An Efficient Design of Intelligent Network Data Plane

Guangmeng Zhou¹, Zhuotao Liu¹,², Chuanpu Fu¹, Qi Li¹,², Ke Xu¹,²

¹Tsinghua University
²Zhongguancun Laboratory

August, 2023
Outline

- Background
- Motivation
- Design
- Evaluation
- Conclusion
Background

- AI experiences increasingly popularity in network.
  - Model are deployed either on end-hosts or network control plane.

- Intelligent Data Plane (IDP)
  - Malicious Traffic Detection
  - ECN Threshold Adjustment
  - ……

However, IDP is “nearly impossible” using traditional switch chip.
Programmable switches enable IDP.

- Customizable Packet Processing
- Stateful and Persistent Storage

- Computation Constraints: simple operations, limited stages
- Storage Constraints: once register access, limited storage

Constraints pose many challenges for IDP.
Motivation

- Prior traffic analysis art on programmable switches
Motivation

- Prior art on programmable switches

**FlowLens, NetWarden**
- Control Plane: Classification
- Data Plane: Collect Information
- Line-speed

**Poseidon, Jaqen**
- Collect & Filters
- Learning-based
- Flow-aware

**Planter, Mousika**
- Extract & Classification

**NetBeacon**
- Line-speed
- Learning-based
- Flow-aware

Collect & Classification
Design

#1 How to get hardware-friendly learning models?
- Feature Engineering
- Multi-Phase Sequential Models

Data Plane Aware Model Design

- Origin Packets
- Feature Engineering
- Training Data
- Model Training
- Multi-phase Sequential Model
Design

#2 How to deploy learning models efficiently?
- Model Representation
- Stateful Storage Management

Data Plane Aware Model Design

Model Deployment
Design

Data Plane Aware Model Design

Model Deployment

#1 Get hardware-friendly learning models
- Feature Engineering
- Multi-Phase Sequential Models

#2 Deploy learning models efficiently
- Model Representation
- Stateful Storage Management
Design

- Feature Engineering
  - Features: extractable or computable on the pipeline
    - Packet-level: ttl, packet size, tcp window, proto, …
    - Flow-level: Attributes
      - Packet size, IPD, …
      - Min, Max, Mean, Var, Distributions,

- Multi-Phase Sequential Models
  - Model: Decision Tree, e.g., RandomForest, XGBoost, …
    - Flow-level features are dynamic as a flow proceeds.
Design

- Model Representation

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$f_1$</td>
<td>$f_2$</td>
</tr>
<tr>
<td>[0, 25)</td>
<td>[0, 46)</td>
</tr>
<tr>
<td>[0, 25)</td>
<td>[46, 256)</td>
</tr>
<tr>
<td>[25, 256)</td>
<td>[0, 256)</td>
</tr>
<tr>
<td>[0, 65)</td>
<td>[0, 256)</td>
</tr>
<tr>
<td>[65, 103)</td>
<td>[0, 256)</td>
</tr>
<tr>
<td>[103, 256)</td>
<td>[0, 256)</td>
</tr>
</tbody>
</table>

(1) Model table under range matching

Not supported by the switch

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$f_1$</td>
<td>$f_2$</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>01000001</td>
<td>[65, 66)</td>
</tr>
<tr>
<td>010001*</td>
<td>[66, 68)</td>
</tr>
<tr>
<td>010001**</td>
<td>[68, 72)</td>
</tr>
<tr>
<td>01001*</td>
<td>[72, 80)</td>
</tr>
<tr>
<td>010010*</td>
<td>[80, 96)</td>
</tr>
<tr>
<td>0100110*</td>
<td>[96, 100)</td>
</tr>
<tr>
<td>0100110</td>
<td>[100, 102)</td>
</tr>
<tr>
<td>0100110</td>
<td>[102, 103)</td>
</tr>
</tbody>
</table>

(2) Model table under ternary matching

Combinatorial explosion of entries

- Not supported by the switch
bit<8> f₁, bit<8> f₂, bit<8> f₃

(1) Model table under range matching
Not supported by the switch

(2) Model table under ternary matching
Combinatorial explosion of entries

Our way

Range Marking

(a₁) Feature table (f₁)

(a₂) Feature table (f₂)

(a₃) Feature table (f₃)

Each leaf node only consumes a single ternary entry in the model table.
bit<8> f₁, bit<8> f₂, bit<8> f₃

Our way

Higher Priority

Entries of a₁ from CRC algorithm

(a₁) Feature table (f₁)

(a₂) Feature table (f₂)

(a₃) Feature table (f₃)

(b) Model table

(1) Model table under range matching
Not supported by the switch

(2) Model table under ternary matching
Combinatorial explosion of entries
Design

- Stateful Storage Management
  - Index = Hash (5-tuple)

  Index Collision!

  Solutions

Differentiating Short and Long Flows

- Network traffic is skewed.
  ✓ Long flows: flow-level features
  ✓ Short flows: per-packet features

Long-short flow classification model

Handling Storage Index Collisions

- New flow can use the occupied register if
  1. Old flow class is determined confidently.
  2. Old flow is finished.
Design

- Integrated Data Plane Processing Logic

 Packet → Query Flow Class Table

1. Miss → Is the flow stored?
   - No → Per-packet classification
   - Yes → Update storage

2. Is the flow stored?
   - No → Could stored flow be occupied?
     - Yes → Long flow prediction
     - No → Update Flow Class Table

3. Could stored flow be occupied?
   - No → Initialize storage
   - Yes → Long flow prediction

4. Initialize storage
   - Long flow → User-defined processing
   - Short flow → Update Flow Class Table

5. Update Flow Class Table
   - Is class confident?
     - Yes → Flow-level classification
     - No → Store flow class

6. Flow-level classification
   - Store flow class

7. Read flow class
   - Yes → Class Table
   - No → No
Evaluation

- Setup

- Metric: Packet-level macro-accuracy

- Tasks:
  - P2P Application Fingerprinting
  - Covert Channel Detection
  - DDoS Attack Detection

- Baseline:
  - Mousika
Evaluation

- End-to-end

Compared to Mousika, NetBeacon has a significant accuracy improvement, i.e., 14% in P2P application fingerprinting, 38% in covert channel detection and 20% in DDoS attack detection.
Evaluation

- Model Representation

- NetBeacon achieves higher accuracies when consuming the same number of table entries.
- NetBeacon uses much fewer table entries when representing similarly-performing models.
Evaluation

- Deep Dive

- The storage size is important for traffic classification.
- The stateful management optimization is useful upon the limited storage.
- The flow-level information could be premature at earlier phase.
- The accuracy increases as more inference phases are appended.

(a) The impact of number of phases.

(c) The impact of the index size.
Conclusion

- An efficient IDP design which outperforms prior art in both model accuracy and representation efficiency.
  - Data plane aware model
  - Efficient model representation
  - Stateful storage management

- Source code: https://github.com/IDP-code/NetBeacon
- Email: zgm19@mails.tsinghua.edu.cn

Thanks! Questions?