Investigating Adversarial Attacks against Network Intrusion Detection Systems in SDNs

James Aiken and Sandra Scott-Hayward
Centre for Secure Information Technologies, Queen's University Belfast, Belfast, BT3 9DT, N. Ireland
Email: jaiken06@qub.ac.uk, s.scott-hayward@qub.ac.uk

***Abstract*—Machine-learning based network intrusion detection systems (ML-NIDS) are increasingly popular in the fight against network attacks. In particular, promising detection results have been demonstrated in conjunction with Software-Defined Networks (SDN), in which the logically centralized control plane provides access to data from across the network. However, research into adversarial attacks against machine learning classifiers has highlighted vulnerabilities in a number of fields. These vulnerabilities raise concerns about the implementation of similar classifiers in anomaly-based NIDSs within SDNs. In this work, we investigate the viability of adversarial attacks against classifiers in this field. We implement an anomaly-based NIDS, *Neptune*, as a target platform that utilises a number of different machine learning classifiers and traffic flow features. We develop an adversarial test tool, *Hydra*, to evaluate the impact of adversarial evasion classifier attacks against *Neptune* with the goal of lowering the detection rate of malicious network traffic. The results demonstrate that with the perturbation of a few features, the detection accuracy of a specific SYN flood Distributed Denial of Service (DDoS) attack by *Neptune* decreases from 100% to 0% across a number of classifiers. Based on these results, recommendations are made as to how to increase the robustness of classifiers against the demonstrated attacks.**

***Index Terms*—Network Security, Software-Defined Networks, Intrusion Detection Systems, Machine Learning, Adversarial Attacks.**

I. Introduction

Networks are an essential part of modern society's infrastructure and are constantly under threat from malicious attacks, resulting in preventative measures being employed to provide security. New technologies changing the way networks are architected, particularly software-defined networks (SDN) and network function virtualization (NFV), have resulted in the implementation of new physical and software-based security measures specific to these architectures. Research into the deployment of network intrusion detection systems (NIDS) within SDNs has been promising [1] . The reason for this is that the centralised control plane within an SDN provides support for network-wide traffic monitoring. Machine learning (ML) is a technology that is becoming increasingly widespread and has enabled anomaly-based network intrusion detection based on the global network data available in the SDN. Despite this, recent research in areas such as image classification and malware detection have demonstrated that machine learning classifiers can be susceptible to adversarial attacks, negatively affecting their detection abilities.

Poisoning is defined as corruption of the training data of the classifier by an adversary, consequently reducing the likelihood of detection. *Overstimulaton*, as theorised in [2], bombards the classifier with benign data (network traffic), to overwhelm it, which results in misclassifications. This research focuses on the *evasion* classifier attack. An *evasion* attack allows attackers to evade detection by making small perturbations to observed features. These attacks are particularly important within a SDN setting, as, if successful, they would allow attackers to break or circumvent the NIDS. As ML classifiers become the default method of implementing NIDSs, the objective of this work is to bring to light the vulnerabilities within such security systems in SDNs.

For this research, *Neptune* is developed as a target NIDS platform. *Neptune* is used to determine the viability of *evasion* attacks against ML classifiers within a NIDS by performing classification on flow statistics between devices. This application implements traffic flow feature extraction and a selection of classification algorithms (Logistic Regression (LR), Random Forest (RF), Support Vector Machine (SVM), and K-Nearest Neighbours (KNN)).

An adversarial testing tool named *Hydra* has been developed to analyse the impact of *evasion* techniques on the accuracy of detecting SYN floods with *Neptune*. A SYN flood is a type of Distributed Denial of Service (DDoS) attack in which initial connection request (SYN) packets are sent to overwhelm all available ports on a target (victim) so that it is unavailable to legitimate traffic. Our attacks are created and performed with the same real-world capabilities of a malicious user i.e. with no knowledge of the classifier or system, operating from a node within a network.

The remainder of the paper is organised as follows. Section II provides a systematic review of relevant literature and identifies the gap in research that is addressed in this work. Section III introduces the NIDS, *Neptune*, with Section IV providing an insight into the performance of this system. Section V explains the adversarial attacks targeting Neptune, and Section VI introduces the *Hydra* test tool and presents an analysis of the test results. In Section VII, we provide recommendations to address the demonstrated NIDS weaknesses. Section VIII concludes the article.

II. Literature Review

Research into adversarial attacks against machine learning classifiers is not a new research area. Vulnerabilities have been documented in a number of different fields, most notably in ML malware classification, image classification and email

978-1-7281-4545-7/19/$31.00 ©2019 IEEE

spam detection [3]–[10]. An adversarial attack generator [11] has been developed to probe for vulnerabilities in image classifiers. With respect to IDSs, Corona et al. [2] present a survey on adversarial attacks, and potential defensive techniques have been discussed in [2], [4], [6]. In this work, we focus on vulnerabilities and defensive strategies in ML classifiers within SDN NIDSs.

