity. In *Proceedings of the IEEE International Conference on Cyber Conflicts*. 371–390.

[13] Giovanni Apruzzese, Michele Colajanni, Luca Ferretti, and Mirco Marchetti. 2019. Addressing adversarial attacks against security systems based on machine learning. In *Proceedings of the IEEE International Conference on Cyber Conflicts*. 1–18.

[14] Giovanni Apruzzese, Michele Colajanni, and Mirco Marchetti. 2019. Evaluating the effectiveness of Adversarial attacks against botnet detectors. In *Proceedings of the IEEE International Symposium Network Computing and Applications*. 1–8.

[15] Giovanni Apruzzese, Fabio Pierazzi, Michele Colajanni, and Mirco Marchetti. 2017. Detection and threat prioritization of pivoting attacks in large networks. *IEEE Trans. Emerg. Top. Comput.* (2017).

[16] Md Ahsan Ayub, William A. Johnson, Douglas A. Talbert, and Ambareen Siraj. 2020. Model evasion attack on intrusion detection systems using adversarial machine learning. In *Proceedings of the IEEE Conference on Information Sciences and Systems*. 1–6.

[17] Daniel S. Berman, Anna L. Buczak, Jeffrey S. Chavis, and Cherita L. Corbett. 2019. A survey of deep learning methods for cyber security. *Information* 10, 4 (2019), 122.

[18] Battista Biggio, Giorgio Fumera, and Fabio Roli. 2013. Security evaluation of pattern classifiers under attack. *IEEE Trans. Knowl. Data Eng.* 26, 4 (2013), 984–996.

[19] Battista Biggio and Fabio Roli. 2018. Wild patterns: Ten years after the rise of adversarial machine learning. *Pattern Recogn.* 84 (2018), 317–331.

[20] Leyla Bilge, Davide Balzarotti, William Robertson, Engin Kirda, and Christopher Kruegel. 2012. Disclosure: Detecting botnet command and control servers through large-scale netflow analysis. In *Proceedings of the ACM Annual Computing Security and Applications Conference*. 129–138.

[21] Anna L. Buczak and Erhan Guven. 2016. A survey of data mining and machine learning methods for cyber security intrusion detection. *IEEE Commun. Surv. Tut.* 18, 2 (2016), 1153–1176.

[22] Manuel Eugenio Morocho Cayamcela and Wansu Lim. 2018. Artificial intelligence in 5G technology: A survey. In *Proceedings of the International Conference Information and Communication Technology Convergence*. 860–865.

[23] Alesia Chernikova and Alina Oprea. 2020. FENCE: Feasible evasion attacks on neural networks in constrained environments. arXiv:1909.10480. Retrieved from https://arxiv.org/abs/1909.10480.

[24] Joseph Clements, Yuzhe Yang, Ankur Sharma, Hongxin Hu, and Yingjie Lao. 2019. Rallying adversarial techniques against deep learning for network security. In *IEEE Symposium Series on Computational Intelligence (SSCI'21)*. IEEE, 01–08.

[25] European Commission. 2020. *White Paper: On Artificial Intelligence—A European Approach to Excellence and Trust*. Technical Report COM (2020) 65 final. European Commission, Brussels.

[26] Igino Corona, Giorgio Giacinto, and Fabio Roli. 2013. Adversarial attacks against intrusion detection systems: Taxonomy, solutions and open issues. *Inf. Sci.* 239 (2013), 201–225.

[27] Michael J. De Lucia and Chase Cotton. 2019. Adversarial machine learning for cyber security. *J. Inf. Syst. Appl. Res.* 12, 1 (2019), 26.

[28] Dorothy E. Denning. 1987. An intrusion-detection model. *IEEE T. Soft. Eng.* 2 (1987), 222–232.

[29] Gerard Draper-Gil, Arash Habibi Lashkari, Mohammad Saiful Islam Mamun, and Ali A. Ghorbani. 2016. Characterization of encrypted and vpn traffic using time-related. In *Proceedings of the International Conference on Information Systems Secururity and Privacy*. 407–414.

