Poster: Speeding Up Network Intrusion Detection

João Romeiras Amado
Instituto Superior Técnico
Universidade de Lisboa
Lisbon, Portugal
joao.amado@tecnico.ulisboa.pt

Salvatore Signorello
Faculdade de Ciências
Universidade de Lisboa
Lisbon, Portugal
ssignorello@ciencias.ulisboa.pt

Miguel Correia
Instituto Superior Técnico
Universidade de Lisboa
Lisbon, Portugal
miguel.p.correia@tecnico.ulisboa.pt

Fernando Ramos
Instituto Superior Técnico
Universidade de Lisboa
Lisbon, Portugal
fvramos@tecnico.ulisboa.pt

***Abstract*—Modern network data planes have enabled new measurement approaches, including efficient sketch-based techniques with provable trade-offs between memory and accuracy, directly in the data plane, at line rate. We thus ask the question: can one leverage this richer measurement plane to improve network intrusion detection? Our answer is SPID, a push-based, feature-rich network monitoring approach to assist learning-based attack detection. SPID switches run a diverse set of measurement primitives and proactively push measurements to the monitoring system when relevant changes occur. Network measurements are then fed as input features to a classifier based on unsupervised learning to detect ongoing attacks, as they occur. In consequence, SPID aims to reduce attack detection time, when comparing to existing solutions present in large scale networks.**

***Index Terms*—NIDS, programmable data planes, sketches**

I. Introduction

To identify malicious network activities, operators resort to Network Intrusion Detection Systems (NIDS) that capture and analyze packet flows to detect attacks. Existing NIDS, however, fall short in responding to an increasingly sophisticated array of attacks, for at least three reasons.

The first problem is that the most common NIDS in operation are signature-based, a type that is simple to implement and effective in identifying known attacks. This class of NIDS is however incapable of detecting unknown attacks, given the lack of knowledge of their signatures. Additionally, the ever-growing sophistication of network attacks makes it increasingly harder to devise effective signatures for detection. By contrast, anomaly-based techniques model normal network behavior, and identify deviations as potential anomalies. They are based on the hypothesis that the attacker's behavior differs from regular traffic patterns, and are thus able to detect new and evolving attacks. In particular, anomaly-based systems based on learning and inference approaches, the approach we follow, have been shown to improve detection accuracy in several environments [1].

The second problem is the speed of detection. As the network monitoring tools traditionally employed in NIDS are based on pull-based approaches, they are unable to capture transient or short-lived events, often taking several minutes to detect attacks and therefore failing to respond. The ideal NIDS should observe changes to traffic and respond promptly to attacks. Unfortunately, the monitoring tools employed for statistics collection in NIDS (e.g., NetFlow) produce highly aggregated statistics at relatively coarse time-scales. To address this problem, several authors have argued for a monitoring paradigm shift, from passive to active [2]. An active monitoring infrastructure should react, in an automated manner, to special events. We follow this approach, shifting from pull-based monitoring, where the controller periodically polls switches for updates, towards a solution where switches proactively push statistics when a particular data plane event is triggered (e.g., a counter reaches a predefined threshold).

The third problem of existing NIDS is the quality of the measurement data available as input to the system. The measurement primitives used are typically based on sampled statistics (e.g., sampled NetFlow records), for scalability reasons. As such, they only provide low accuracy, coarse-grained views of the network, reducing the effectiveness of intrusion detection. We leverage sketch-based algorithms [3] running at line rate in switch hardware to complement traditional packet/byte counters, with the goal of increasing the variety of features available as input for the detection system.

To assist network operators in overcoming these three problems we present SPID, our Switch-Powered Intrusion Detection framework. SPID leverages modern switch data planes [4] to improve the detection rate of zero-day attacks, including short-lived attacks that require fast response times.

II. System Architecture

Fig. 1 presents the architecture of SPID, which follows a bottom-up approach to intrusion detection, built upon the typical SDN control and data plane separation.

In the data plane, high-speed switching ASICs compute and store a diverse set of flow statistics and multiple sketch-based counters, implemented through a P4 program [5]. SPID can be configured to monitor specific packet features, from single packet fields to combinations of multiple fields. The specific set of active counters can be configured by the operator at runtime. Each switch's available memory is dynamically allocated between all active counters. SPID switches can be used as regular switches that perform this form of enhanced monitoring alongside regular forwarding. Besides simple counting, each switch performs traffic change detection over short, configurable time windows. Relevant traffic changes are used as a trigger to send measurement data to the control plane. This proactive behavior avoids the coarse time-scales of traditional polling-based monitoring approaches coordinated by a network

978-1-7281-6992-7/20/$31.00 ©2020 IEEE

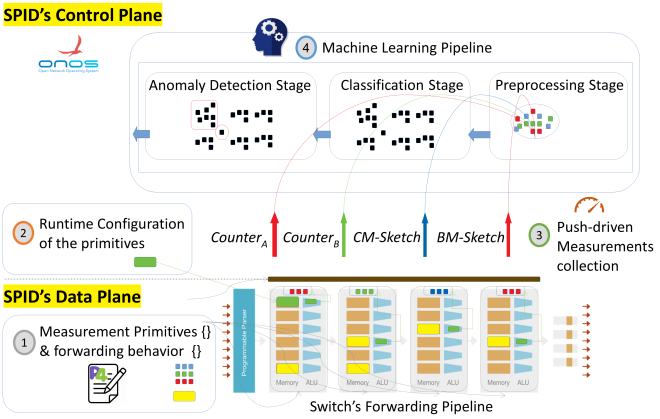

Fig. 1. SPID's architecture and operation (with a single switch for simplicity). (1) the packet processing pipeline is programmed with a large set of measurement primitives; (2) a set of primitives is enabled by a specific runtime configuration; (3) push-driven measurements collection based on traffic changes in the data plane; (4) analysis of the collected measurements.

monitor, enabling a faster attack detection. In addition, it relieves the control plane of the burden of performing periodic polling, which leads to scalability issues in large networks.