ML has become prominent in the detection of android malware applications. Abaid et al. [8] demonstrated the vulnerability of these ML classifiers against adversarial *evasion* attacks. A systematic approach of generating an adversarial *evasion* attack with different levels of adversarial knowledge of the classifier is used to test the ability to avoid the detection of a malicious application. The results demonstrate that regardless of the level of attacker knowledge, there is potential to render linear classifiers and most non-linear classifiers redundant. It was proven that even a blind adversary was able to lower the detection rate of linear classifiers from 100% to 12%.

The introduction of ML image classification into safety-critical systems such as autonomous cars, and facial recognition for access control purposes has also sparked concern about potential vulnerabilities of these classifiers. Studies have proven the viability of *poisoning* and *evasion* attacks against image classification [3], [4]. In [4], Goodfellow et al. provide an example of how an adversary can *poison* a classifier resulting in an incorrect interpretation of road signs, thus leading the autonomous vehicle to disobey road laws. Similar to [8], Goodfellow et al. [4] prove that even with very limited knowledge, these attacks can still be successful.

Adversarial attacks on email spam filters have also proven successful across a range of ML classifiers, specifically using an *evasion* technique of inserting benign words into text [5].

These *evasion* and *poisoning* attacks are equally applicable in the domain of network security. An in-depth taxonomy on adversarial attacks against IDSs was published in 2013 [2]. In [2], Corona et al. present the broad scope of attacks against the three key elements of both an anomaly-based and misuse-based IDS – Measurement, Classification and Response. Furthermore, defences against these attacks were also proposed. The *overstimulation* attack noted in the introduction of this article is theorised within this work.

More recently, ML-NIDSs have become popular in network architectures such as SDNs. Lee et al. designed and tested a ML-based anomaly detector for SDNs named *Athena* [12], implementing a series of classifiers and reporting a DDoS detection rate of 99.23% using a K-Means-based algorithm. Given that this IDS is built upon the same classifiers used in prior work, we propose that it is exposed to the same set of vulnerabilities. Similarly, the ML-based IDSs presented in [13]–[18] could be vulnerable to adversarial attacks. A high flow rate of benign traffic against the ML classifier may allow malicious traffic to pass in the network without detection – an *overstimulation* attack. In the case of *poisoning*, with direct access to the training data, an adversary could input malicious traffic labelled as benign traffic thereby enabling malicious traffic to flow freely in the SDN. Even in the case of limited adversary knowledge or capability, perturbation of malicious network traffic features to represent benign traffic has strong potential to evade detection by the IDS.

In this work, we extend the existing research to investigate and quantify the vulnerability of a range of classifiers to adversarial attacks on NIDSs in SDNs. We demonstrate attack generalisation i.e. the capability of the attack to work across different classifiers.

III. Introduction to Neptune IDS Design

Neptune, an anomaly-based NIDS for SDNs, was developed to provide a target for adversarial attack experimentation. This system uses supervised learning on network flow statistics to train and classify live traffic. It was developed with the core goal of detecting DDoS attacks, most notably SYN floods to enable evaluation of adversarial *evasion* attacks based on attack detection accuracy.

In order to demonstrate the feasibility of adversarial attacks in state-of-the-art systems, *Neptune* was inspired by Athena [12]. Athena is a SDN-based anomaly detection system, providing a development framework that scales to larger networks. The application consists of 125 network features available to a developer to implement classification. The authors provide an example of one detection model for SYN floods using a K-Means clustering algorithm, highlighting 9 key features used for attack detection. With Athena achieving a 99.23% detection rate, the goal of *Neptune* is to use similar features in order to achieve a comparable detection accuracy.

A. Flow Statistic Collection

Neptune acquires live flow statistics from the network by listening to a dedicated traffic mirror host. The open-source SDN controller, Faucet [19], is used, enabling flexible flow rule implementations, to forward all network traffic to a specialised mirror host, as shown in Figure 1.

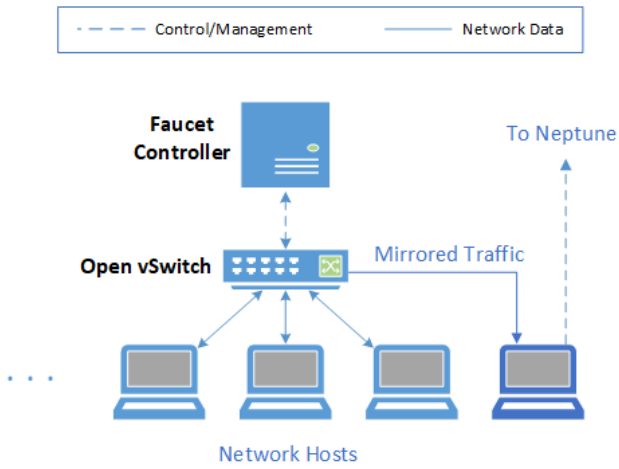

Fig. 1. Example topology highlighting the mirror host from which Neptune gathers flow statistics

Argus [20] is an open source layer 2+ auditing tool that *Neptune* commands to listen to the mirror host and record all

flow statistics being mirrored. These flow statistics are read by *Neptune* in time-based batches before being pre-processed based on unique flows between devices on the network to extract/construct relevant features and passed through the IDS. The choice of Argus was influenced by the quality of flow statistics obtained[1], in addition to the high level of performance with respect to speed and storage space. We acknowledge that the flow statistics mirroring technique would require further development to scale to larger networks with multiple switches.