[30] Sebastian Garcia, Martin Grill, Jan Stiborek, and Alejandro Zunino. 2014. An empirical comparison of botnet detection methods. *Comput. Secur.* 45 (2014), 100–123.

[31] Joseph Gardiner and Shishir Nagaraja. 2016. On the security of machine learning in malware C&C detection: A survey. *ACM Comput. Surv.* 49, 3 (2016), 59.

[32] Sanjam Garg, Somesh Jha, Saeed Mahloujifar, Mohammad Mahmoody, and Abhradeep Thakurta. 2020. Obliviousness makes poisoning adversaries weaker. arXiv:2003.12020. Retrieved from https://arxiv.org/abs/2003.12020.
[33] Zhenyu Guan, Liangxu Bian, Tao Shang, and Jianwei Liu. 2018. When machine learning meets security issues: A survey. In *Proceedings of the International Conference on International Safety Robotics*. 158–165.
[34] Dongqi Han, Zhiliang Wang, Ying Zhong, Wenqi Chen, Jiahai Yang, Shuqiang Lu, Xingang Shi, and Xia Yin. 2020. Practical traffic-space adversarial attacks on learning-based NIDSs. arXiv:2005.07519. Retrieved from https://arxiv.org/abs/2005.07519.
[35] Mohammad J. Hashemi, Greg Cusack, and Eric Keller. 2019. Towards evaluation of NIDSs in adversarial setting. In *Proceedings of the ACM CoNEXT Workshop Big Data, Machine Learning and Artificial Intelligence for Data Commununication Networks*. 14–21.
[36] Mohammad J. Hashemi and Eric Keller. 2020. Enhancing robustness against adversarial examples in network intrusion detection systems. In *Proceedings of the IEEE Conference Netw. Funct. Virt. Softw. Defined Netw.* 37–43.
[37] Ling Huang, Anthony D. Joseph, Blaine Nelson, Benjamin I. P. Rubinstein, and J. D. Tygar. Oct. 2011. Adversarial machine learning. In *Proceedings of the ACM Workshop on Security and Artificial Intelligence*. 43–58.
[38] Yonghong Huang, Utkarsh Verma, Celeste Fralick, Gabriel Infantec-Lopez, Brajesh Kumar, and Carl Woodward. 2019. Malware evasion attack and defense. In *Proceedings of the IEEE International Conference Dependable Systems and Networks Workshops*. 34–38.
[39] Olakunle Ibitoye, Rana Abou-Khamis, Ashraf Matrawy, and M. Omair Shafiq. 2019. The threat of adversarial attacks on machine learning in network security—A survey. arXiv:1911.02621. Retrieved from https://arxiv.org/abs/1911.02621.
[40] Olakunle Ibitoye, Omair Shafiq, and Ashraf Matrawy. 2019. Analyzing adversarial attacks against deep learning for intrusion detection in IoT networks. In *Proceedings of the IEEE Global Communication Conference*. 1–6.
[41] Andrew Ilyas, Logan Engstrom, Anish Athalye, and Jessy Lin. 2018. Black-box adversarial attacks with limited queries and information. In *Proceedings of the International Conference on Machine Learning*. 2137–2146.
[42] Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. 2019. Adversarial examples are not bugs, they are features. *Advances in Neural Information Processing Systems* 32 (2019).
[43] Nathan Inkawhich, Wei Wen, Hai Helen Li, and Yiran Chen. 2019. Feature space perturbations yield more transferable adversarial examples. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 7066–7074.
[44] A. Janusz, D. Kałuza, A. Chądzyńska-Krasowska, B. Konarski, J. Holland, and D. Ślęzak. 2019. IEEE BigData 2019 Cup: Suspicious network event recognition. In *Proceedings of the IEEE International Conference on Big Data*. 5881–5887.
