Extracting Network Flow Features Efficiently in P4 Data plane

Sankalp Mittal
cs21mtech12010@iith.ac.in

ABSTRACT

There is an increase in unmanaged IoT devices which in turn increases the attack surfaces. The IoT devices are generally from different manufacturers who do not pay much heed to the security features of these devices. As such, the devices are vulnerable to divert attack traffic on vendor server, thereby compromising it. Therefore, there is a need of security in IoT devices and the urge to detect the attack traffic as quickly and "cheaply" as possible. In this paper, we try to extract flow features of attack traffic in P4 dataplane itself instead of control plane. Further, after extracting the flow features, we try to implement trained Machine Learning Algorithms in control plane to classify various attack traffic. Extracting flow features in the data plane instead of control plane will reduce the data collection overhead on the SDN controller and also the processing time to extract flow features.

1 INTRODUCTION

In this work, we will be leveraging state of the art Machine Learning Algorithms that classify the traffic as benign or attack based on the extracted flow features. This need to classify the traffic comes with the urge of recognizing various attacks implemented on IoT devices. We first introduce how previous works have deployed trained Machine Learning Algorithms in the control plane.

1.1 Current Work: Flow-Meter Stats

The way the previous models function is that the packets incident on a P4 data plane are sent to the control plane. Using information of the packet headers, the traffic flows and the features are extracted in the control plane (for instance, using CICFlowMeter or NFStream). The packets are incident on the ingress port of P4 data plane switch and from there, the packets are sent to CIC Flow Meter which is running in the centralized SDN controller (control plane). This flow meter via mirroring (and tapping) maps the incoming packets to the flows and extracts the flow based features (depending on header field values corresponding to each flow). The flow features are then collected and sent to some state of the art trained Machine Learning model that also resides in the control plane. The classification is then done dynamically by the trained model as it collects the features. The idea is that whenever packets are incident on an ingress port of a P4 data plane switch and are forwarded out of the egress port, the outgoing packets are mirrored/tapped and their copy is sent to some agent residing in the same network. We can either run the training algorithm in the same agent or we can also run it on a different device that will behave as an Intrusion Detection System (IDS). One usecase is that, we can implement the traffic flow classification either in the edge router of a subnet or the packets can be first mirrored/tapped and sent to firewall/IDS.

On the flip side, implementing trained machine learning algorithm on firewall besides extracting flow features could be very expensive. Say we have 3.2 Tbps incoming traffic and each firewall is able to typically manage no more than 10gbps worth of traffic. Then we need to split the traffic between 32 firewalls. If each firewall costs 10 lakhs, 30 firewalls would cost 1.5 crore which is way too expensive. All this key issue is lack of scalability. Since we are sending packets themselves from data plane to the control plane, the information exchange is very high and this incurs a lot of processing overhead. We won't be able to scale the current system for large number of IoT devices (huge traffic). Say if 100 packets are to be sent from data plane to the control plane, that's an exchange of 6000B of information assuming each packet contains 60B of header fields. Therefore, the current approach is just not scalable to large number of IoT devices or huge amount of traffic.

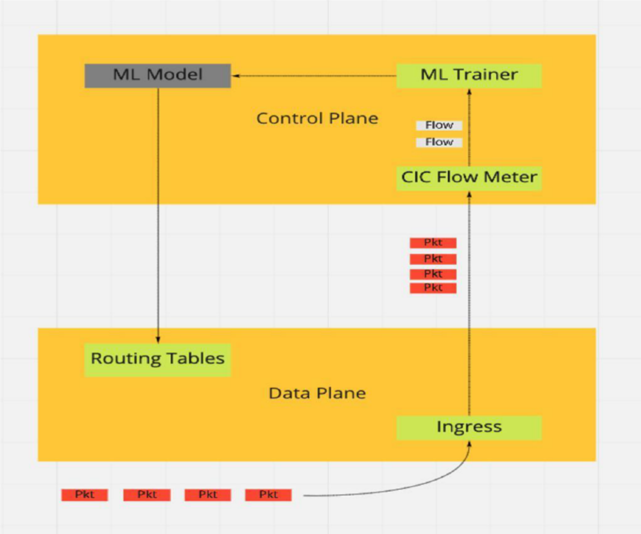

Figure 1: Flow Meter Stats Approach

1.2 Key Idea: P4 Stats

Due to the presence of high speed switching fabric, P4 data plane switches are much faster than a control plane can function. This is the key aspect of our rationale, to implement all the packets to flow-feature pre-processing to data plane instead of a control plane.