B. ML Classifiers

A range of ML classifiers are implemented for intrusion detection, as shown in Table I. This enables comparisons to be made across different classifiers to evaluate individual classifier detection of SYN floods, and responses to adversarial attacks.

TABLE I
MACHINE LEARNING CLASSIFIERS AVAILABLE WITHIN *Neptune*

Category	Classifier
Classification	Random Forest (RF) Support Vector Machine (SVM) Logistic Regression (LR) K-Nearest Neighbour (KNN)

These models are trained on a number of key flow-based traffic statistics generated from Argus that are further processed by *Neptune*. The Argus features include the quantities of packets and bytes within specific flows, as well as state flags of different types of connections. For the classification of SYN floods, Neptune generates new features from the acquired flow statistics and discards those that are not required. The final features used for classification reflect the features identified as important for SYN flood DDoS detection in similar works e.g. [12], [13], [15], [18]. The features are listed in Table II.

TABLE II
SELECTION OF TRAFFIC FLOW FEATURES AVAILABLE FOR EACH FLOW

Feature Category	Feature
Packet header	eth_src,eth_dst,ip_proto,state_flags
Stateful	pkt_count,src_pkts,dst_pkts,bytes,src_bytes, dst_bytes,pkts_per_second, bytes_per_second,bytes_per_packet, packet_pair_ratio,pair_flow

The calculation of values for the stateful features listed in Table II is based on a dictionary lookup implementation to avoid polynomial time complexity, which would not scale well with increased traffic flow counts, which is a particular consideration for DDoS attacks. The full process flow of *Neptune* is illustrated in Figure 2.

[1]An initial implementation based on the SDN controller polling the switch for flow statistics demonstrated unreliable detection performance (as low as 40%) due to missing flow information.

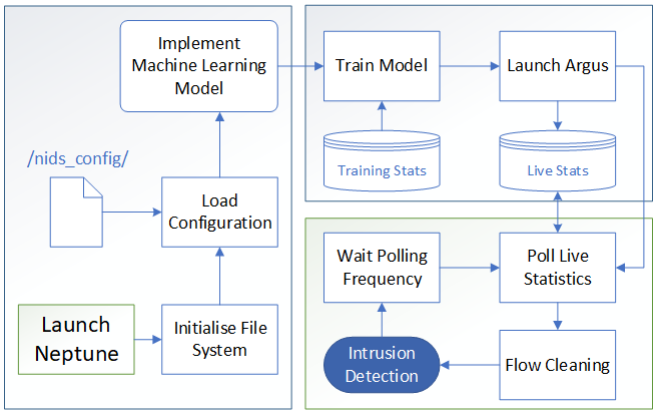

Fig. 2. Main process flow of *Neptune*

C. Traffic Dataset

While the KDD99 [21] dataset is a prominent dataset for the testing of IDSs, it has known limitations [22]. Newer and more relevant datasets are surfacing, including the CICIDS dataset [23] generated in 2017, which contains various types of malicious and benign traffic. In this work, we use the benign traffic from the CICIDS dataset. For malicious traffic, we use the DARPA SYN flood set [24] and a range of generated SYN floods. The DARPA flood exhibits a gradually increasing rate of SYN packets and the generated attacks consist of floods of varying speeds from 10 to 1,000,000 packets per second (pps), ensuring that the malicious dataset consisted of a wide range of intensities. The total dataset consists of 5 million packets. The ratio of benign to malicious traffic in the dataset is approximately 60/40, with 80/20 train/test splits taken from both sets to train and evaluate the system.

IV. NEPTUNE PERFORMANCE

The ML classifiers within *Neptune* were tuned using randomised search cross validation on the training dataset to find the optimum hyperparameters. Performing classification on the testing dataset, *Neptune* achieved the highest overall classification accuracy using the LR and RF algorithms, with an accuracy of 99.79%. These models generalised very well across the testing set. SVM achieved a similarly high accuracy of 99.59% with KNN less accurate at 97.45%. All classifiers obtained above 90% true positive rates for SYN flood detection. These results are illustrated in Table III and Figure 3. The metrics are formally defined as follows:

$$Acc = \frac{TP + TN}{TP + TN + FP + FN} \qquad Pr = \frac{TP}{TP + FP}$$

$$Re = \frac{TP}{TP + FN} \qquad F1 = 2 \cdot \frac{Pr \cdot Re}{Pr + Re}$$

where *Acc=Accuracy*, *Pr=Precision*, *Re=Recall*, *F1=F1 Score* *TP=True Positives*, *FP=False Positives*, *FN=False Negatives*.