[45] Min Suk Kang, Virgil D. Gligor, Vyas Sekar, et al. 2016. SPIFFY: Inducing cost-detectability tradeoffs for persistent link-flooding attacks. In *Proceedings of the Network and Distributed System Security Symposium*
[46] Alex Kantchelian, J. Doug Tygar, and Anthony Joseph. 2016. Evasion and hardening of tree ensemble classifiers. In *Proceedings of the International Conference on Machine Learning*. 2387–2396.
[47] Karim Khalil, Zhiyun Qian, Paul Yu, Srikanth Krishnamurthy, and Ananthram Swami. 2016. Optimal monitor placement for detection of persistent threats. In *Proceedings of the IEEE Global Communication Conference* 1–6.
[48] Marius Kloft and Pavel Laskov. 2010. Online anomaly detection under adversarial impact. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*. 405–412.
[49] Nickolaos Koroniotis, Nour Moustafa, Elena Sitnikova, and Benjamin Turnbull. 2019. Towards the development of realistic botnet dataset in the internet of things for network forensic analytics: Bot-iot dataset. *Fut. Gener. Comput. Syst.* 100 (2019), 779–796.
[50] A. Kott and P. Theron. 2020. Doers, not watchers: Intelligent autonomous agents are a path to cyber resilience. *IEEE Secur. Priv.* 18, 3 (2020), 62–66.
[51] Ram Shankar Siva Kumar, Magnus Nyström, John Lambert, Andrew Marshall, Mario Goertzel, Andi Comissoneru, Matt Swann, and Sharon Xia. 2020. Adversarial machine learning—Industry perspectives. In *IEEE Security and Privacy Workshops (SPW'20)*. IEEE, 69–75.
[52] Aditya Kuppa, Slawomir Grzonkowski, Muhammad Rizwan Asghar, and Nhien-An Le-Khac. 2019. Black box attacks on deep anomaly detectors. In *Proceedings of the International Conference Availab., Reliab. Secur.* 1–10.
[53] Arash Habibi Lashkari, Gerard Draper-Gil, Mohammad Saiful Islam Mamun, and Ali A. Ghorbani. 2017. Characterization of tor traffic using time based features. In *Proceedings of the International Conference on Information Systems Security and Privacy*. 253–262.
[54] Pavel Laskov et al. 2014. Practical evasion of a learning-based classifier: A case study. In *Proceedings of the IEEE Symposium on Security and Privacy*. 197–211.
[55] Pan Li, Qiang Liu, Wentao Zhao, Dongxu Wang, and Siqi Wang. 2018. Chronic poisoning against machine learning based IDSs using edge pattern detection. In *Proceedings of the IEEE International Conference on Communications*. 1–7.
[56] Bin Liang, Miaoqiang Su, Wei You, Wenchang Shi, and Gang Yang. 2016. Cracking classifiers for evasion: A case study on the google's phishing pages filter. In *Proceedings of the International Conference World Wide Web*. 345–356.
[57] Hung-Jen Liao, Chun-Hung Richard Lin, Ying-Chih Lin, and Kuang-Yuan Tung. 2013. Intrusion detection system: A comprehensive review. *J. Netw. Comp. Appl.* 36, 1 (2013), 16–24.
[58] Zilong Lin, Yong Shi, and Zhi Xue. 2018. Idsgan: Generative adversarial networks for attack generation against intrusion detection. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 79–91.
[59] Qiang Liu, Pan Li, Wentao Zhao, Wei Cai, Shui Yu, and Victor C. M. Leung. 2018. A survey on security threats and defensive techniques of machine learning: A data driven view. *IEEE Access* 6 (2 2018), 12103–12117.

