Diogo Barradas **Nuno Santos** **Luís Rodrig**

Fernando Ramos **André Madeira**

INESC-ID, Instituto Superior Técnico, Universidade de Lisboa

Salvatore Signorello

LASIGE, Faculdade de Ciências, Universidade de Lisboa

FlowLens: Enabling Efficient Traffic Analysis for Security Applications Using Programmable Switches

Performance Breakthroughs with Programmable Switches

- **Line-speed packet processing at Tbps**
- **Fully programmable in the P4 language**
- **Recent focus of HW manufacturers**

STRATAXGS
TOMAHAWK

New opportunities for network security

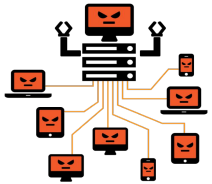
Securing High-Speed Networks

- **Programmable switches are used to:**
  - Obfuscate Network Topologies [NetHide, SEC'18]
  - Filter spoofed IP traffic [NetHCF, ICNP'19]
  - Mitigate DDoS attacks [Poseidon, NDSS'20]
  - Thwart network covert channels [NetWarden, SEC'20]

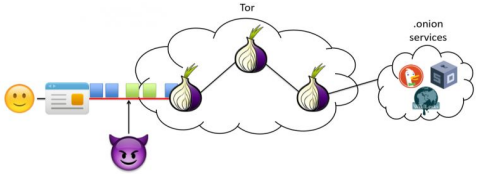
Line-speed packet processing
Highly efficient

Fine-tuned for specific application domain

There are Other Prominent ML-based Security Applications

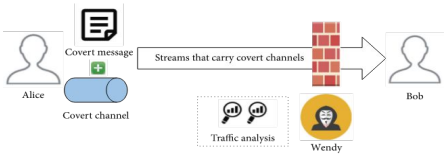


Botnet Detection



Website Fingerprinting

IoT Behavioral Analysis



Detection of Covert Channels

Statistical Traffic Analysis

Packet lengths

Packets inter-arrival time

\+ ML-based classifier

Generic approach towards detecting multiple attacks

Collecting Packet Distributions in Programmable Switches is Hard

- **Stateful memory is severely limited**
  - ~100 MB SRAM
  - No memory for storing many flows
- **Packets must be processed at line speed (< a few tens of ns)**
  - Limited number of operations
  - Reduced [domain-specific] instruction set

It does not seem feasible to obtain packet distributions in programmable switches at scale

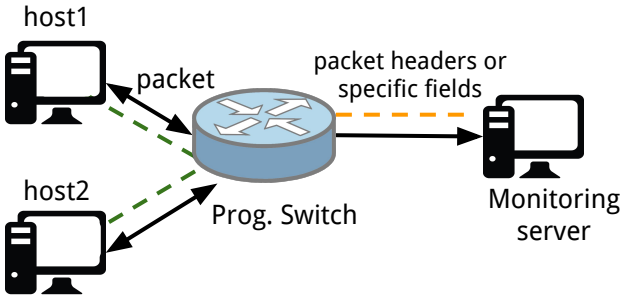
Research Question

- **Can we collect packet distributions within programmable switches?**

Efficient

Generic

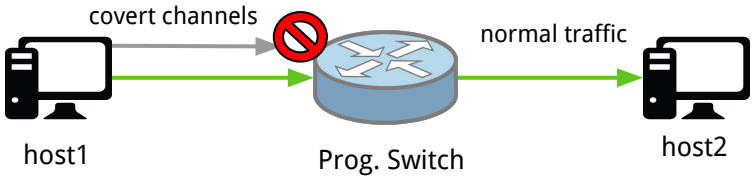
Solutions for Collecting Packet Distributions Have a Few Drawbacks