In the control plane, at every switch notification the collected measurements are fed to a machine learning (ML)-based anomaly detection pipeline. A preprocessing stage is executed for data normalization. Afterwards, a classification stage is performed over the resulting features through a clustering algorithm, aiming to aggregate related flows according to their characteristics. Finally, the anomaly detection stage analyzes the obtained results and identifies potential anomalies, represented as new clusters or as outliers. Depending on the operator's intent, suspicious traffic can then be dropped or steered to a defense system for further inspection.

III. Evaluation

The goal of the preliminary evaluation of our prototype is to obtain initial empirical evidence of the three main arguments for SPID. This evaluation, based on real traffic traces with labelled attacks [6], used the P4 software reference switch on a virtualized environment.

Detection of unknown attacks. We tested the ability of SPID to detect four attacks of which the system had no previous knowledge: TCP Syn Flood, UDP Fraggle, Ping-of-Death, Smurf. We compare SPID with two other approaches, 1Sk and Smp, which respectively use only one Count-Min sketch and only packet/byte counters for sampled data. Across our tests, SPID always detects more attacks (higher True Positives percentage) than the other baseline NIDS. Regarding TCP SYN flood, Smp is not even able to detect the attack. Although our preliminary results also show that SPID does not yet have the accuracy and precision required by a NIDS, the main take-away is that a combination of multiple measurement primitives is always much better than any single metric.

Stream-based over sample-based. In our simulation of a sampling-based NIDS (Smp) we set the sampling rate at 1:500. Across all tested attacks, the detection rate with Smp is considerably lower (True Positive rate 20 to 50 percentage points lower) than the detection with SPID. These results confirm previous research that reported on the limitations of sampling-based approaches for attack detection.

TABLE I
SPID detection results.

		TP	FP	TN	FN	Acc.	Precision	Recall
TCP	**SPID**	40	66	99.5	0.4	67.8	37.7	99.0
	1Sk	30	69.7	99.4	0.4	64.9	30.1	98.6
	Smp	0	100	95.1	4.9	47.5	0	0
UDP	**SPID**	50	18.6	99.6	0.4	88.8	72.9	99.2
	1Sk	40	54.7	99.5	0.5	71.7	42.3	98.8
	Smp	30	83.5	98.6	1.4	60.2	26.4	95.4
PoD	**SPID**	93.3	44.8	99.9	0.1	81.1	67.5	99.9
	1Sk	30	94.2	99.4	0.6	57.7	24.2	98.2
	Smp	46.7	68.4	97.2	2.8	66.9	40.6	94.4
Smurf	**SPID**	90	52	99.9	0.1	78.4	63.3	99.9
	1Sk	60	83.5	99.6	0.4	65.5	41.8	99.3
	Smp	60	84	98.5	1.5	64.9	41.6	87.5

TP/FP: True/False Positives; TN/FN: True/False Negatives; Acc.: Accuracy

Detection time. We performed experiments to showcase that a push-driven approach offers a fast detection of ongoing attacks. While the detection time of a sampling-based NIDS is inherently constrained by the sampling frequency, SPID detects anomalous patterns as soon as they happen in the data plane. The average triggering time, specifically, the difference between the time the attack actually starts and the time the switch pushes the related measurements to the control plane, is nearly always on the order of the hundreds of milliseconds.

IV. Conclusion

We presented SPID, a framework that leverages programmable data planes to speed up the detection of network attacks. On the data plane, we are currently exploring the integration of other sketching algorithms to improve the measurement plane within the constraints of programmable networking hardware. On the control plane, we are investigating other anomaly detection approaches to improve the precision of SPID to the level required by NIDS environments.

References

[1] A. L. Buczak and E. Guven, "A survey of data mining and machine learning methods for cyber security intrusion detection," IEEE Communications Surveys & Tutorials, vol. 18, no. 2, pp. 1153–1176, 2016.

[2] M. Moshref, M. Yu, R. Govindan, and A. Vahdat, "Trumpet: Timely and precise triggers in data centers," in Proceedings of the 2016 ACM SIGCOMM Conference, 2016.

[3] M. Yu, L. Jose, and R. Miao, "Software defined traffic measurement with opensketch." in NSDI, vol. 13, 2013, pp. 29–42.

[4] P. Bosshart, G. Gibb, H.-S. Kim, G. Varghese, N. McKeown, M. Izzard, F. Mujica, and M. Horowitz, "Forwarding metamorphosis: Fast programmable match-action processing in hardware for SDN," in ACM SIGCOMM Computer Communication Review, vol. 43, no. 4, 2013, pp. 99–110.

[5] "P4-16 Language Specification" https://p4.org/p4-spec/docs/P4-16-v1.2.1.html, accessed: 2020-07-23.

[6] A. Hamza, H. H. Gharakheili, T. A. Benson, and V. Sivaraman, "Detecting volumetric attacks on lot devices via sdn-based monitoring of mud activity," in Proceedings of the 2019 ACM Symposium on SDN Research, 2019, pp. 36–48.