[60] Mirco Marchetti, Fabio Pierazzi, Alessandro Guido, and Michele Colajanni. 2016. Countering advanced persistent threats through security intelligence and big data analytics. In *Proceedings of the IEEE International Conference on Cyber Conflict (CyCon'16)*. NATO CCD COE, 243–261.
[61] Daniel Ll Marino, Chathurika Sl Wickramasinghe, and Milos Manic. 2018. An adversarial approach for explainable ai in intrusion detection systems. In *Proceedings of the IEEE Conference of the Industrial Electronics Society*. 3237–3243.
[62] Nuno Martins, José Magalhães Cruz, Tiago Cruz, and Pedro Henriques Abreu. 2019. Analyzing the footprint of classifiers in adversarial denial of service contexts. In *Proceedings of the Springer EPIA Conference on Artificial Intelligence*. 256–267.
[63] Nuno Martins, José Magalhães Cruz, Tiago Cruz, and Pedro Henriques Abreu. 2020. Adversarial machine learning applied to intrusion and malware scenarios: A systematic review. *IEEE Access* 8 (2020), 35403–35419.
[64] Sumit More, M. Lisa Mathews, Anupam Joshi, Tim Finin, et al. 2012. A semantic approach to situational awareness for intrusion detection. In *Proceedings of the National Symposium on Moving Target Research*.
[65] N. Moustafa and J. Slay. 2015. UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). In *Proceedings of the Military Communications and Information Systems Conference* 1–6.
[66] Carlos Novo and Ricardo Morla. 2020. Flow-based detection and proxy-based evasion of encrypted malware C2 traffic. In *Proceedings of the 13th ACM Workshop on Artificial Intelligence and Security*. 83–91.
[67] Yulexis Pacheco and Weiqing Sun. 2021. Adversarial machine learning: A comparative study on contemporary intrusion detection datasets. In *Proceedings of the International Conference on Information Systems Security and Privacy*, Vol. 1. 160–171.
[68] Pavlos Papadopoulos, Oliver Thornewill von Essen, Nikolaos Pitropakis, Christos Chrysoulas, Alexios Mylonas, and William J. Buchanan. 2021. Launching Adversarial Attacks against Network Intrusion Detection Systems for IoT. *J. Cybersecur. Priv.* 1, 2 (2021), 252–273.
[69] Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z. Berkay Celik, and Ananthram Swami. 2017. Practical black-box attacks against machine learning. In *Proceedings of the ACM Conference on Computing Communications and Security*. 506–519.
[70] Nicolas Papernot, Patrick McDaniel, Arunesh Sinha, and Michael Wellman. 2018. SoK: Security and privacy in machine learning. In *Proceedings of the IEEE European Symposium on Security and Privacy*. 399–414.
[71] Marek Pawlicki, Michał Choraś, and Rafał Kozik. 2020. Defending network intrusion detection systems against adversarial evasion attacks. *Fut. Gener. Comput. Syst.* 110 (2020), 148–154.
[72] Xiao Peng, Weiqing Huang, and Zhixin Shi. 2019. Adversarial attack against DoS intrusion detection: An improved boundary-based method. In *Proceedings of the IEEE International Conference Tools on Artificial Intelligence*. 1288–1295.