*Flow, USENIX ATC'18

Generic

Large Bandwidth Costs



Netwarden, USENIX SEC'20

Efficient

Application-tailored

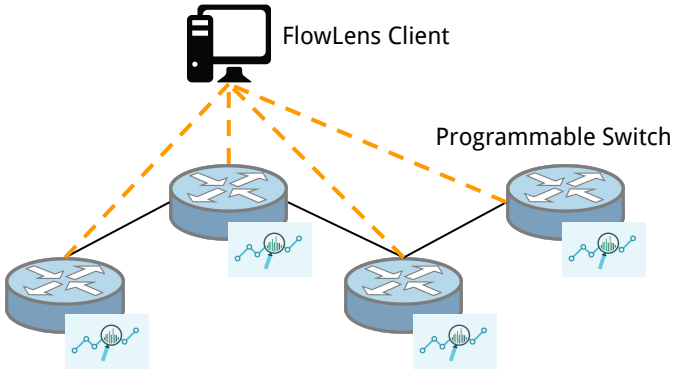
Contributions

FlowLens: a flow classification system for generic ML-based security applications

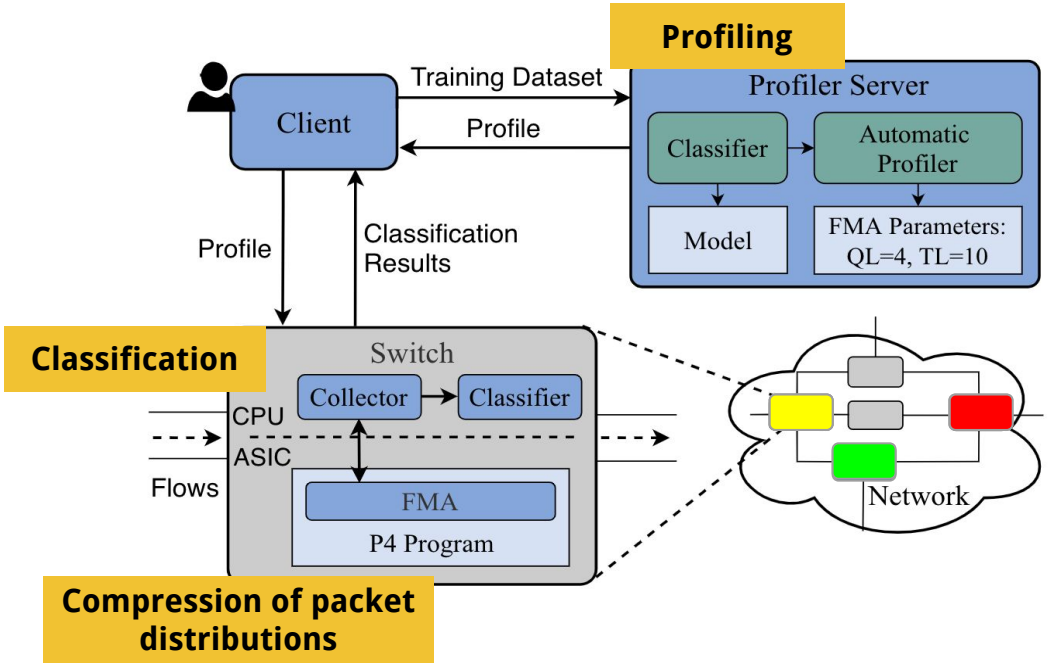
- **Flow markers:** Compact representation of packet distributions in prog. switches
- **Flow marker accumulator:** Implementation of flow marker collection in switching hardware
- **Automatic profiling:** Application-tailored configuration of flow markers
- **Evaluation:** Tested in 3 different security tasks