In order to identify the influence of each feature on the classification process, we use recursive feature elimination (RFE) [25] to rank feature importance. The RFE results

identify that pair_flow ratio and other stateful features rank highest. This reflects the expected response based on the attack behaviour. For example, the pair_flow feature is the ratio of the number of unique flows travelling between a pair of network hosts. This feature would produce extreme values during a DDoS as the majority of the traffic will be unidirectional. The set of highest ranked features is consistent across the different classifiers, as illustrated in Table IV. (Note that KNN does not support feature importance.) These results suggest that perturbation of these features will result in the highest impact of an *evasion* adversarial attack.

TABLE III
NEPTUNE CLASSIFICATION RESULTS

Classifier	Accuracy	TP	FP	F1
RF	99.79	98.4	1.6	98.97
LR	99.79	98.4	1.6	99.18
SVM	99.59	97.1	2.9	98.35
KNN	97.45	92.2	7.8	89.86

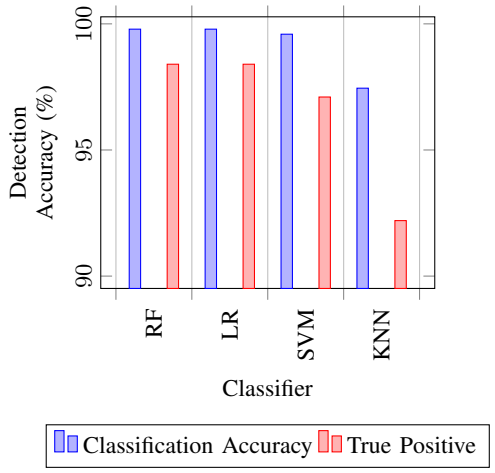

Fig. 3. Graph showing both the overall classification accuracy over the test dataset, and the true positive rate for SYN flood detection across each classifier

TABLE IV
FEATURE IMPORTANCE RANKING SHOWCASING TOP 5 FEATURES USING RFE FOR RF, LR, AND SVM CLASSIFIERS

Rank	RF Features	LR Features	SVM Features
1	pair_flow	bytes_per_pkt	bytes_per_pkt
2	bytes_per_pkt	state_flag	pair_flow
3	state_flag	pair_flow	state_flag
4	packet_pair_ratio	dst_bytes	dst_bytes
5	pkts_per_sec	ip_proto	pkts

The objective of this work is not to optimize the NIDS but to implement a suitable target platform for the adversarial attack analysis. The classification accuracy results presented are comparable with similar works [12], [13], [15], [16], [18] confirming that *Neptune* provides an appropriate baseline for adversarial testing. The feature importance ranking provides input to the adversarial attack analysis presented in Section V.

V. ADVERSARIAL ATTACKS

This research focuses on the adversarial *evasion* attack, with the goal of quantifying the impact it has against *Neptune*'s detection accuracy. The success of adversarial attacks depends on both the adversary's knowledge of the target system [8], [10] and their physical access to the network/system. It is understood that with more knowledge of the system e.g. knowledge of features used for classification, the task of crafting a successful attack is simplified. In order to produce meaningful results, the adversary's goals and knowledge must be defined.

A. Adversary Profile

The general adversary model proposed by [10] is used to disclose assumptions about the adversary in relation to their goal, knowledge and capability.

1) The Goal of the Adversary: The aim of an adversary with the intent to carry out a network attack such as a DDoS is to enable the attack to pass unnoticed. This inflicts the highest possible damage on resource availability. To counteract this, a NIDS must detect the attack with minimal false negatives, enabling further action to be taken based on the detection. A malicious user utilising adversarial techniques aims to subvert the classifier detection and increase false negatives in order to maximise the attack duration prior to detection.

2) The Knowledge of the Adversary: In this work, we make the assumption that an attacker has access to a single host within a network, with no direct access to the NIDS itself, or the classifiers used. This can be considered a blind attack. The adversary can make assumptions about the classifiers used as IDSs are well researched and generally use similar classifiers. [26] reports how attacks can generalise across a variety of classifiers. Therefore, with the correct perturbation of highly important features in DDoS detection, *evasion* should be apparent regardless of the ML algorithm.

As previously noted, the DDoS attack focused on within this research is a SYN flood. The characteristics of such an attack are known and documented, and influence the type of features used for detection. SYN floods have typically unidirectional communications with large packet counts and fast packet rates. As a result, the adversary will have the goal of crafting an *evasion* attack that perturbs these features to resemble those of benign traffic.