[73] Fabio Pierazzi, Giovanni Apruzzese, Michele Colajanni, Alessandro Guido, and Mirco Marchetti. 2020. Scalable architecture for online prioritisation of cyber threats. In *2020 IEEE symposium on security and privacy (SP)*. IEEE, 1332–1346.
[74] Fabio Pierazzi, Feargus Pendlebury, Jacopo Cortellazzi, and Lorenzo Cavallaro. 2020. Intriguing properties of adversarial ML attacks in the problem space. In *Proceedings of the IEEE Symposium on Security and Privacy*.
[75] Aritran Piplai, Sai Sree Laya Chukkapalli, and Anupam Joshi. 2020. NAttack! adversarial attacks to bypass a GAN based classifier trained to detect network intrusion. (2020), 49–54.
[76] Nikolaos Pitropakis, Emmanouil Panaousis, Thanassis Giannetsos, Eleftherios Anastasiadis, and George Loukas. 2019. A taxonomy and survey of attacks against machine learning. *Comput. Sci. Rev.* 34 (2019), 100199.
[77] Shilin Qiu, Qihe Liu, Shijie Zhou, and Chunjiang Wu. 2019. Review of artificial intelligence adversarial attack and defense technologies. *MDPI Appl. Sci.* 9, 5 (2019), 909.
[78] Ishai Rosenberg, Asaf Shabtai, Yuval Elovici, and Lior Rokach. 2021. Adversarial machine learning attacks and defense methods in the cyber security domain. *ACM Comput. Surv.* 54, 5 (2021), 1–36.
[79] K. Sadeghi, A. Banerjee, and S. K. S. Gupta. 2020. A system-driven taxonomy of attacks and defenses in adversarial machine learning. *IEEE Trans. Emerg. Top. Comput. Intell.* (2020), 1–18.
[80] Amir Mahdi Sadeghzadeh, Rasool Jalili, and Saeed Shiravi. 2021. Adversarial network traffic: Toward evaluating the robustness of deep learning based network traffic classification. *IEEE Transactions on Network and Service Management* 18, 2 (2021), 1962–1976.
[81] Tegjyot Singh Sethi and Mehmed Kantardzic. 2018. Data driven exploratory attacks on black box classifiers in adversarial domains. *Neurocomputing* 289 (2018), 129–143.
[82] K. Shanthi and D. Seenivasan. 2015. Detection of botnet by analyzing network traffic flow characteristics using open source tools. In *Proceedings of the IEEE International Conference on International Systems and Control*. 1–5.
[83] Iman Sharafaldin, Arash Habibi Lashkari, and Ali A. Ghorbani. 2018. Toward generating a new intrusion detection dataset and intrusion traffic characterization. In *Proceedings of the Springer International Conference Inf. Syst. Secur. Privacy*. 108–116.
[84] Mahmood Sharif, Sruti Bhagavatula, Lujo Bauer, and Michael K. Reiter. 2016. Accessorize to a crime: Real and stealthy attacks on state-of-the-art face recognition. In *Proceedings of the ACM Conference Comput. Commun. Secur.* 1528–1540.
[85] Alex Shenfield, David Day, and Aladdin Ayesh. 2018. Intelligent intrusion detection systems using artificial neural networks. *ICT Express* 4, 2 (2018), 95–99.
[86] S. Shetty, I. Ray, N. Ceilk, M. Mesham, N. Bastian, and Q. Zhu. 2019. Simulation for cyber risk management – where are we, and where do we want to go?. In *Proceedings of the IEEE Winter Simulation Conference* 726–737.