Efficient

Generic



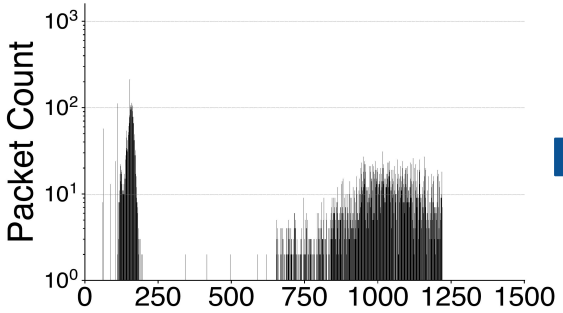
FlowLens Architecture



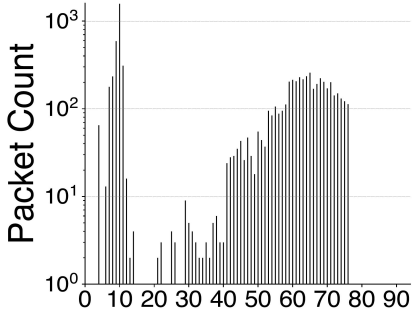
- **Distributed Deployment**
  - Scale # of measured flows
  - Ensure network visibility
- **Coordinated Operation**
  - Multiple ML applications

How can we Compress Packet Distributions Efficiently?

- **Produce flow markers with two operators**
  - **Quantization**
  - **Truncation**

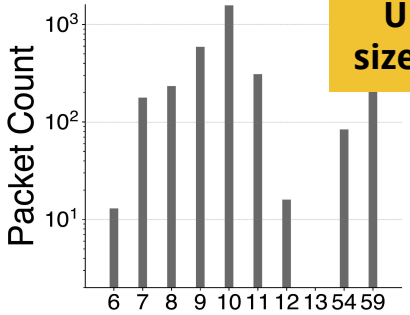


Raw packet size distribution



Quantized distribution

QL = 4 (2^4x compression)



Truncated distribution

Top-10 bins

Up to 150x size reduction

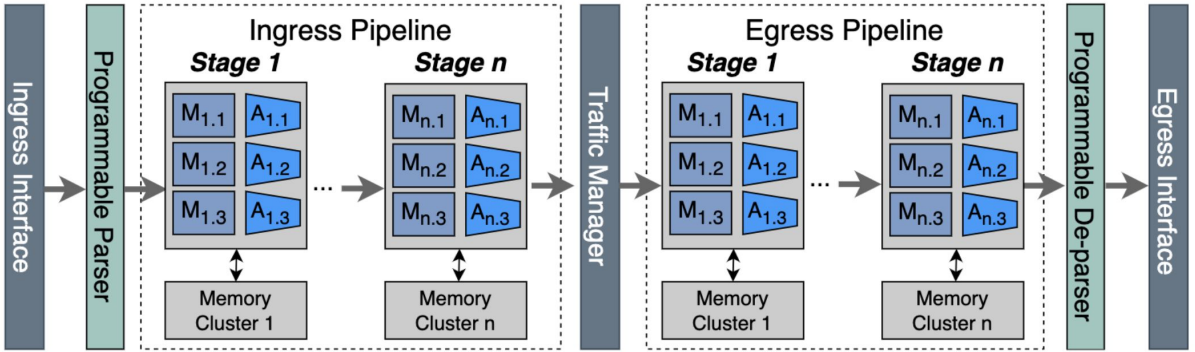
How to Automatically Choose Quant/Trunc Parameters?

- **Large configuration space**
  - **Quantization x Truncation**
- **Leverage Bayesian Optimization**
- **Automatic Profiler with three criteria**
  - Smaller marker for target accuracy
  - Best accuracy given a size constraint
  - Compromise between marker size and accuracy