3) The Capability of the Adversary: An adversary can take advantage of altering a number of attack parameters in order to change the appearance of the attack to the NIDS. Without explicitly knowing the features used by the classifier, the adversary can still make assumptions as to what is typically measured for detection. The parameters that can be directly altered by an attacker when sending packets across a network are packet payload sizes, packet rates and packet counts. Moreover, the technique of forging additional traffic may influence flow features, disguising attacks.

B. Attack Profile

This paper proposes the development of *evasion* attacks by perturbing a combination of three fundamental SYN flood

characteristics. Considering the results from Table IV, the most important features can be perturbed by altering packet rates and payload sizes. The state flag cannot be altered as this would change the attack from a SYN flood. However, the pair_flow can be perturbed with forged bidirectional traffic. While using much smaller packet counts may evade detection, this would significantly weaken an attack. Therefore, packet counts will be maintained at appropriate levels.

Drawing from the knowledge and capabilities of the adversary as described in this section, three perturbation models are proposed to enable the *evasion* of a classifier:

1. Payload Size

Zhou et al. demonstrate in [27] that benign and attack SYN packet sizes can differ greatly with attack packets generally having much smaller payloads than their benign counterparts. This provides a distinctive characteristic to aid DDoS detection. This adversarial technique proposes the adjustment of the SYN packet payload size in order to appear more similar to benign traffic. Therefore, by increasing the packet payloads in an attack, detection confidence may be decreased.

2. Packet Rate

With the ability to control the packet rates of an attack, a low and slow technique may be used which is common in both HTTP-based DDoS attacks such as Slowloris, and TCP. This involves sending the packets of the SYN flood at a slower rate but still fast enough to have the desired effect of a DDoS. If packets are sent too slowly, a real-world target may be able to handle incomplete connections with timeouts as it is never overwhelmed completely, which would weaken the DDoS attack.

3. Bidirectional Traffic

A prominent sign of a DDoS attack is concentrated volumes of unidirectional traffic to one destination. An attacker can assume that an IDS uses high unidirectional packet counts to detect a SYN flood, similar to Athena's pair_flow feature. By forging traffic with the reversed source and destination to that of the attack packets, this perturbation has the aim of assuming the appearance of benign bidirectional communications. This introduces an increased number of attack flows into the network and therefore carries higher risk.

The adversarial attack surface available to an attacker performing a SYN flood against *Athena* and *Neptune* is not large, with a few important features and attack characteristics used for detection. Despite this, with the capability to perturb such important features, the potential for *evasion* is real.

VI. Results and Analysis

This section presents the analysis of the proposed perturbation attack models including their individual effect on classification confidence, and a comprehensive set of live detection results corresponding to varying combinations of perturbations.

A. Hydra Adversarial Test Tool

An adversarial evaluation tool, *Hydra* has been developed to provide a user with an interface and platform to test their ML-NIDS's resistance to adversarial attacks. This system performs network attacks within a SDN environment, applying different adversarial techniques to the attacks in order to subvert attack classification. *Hydra* launches its own emulated SDN (using Mininet) within which it performs attacks against a running NIDS providing live traffic flow classification. In the case of this research, the NIDS is *Neptune*. The test framework is illustrated in Figure 4. *Hydra* and *Neptune* will be made available open-source.

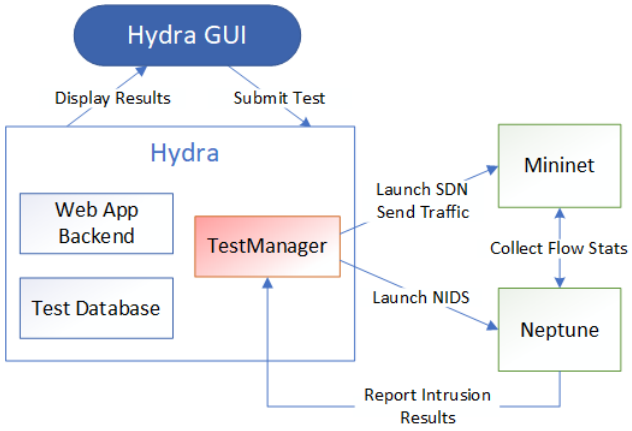

Fig. 4. Overview process flow of the full interaction between *Hydra* and *Neptune* for adversarial attack testing

The three main features an adversary has the potential to perturb were outlined in Section V. To investigate the impact of each individual perturbation, tests were carried out to observe the classification confidence of each algorithm detecting a perturbed SYN flood. The investigated feature was altered while maintaining all other features constant. Each test comprised of 20 attacks per perturbed value e.g. 20 attacks per payload size per classifier. The constant values are the default hping payload size (0 bytes) with approximately 650 pps for the packet rate and bidirectional packet rate. The confidence results for varying feature values are displayed in the graphs in Figure 5. A confidence level below 50% is a misclassification. Note that the SVM classifier confidences are not available due to the classifier design; SVM class assignments (benign/malicious) cannot be transformed into probabilities.