[87] Dule Shu, Nandi O. Leslie, Charles A. Kamhoua, and Conrad S. Tucker. 2020. Generative adversarial attacks against intrusion detection systems using active learning. In *Proceedings of the ACM Workshop on Wireless Security and Machine Learning*. 1–6.
[88] Robin Sommer and Vern Paxson. 2010. Outside the closed world: On using machine learning for network intrusion detection. In *Proceedings of the IEEE Symposium on Security and Privacy*. 305–316.
[89] Jungsuk Song, Hiroki Takakura, Yasuo Okabe, Masashi Eto, Daisuke Inoue, and Koji Nakao. 2011. Statistical analysis of honeypot data and building of Kyoto 2006+ dataset for NIDS evaluation. In *Proceedings of the Workshop on Building Analysis Datasets and Gathering Experience Returns for Security*. 29–36.
[90] Anna Sperotto, Gregor Schaffrath, Ramin Sadre, Cristian Morariu, Aiko Pras, and Burkhard Stiller. 2010. An overview of IP flow-based intrusion detection. *IEEE Commun. Surv. Tutor.* 12, 3 (2010), 343–356.
[91] Jiawei Su, Danilo Vasconcellos Vargas, and Kouichi Sakurai. 2019. One pixel attack for fooling deep neural networks. *IEEE Trans. Evol. Comput.* 23, 5 (2019), 828–841.
[92] Elham Tabassi, Kevin Burns, Michael Hadjimichael, Andres Molina-Markham, and Julian Sexton. 2019. *A Taxonomy and Terminology of Adversarial Machine Learning*. Technical Report. NIST.
[93] Rahim Taheri, Reza Javidan, Mohammad Shojafar, Zahra Pooranian, Ali Miri, and Mauro Conti. 2020. On defending against label flipping attacks on malware detection systems. *Neural Comput. Appl.* (2020), 1–20.
[94] Thanh Cong Truong, Quoc Bao Diep, and Ivan Zelinka. 2020. Artificial intelligence in the cyber domain: Offense and defense. *Symmetry* 12, 3 (2020), 410.
[95] Muhammad Usama, Muhammad Asim, Siddique Latif, Junaid Qadir, et al. 2019. Generative adversarial networks for launching and thwarting adversarial attacks on network intrusion detection systems. In *Proceedings of the International IEEE Conference on Wireless Communications and Mobile Computing*. 78–83.
[96] Muhammad Usama, Adnan Qayyum, Junaid Qadir, and Ala Al-Fuqaha. 2019. Black-box adversarial machine learning attack on network traffic classification. In *Proceedings of the IEEE International Conference on Wireless Communications and Mobile Computing*. 84–89.
[97] Kush R. Varshney. 2019. Trustworthy machine learning and artificial intelligence. *XRDS: Crossroads* 25, 3 (2019), 26–29.
[98] Eduardo K. Viegas, Altair O. Santin, and Luiz S. Oliveira. 2017. Toward a reliable anomaly-based intrusion detection in real-world environments. *Comput. Netw.* 127 (2017), 200–216.
[99] Zheng Wang. 2018. Deep learning-based intrusion detection with adversaries. *IEEE Access* 6 (2018), 38367–38384.
[100] Arkadiusz Warzyński and Grzegorz Kołaczek. 2018. Intrusion detection systems vulnerability on adversarial examples. In *Proceedings of the IEEE Conference on Innovations in Intelligent Systems and Applications*. 1–4.
[101] Di Wu, Binxing Fang, Junnan Wang, Qixu Liu, and Xiang Cui. 2019. Evading machine learning botnet detection models via deep reinforcement learning. In *Proceedings of the IEEE International Conference on Communications*. 1–6.
[102] Yang Xin, Lingshuang Kong, Zhi Liu, Yuling Chen, Yanmiao Li, Hongliang Zhu, Mingcheng Gao, Haixia Hou, and Chunhua Wang. 2018. Machine learning and deep learning methods for cybersecurity. *IEEE Access* 6 (2018), 35365–35381.
[103] Qiao Yan, Mingde Wang, Wenyao Huang, Xupeng Luo, and F. Richard Yu. 2019. Automatically synthesizing DoS attack traces using generative adversarial networks. *International J. Mach. Learn. Cyber.* 10, 12 (2019), 3387–3396.
[104] Kaichen Yang, Jianqing Liu, Chi Zhang, and Yuguang Fang. 2018. Adversarial examples against the deep learning based network intrusion detection systems. In *Proceedings of the IEEE Military Communications Conference* 559–564.
[105] Xiaoyong Yuan, Pan He, Qile Zhu, and Xiaolin Li. 2019. Adversarial examples: Attacks and defenses for deep learning. *IEEE Trans. Neural Netw. Learn. Syst.* 30, 9 (2019), 2805–2824.
[106] Stefano Zanero and Sergio M. Savaresi. 2004. Unsupervised learning techniques for an intrusion detection system. In *Proceedings of the ACM Symposium on Applied Computing*. 412–419.
[107] Wei Emma Zhang, Quan Z. Sheng, Ahoud Alhazmi, and Chenliang Li. 2020. Adversarial attacks on deep-learning models in natural language processing: A survey. *ACM Trans. Intell. Syst. Technol.* 11, 3 (2020), 1–41.
[108] Yan Zhou, Murat Kantarcioglu, and Bowei Xi. 2019. A survey of game theoretic approach for adversarial machine learning. *Data Min. Knowl. Discov.* 9, 3 (2019), e1259.
[109] Daniel Zügner, Amir Akbarnejad, and Stephan Günnemann. 2018. Adversarial attacks on neural networks for graph data. In *Proceedings of the ACM SIGKDD International Conference Knowledge Discovery and Data Mining*. 2847–2856.

Received November 2020; revised May 2021; accepted June 2021