Saves many hours of testing sub-optimal configurations

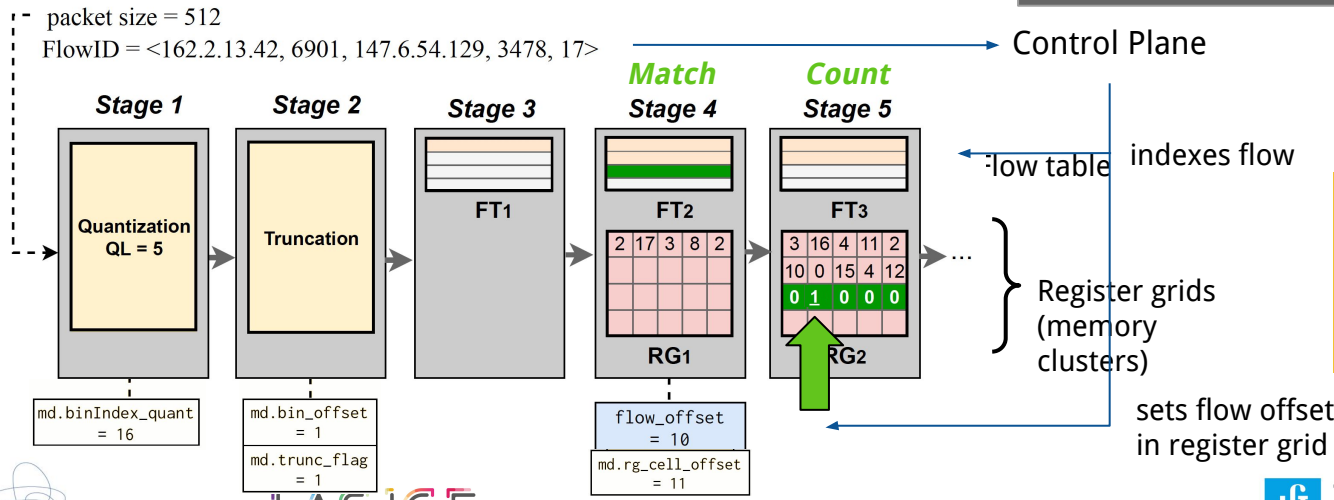
How are Flow Markers Collected in the Switch?

- **Programmable packet parsing**
- **Leverage match-action tables**
  - Arranged in stages
  - Match some packet field
  - Change packet headers or metadata



How are Flow Markers Collected in the Switch?

- **Flow Marker Accumulator**
  - Simple per-stage operations
  - Offload complex operations (control plane)



Compress packet distributions at line-speed

Evaluation

- **Scalability in three use cases**
  - Covert Channel Detection
  - Website Fingerprinting
  - Botnet Detection
- **Performance of FlowLens's profiler**
- **Resources consumption**
  - CPU usage (control plane)
  - ASIC usage (data plane)

Scalability Gains Overview

- **Scalability in three use cases**
  - Covert Channel Detection
  - Website Fingerprinting
  - Botnet Detection

Use Case	Scaling (# flows)	Performance Loss
Covert Channels	150x	-3% accuracy
Website Fingerprinting	32x	-2% accuracy
Botnet Detection	34x	-3% recall -2% precision

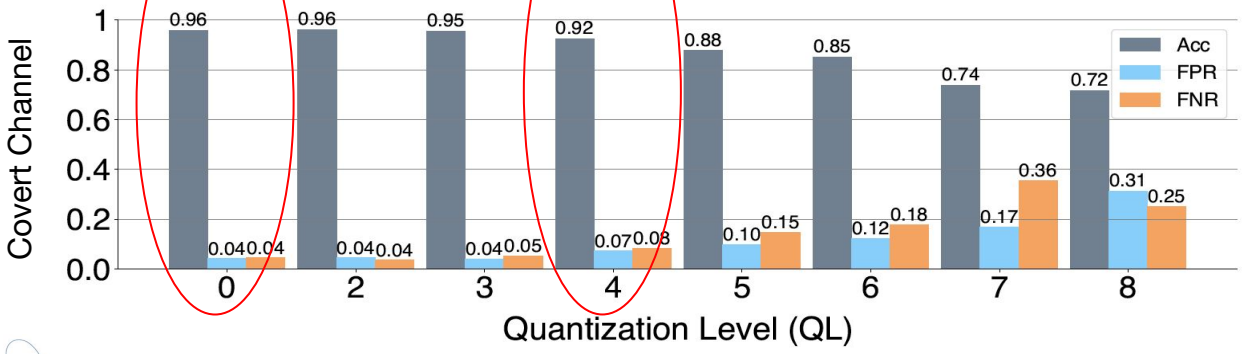
Check the paper for our comprehensive evaluation!