As illustrated in Figure 5, the LR classifier displays little change in confidence levels when one feature is perturbed at a time. On the other hand, the variation in RF and KNN results provide an insight into potential weaknesses in these classifiers. Payload size perturbation is shown to reduce the confidence of the RF classifier. Lower SYN flood packet rates decrease the classification confidence noticeably for both RF and KNN, while slightly decreasing the LR confidence. This indicates that the classifiers have generally been trained on DDoS attacks with a higher flow rate than those tested in Figure 5. Furthermore, as the bidirectional packet rate

increases, RF and KNN confidences decrease, indicating that matching the constant SYN flood packet rate may have the effect of disguising the attack flood.

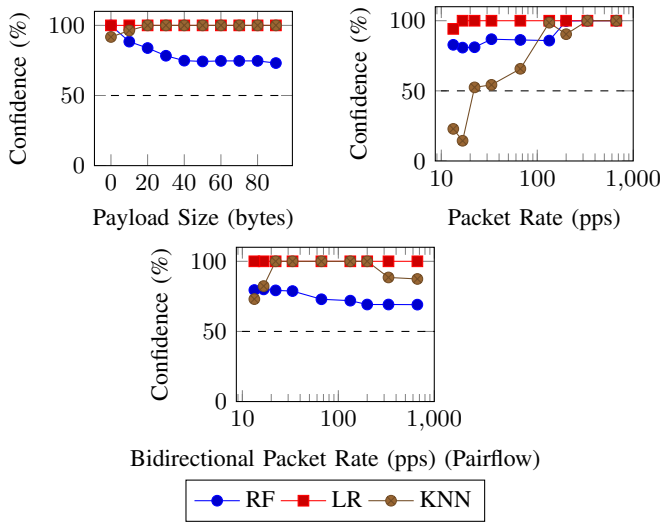

Fig. 5. Graphs showing SYN flood classification confidence with varied feature perturbation for RF, LR, and KNN. Below 50% confidence is a misclassification

TABLE V
PERTURBATION TEST SETTINGS (TABLE VI AND FIGURE 6)

	Payload	Rate	Pairflow
Constant Values	0 B	650 pps	650 pps
Perturbation Values	90 B	22 pps	22 pps
Payload+PairFlow	90 B	650 pps	22 pps
Rate+PairFlow	0 B	22 pps	22 pps
Payload+Rate+PairFlow	90 B	22 pps	22 pps

Based on these results, optimum perturbation values for each of the three features are determined. These values are detailed in Table V. An increased payload size perturbation value is based on the RF confidence trend. The packet rate is set as a compromise to ensure that the DDoS attack can still be effective. The bidirectional packet rate perturbation value is chosen to match the attack flow rate. Perturbing one feature, as shown in Figure 5, does not necessarily have the impact of *evasion*. However, combining perturbations should theoretically further reduce the classifier confidence. To quantify the impact of these perturbations, a base SYN flood was chosen that all classifiers could detect with 100% accuracy. For each classifier and combination of feature perturbations, 20 perturbed SYN floods were executed and an overall detection accuracy calculated by *Hydra* based on the detection results from *Neptune*. Similar to the initial confidence tests, any features that were not perturbed in a test were kept constant. The results are presented in Table VI and Figure 6.

The results in Table VI reflect the confidence results presented in Figure 5 demonstrating that by perturbing only one feature, the other two features are weighted high enough by all classifiers to outweigh the anomalous feature thus proving robust to *evasion*. Introducing the payload+rate perturbation combination does not affect the accuracy either. However, introducing pairflow perturbation, combined with another feature has a very strong influence on the success of the *evasion* attack. As identified in Table IV, the pair_flow feature is ranked as one of the highest across all classifiers. The *evasion* results for payload+pairflow, rate+pairflow, and the combination of all three feature perturbations are presented in Figure 6.