Our key rationale is to collect the flow features in the P4 data plane itself while not burdening control plane. This significantly reduces the data collection overhead and also the pre-processing time to extract features. We are basically collecting and maintaining per-flow record and per-flow aggregated in the P4 data plane. The data plane needs to forward the aggregated flow features and records periodically to the control plane. The control plane will feed these features to the trained model to classify the flows. This very approach not only significantly reduces the processing overhead at control plane besides bringing down the processing delay, it also paves way for fast classification of incoming traffic. This makes the whole model deployment scalable to large number of IoT devices unlike

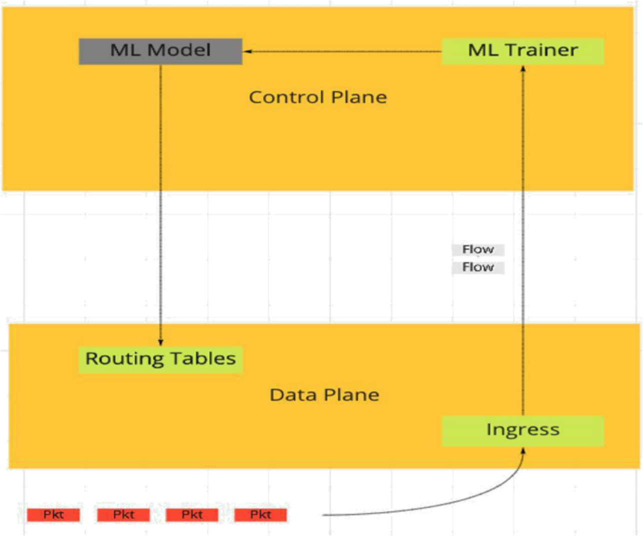

Figure 2: P4 Flow Stats Approach

the current approach. Let us consider that 100 packets constitute 5 flows which are being sent to the control plane. Then 5-tuples need to be sent since packets are classified into flows based on this information. Plus, we also need to send some key aggregate features, say k of them. So, total data to be sent becomes $56B + 8k$ Bytes. Even if we assume $k = 83$, we still need to send only 3600B as opposed to 6000B. We can reduce k to 11 by collecting only important discriminating features and thus we will be sending only 720B.

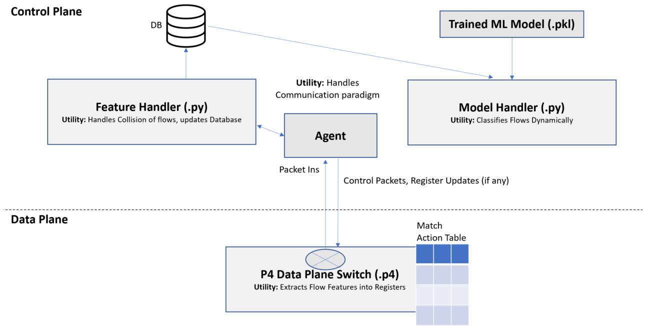
Control Plane
DB
Trained ML Model (.pkl)
Utility: Handles Communication paradigm
Feature Handler (.py)
Utility: Handles Collision of flows, updates Database
Agent
Model Handler (.py)
Utility: Classifies Flows Dynamically
Packet Ins
Control Packets, Register Updates (if any)
Data Plane
Match Action Table
P4 Data Plane Switch (.p4)
Utility: Extracts Flow Features into Registers

Figure 3: System Design

2 NEXT STEPS

– Classify flows using trained ML models (normal vs malicious).

– Push Machine Learning model driven decisions to match-action tables in the data plane.

– Develop P4 program that allows dynamic rule installation (deny, forward, ratelimit).

- Design a system that does feature collection and classification entirely in the data plane. Challenge: A single device may not have enough number of stages and memory.

– Split P4 program across multiple devices: A combination of smartNICs, switches, GPUs, and CPUs.

- Continuous learning: Framing this problem as a unsupervised anomaly detection and training the model periodically.

REFERENCES

[1] Detecting Volumetric Attacks on IoT Devices via SDN-Based Monitoring of MUD Activity.
[2] CNN-Based Network Intrusion Detection against Denial-of-Service Attacks.
[3] Features Dimensionality Reduction Approaches for Machine Learning Based Network Intrusion Detection.
[4] pForest: In-Network Inference with Random Forests.
[5] SwitchTree: In-network Computing and Traffic Analyses with Random Forests.
[6] Do Switches Dream of Machine Learning?: Toward In-Network Classification