FlowLens Scales the Amount of Inspected Flows and Retains Acc.

- **Covert Channel Detection** [Barradas et al.]
  - Legitimate / Modified Skype flows
  - Packet lengths + XGBoost

16x increase in measured flows

# Bins	1500	375	188	94	47	24	12	6
Memory (B)	3000	750	376	188	94	48	24	12



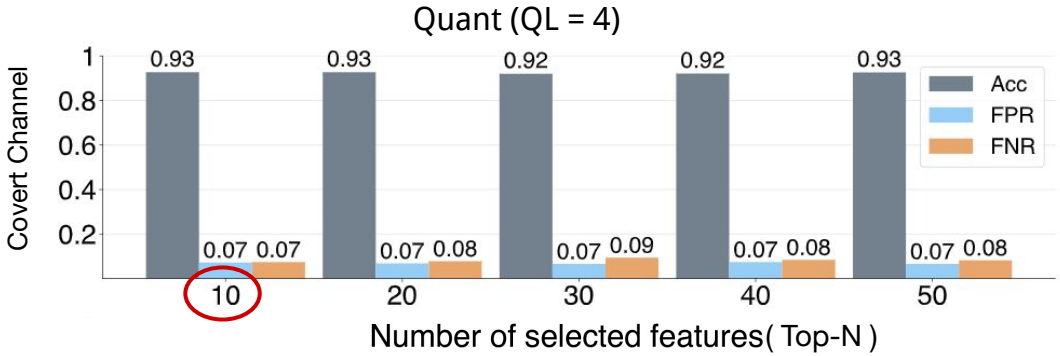
Full information = **3000B**
Detection: 96% accuracy

Quant (QL=4) = **188B**
Detection: 92% accuracy

FlowLens Scales the Amount of Inspected Flows and Retains Acc.

- **Covert Channel Detection** [Barradas et al.]
  - Legitimate / Modified Skype flows
  - Packet lengths + XGBoost

150x increase in measured flows



Full information = **3000B**
Detection: 96% accuracy

Quant (QL=4) + Trunc (top-10) = **20B**
Detection: 93% accuracy

FlowLens' Profiler Finds Good Quant. / Trunc. Parameters

- **Automatic profiling (Covert Channel):**
  - **48 valid** parameter combinations
  - Set max exploration of **10** combinations

Rank (accuracy-wise)	Combination
#1	(QL = 2, Top-n = all) = 0.960
#2	(QL = 3, Top-n = 50) = 0.951
#3	(QL = 0, Top-n = 30) = 0.947
Output	(QL = 3, Top-n = 10) = 0.944

Optimize for a reasonable Size vs Accuracy trade-off

FlowLens Imposes a Small Overhead on the Switch

- **CPU usage (ML component):**
  - Botnet detection (our largest model)
  - 140MB out of 32GB RAM
  - 5.6MB storage
  - ~200 µs per prediction

- **ASIC usage (Flow Marker Accumulator):**

Computational			Memory	
eMatch xBar	Gateway	VLIW	TCAM	SRAM
8.46%	5.21%	3.39%	0.00%	38.54%

Supports flow classification in the control plane

Supports the concurrent execution of other forwarding behaviors

Conclusions

- **FlowLens:** First traffic analysis system for generic ML-based security applications in prog. switches
- **Collects** compressed packet distributions, ensuring:
  - Classification accuracy
  - Small memory footprint
- **Classifies** flows directly on the switch
  - Saves communication, compute, and storage costs

Our code is available!
`https://github.com/dmbb/flowlens`

Thank You!

`https://web.ist.utl.pt/diogo.barradas`