TABLE VI
UNSUCCESSFUL PERTURBATION TECHNIQUES ACROSS ALL CLASSIFIERS, RESULTING IN NO DECREASE IN DETECTION ACCURACY

Perturbation	Payload	Rate	Pairflow	Payload+Rate
Detection %	100	100	100	100

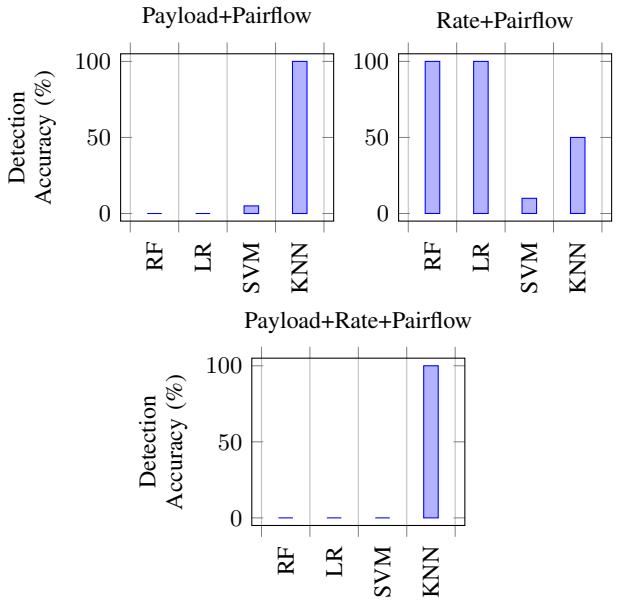

Fig. 6. Graphs for each combination of perturbed features, displaying the detection accuracy for each classifier

RF and LR both appear weak to pairflow perturbation coupled with payload perturbation, but are robust when the payload is not changed (in the case of rate+pairflow). This confirms the theory regarding distinguishable payload sizes between benign and malicious traffic [27]. The *Neptune* NIDS was trained with malicious DDoS packets with a lower payload size in relation to the CICIDS benign traffic. Therefore, increasing the payload size of the attack packets begins to resemble benign traffic. As presented in Section IV, SVM recorded one of the lowest true positive rates and the results presented in Figure 6 confirm that it is weak against pairflow combined with any other feature perturbation. KNN, however, shows a different trend to the other classifiers, with successful evasion only brought about by a rate+pairflow perturbation.

The trend of the RF, LR, and SVM classifiers prove that *evasion* attacks generalise across different algorithms provided that the correct, highly important features are perturbed. Furthermore, the KNN classifier appears to be the most robust classifier against the proposed *evasion* attacks although it achieved the lowest classification accuracy for SYN flood detection out of all the tested classifiers.

VII. Recommendations

The results presented in Section VI confirm that, similar to other domains, adversarial attacks are applicable to anomaly-based NIDSs in SDNs. It is, therefore, important to investigate preventative measures to limit the impact of such attacks.

In the development of ML-based NIDSs, it is common to undertake a feature selection/engineering phase to tune the algorithm to achieve the highest possible detection accuracy. We propose that this phase include adversarial robustness. For example, a multi-objective optimization formulation would target maximum detection accuracy *and* adversarial robustness of the ML-based NIDS. The *Hydra* tool presented in this work provides a platform to evaluate a classifier to ensure that it is adequately robust against adversarial attacks.

Secondly, despite the high detection accuracy of many classifiers (as evidenced here and in similar works [12]), it is clear that the classifiers rely heavily on features with perturbation potential suitable for blind knowledge attacks. In order to increase the effort required by the attacker, consideration should be made of introducing ensemble methods to the machine learning element such as combining the results of multiple classifiers in decision-making. Our future work will explore this along with further attack types.

VIII. Conclusion

The motivation for this work is the increasing deployment of ML-based NIDSs leveraging the global network visibility offered by SDNs. These solutions prioritise detection accuracy neglecting to consider the potential vulnerability of the ML algorithms to adversarial attacks. For an example use-case of a SYN Flood DDoS attack, we have demonstrated the ability to reduce the NIDS detection accuracy from 100% to 0% on multiple classifiers using *evasion* attacks. To support this research, we have developed the *Hydra* adversarial testing tool, a first of its kind in providing execution and evaluation of adversarial attacks against ML-based NIDSs in SDNs. *Neptune* was developed as the adversarial target and implements multiple classifiers demonstrating the concept of attack generalisation. KNN proved to be the most robust classifier against the adversarial attacks performed within this research, with only one combination of feature perturbations halving the detection accuracy from 100% to 50%. In contrast, RF, LR, and SVM were generally vulnerable to the same perturbations resulting in similar detection accuracy reductions. We propose that both research and industry adopt adversarial testing and integrate adversarial robustness as a performance measure in the development of ML-based NIDSs.

References

[1] I. Ahmad, S. Namal, M. Ylianttila, and A. Gurtov, "Security in Software Defined Networks: A Survey," *IEEE Communications Surveys Tutorials*, vol. 17, no. 4, pp. 2317–2346, Fourthquarter 2015.

[2] I. Corona, G. Giacinto, and F. Roli, "Adversarial Attacks against Intrusion Detection Systems: Taxonomy, Solutions and Open Issues," *Information Sciences*, vol. 239, pp. 201 – 225, 2013.

[3] H. Kwon, Y. Kim, K. Park, H. Yoon, and C. Choi, "Multi-Targeted Adversarial Example in Evasion Attack on Deep Neural Network," *IEEE Access*, vol. 6, pp. 46 084–46 096, 2018.

[4] I. Goodfellow, P. McDaniel, and N. Papernot, "Making Machine Learning Robust Against Adversarial Inputs," *Commun. ACM*, vol. 61, no. 7, pp. 56–66, Jun. 2018.

[5] K. Pawar and M. Patil, "Pattern classification under attack on spam filtering," in *2015 IEEE International Conference on Research in Computational Intelligence and Communication Networks (ICRCICN)*, Nov 2015, pp. 197–201.

[6] Q. Liu, P. Li, W. Zhao, W. Cai, S. Yu, and V. C. M. Leung, "A Survey on Security Threats and Defensive Techniques of Machine Learning: A Data Driven View," *IEEE Access*, vol. 6, pp. 12 103–12 117, 2018.

[7] Z. Wang, "Deep Learning-Based Intrusion Detection With Adversaries," *IEEE Access*, vol. 6, pp. 38 367–38 384, 2018.

[8] Z. Abaid, M. A. Kaafar, and S. Jha, "Quantifying the impact of adversarial evasion attacks on machine learning based android malware classifiers," in *2017 IEEE 16th International Symposium on Network Computing and Applications (NCA)*, Oct 2017, pp. 1–10.

[9] X. Liu, Y. Lin, H. Li, and J. Zhang, "Adversarial Examples: Attacks on Machine Learning-based Malware Visualization Detection Methods," *CoRR*, vol. abs/1808.01546, 2018.

[10] B. Biggio *et al.*, "Evasion attacks against machine learning at test time," in *Joint European conference on machine learning and knowledge discovery in databases*. Springer, 2013, pp. 387–402.

[11] I. J. Goodfellow, N. Papernot, and P. D. McDaniel, "Cleverhans v2.1.0: An Adversarial Machine Learning Library," *CoRR*, vol. abs/1610.00768v6, 2018.

[12] S. Lee, J. Kim, S. Shin, P. Porras, and V. Yegneswaran, "Athena: A Framework for Scalable Anomaly Detection in Software-Defined Networks," in *2017 47th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, June 2017, pp. 249–260.

[13] A. Abubakar and B. Pranggono, "Machine learning based intrusion detection system for software defined networks," in *2017 Seventh International Conference on Emerging Security Technologies (EST)*, Sept 2017, pp. 138–143.

[14] N. Sultana *et al.*, "Survey on SDN based network intrusion detection system using machine learning approaches," *Peer-to-Peer Networking and Applications*, pp. 1–9, 01 2018.

[15] R. Braga, E. Mota, and A. Passito, "Lightweight DDoS flooding attack detection using NOX/OpenFlow," in *IEEE Local Computer Network Conference*, Oct 2010, pp. 408–415.

[16] S. A. Mehdi, J. Khalid, and S. A. Khayam, "Revisiting Traffic Anomaly Detection Using Software Defined Networking," in *Proceedings of the 14th Int. Conf. on Recent Advances in Intrusion Detection*, ser. RAID'11. Springer-Verlag, 2011, pp. 161–180.

[17] T. A. Tang, L. Mhamdi, D. McLernon, S. Zaidi, and M. Ghogho, "Deep learning approach for Network Intrusion Detection in Software Defined Networking," in *2016 Int. Conf. on Wireless Networks and Mobile Communications (WINCOM)*, Oct 2016, pp. 258–263.

[18] Q. Niyaz, W. Sun, and A. Y. Javaid, "A Deep Learning Based DDoS Detection System in Software-Defined Networking (SDN)," *CoRR*, vol. abs/1611.07400, 2016.

[19] faucetsdn. (2019) Faucet SDN Controller. [Online]. Available: https://faucet.nz/

[20] QoSient. (2019) Argus Network Activity. [Online]. Available: https://qosient.com/argus/index.shtml

[21] I. University of California. (1999) KDD Cup 1999 Data. [Online]. Available: http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html

[22] J. McHugh, "Testing intrusion detection systems: a critique of the 1998 and 1999 darpa intrusion detection system evaluations as performed by lincoln laboratory," *ACM Transactions on Information and System Security (TISSEC)*, vol. 3, no. 4, pp. 262–294, 2000.

[23] I. Sharafaldin, A. Habibi Lashkari, and A. Ghorbani, "Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization," in *4th International Conference on Information Systems Security and Privacy (ICISSP)*, 01 2018, pp. 108–116.

[24] C. S. U. Manaf Gharaibeh. (2009) DARPA 2009 Intrusion Detection Dataset. [Online]. Available: http://www.darpa2009.netsec.colostate.edu

[25] Scikit-Learn. (2017) Recursive Feature Elimination (RFE). [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.RFE.html

[26] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and Harnessing Adversarial Examples," *arXiv preprint arXiv:1412.6572*, 2014.

[27] L. Zhou, M. Liao, C. Yuan, and H. Zhang, "Low-rate DDoS attack detection using expectation of packet size," *Security and Communication Networks*, vol. 2017, 2017.