FlowRadar: A Better NetFlow For Data Centers

Yuliang Li
Rui Miao  Changhoon Kim  Minlan Yu

USC University of Southern California  BAREFOOT NETWORKS
Flow coverage in data centers

• Flow coverage
  – Traffic monitoring needs to cover all the flows

Transient loop/blackhole

Fine-grained traffic analysis
Temporal coverage in data centers

- Temporal coverage
  - Traffic monitoring needs millisecond-level flow information

![Graph showing short-time scale loss rate and timely attack detection]

Short-time scale Loss rate  Timely attack detection
Key insight: division of labor

• Goal: report counters for all flows in fine-grained time granularity

Collector has limited bandwidth and storage

Overhead at the collector

NetFlow

Overhead at the switches

Limited per-packet processing time

Limited memory (10s of MB)

Mirroring

Needs sampling
Key insight: division of labor

- Goal: report counters for all flows in fine-grained time granularity
FlowRadar architecture

**Collector**

Correlate network-wide info to extract per-flow counter

**Network**

Each switch maintains a fast and efficient data structure for half-baked per-flow counter

**Periodic report**

Analyze

Flow & counter
Challenge: handling collision?

- **Handling** hash collision is hard
  - Large hash table → high memory usage
  - Linked list/Cuckoo hashing → multiple, non-constant memory accesses

<table>
<thead>
<tr>
<th>Flow</th>
<th>a</th>
<th>b</th>
<th>c</th>
</tr>
</thead>
<tbody>
<tr>
<td>PacketCount</td>
<td>...</td>
<td>...</td>
<td></td>
</tr>
</tbody>
</table>

Flow d collision

```
Flow d
```

PacketCount

<table>
<thead>
<tr>
<th>d</th>
<th>1</th>
</tr>
</thead>
</table>
Switch embraces collisions!

- **Handling** hash collision is hard
  - Large hash table → high memory usage
  - Linked list/Cuckoo hashing → multiple, non-constant memory accesses
- **Embrace** the collision
  - Less memory and constant #accesses
Switch embraces collisions!

- **Embrace** the collision: xor up all the flows
- **Less memory and constant #accesses**

Counting table

<table>
<thead>
<tr>
<th>FlowXor</th>
<th>a</th>
<th>a⊕b</th>
<th>b⊕c</th>
<th>b⊕c</th>
<th>a</th>
<th>c</th>
</tr>
</thead>
<tbody>
<tr>
<td>FlowCount</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>PacketCount</td>
<td>S(a)</td>
<td>S(a)+S(b)</td>
<td>S(b)+S(c)</td>
<td>S(b)+S(c)</td>
<td>S(a)</td>
<td>S(c)</td>
</tr>
</tbody>
</table>

S(x): #packets in x

[Invertible Bloom Lookup Table (arXiv 2011)]
Switch embraces collisions!

• 1. Check and update the flow filter
• 2. Update counting table

  – Packet from a new flow, update all fields
  – Subsequent packets update only PacketCount

Flow filter: identify new flow

Encoded flowset

[Invertible Bloom Lookup Table (arXiv 2011)]
Easy to implement in merchant silicon

• **Switch data plane**
  – Fixed operations in hardware
  – Small memory, 2.36MB for 100K flows

• **Switch control plane**
  – Control plane gets the small flowset every 10ms

• **We implemented it using P4 Language.**
FlowRadar architecture

Collector

Stage1. Single Decode

Stage2. Network-wide Decode

Network

Periodic report

Each switch maintains a fast and efficient data structure for half-baked per-flow counter.

Collector

Network

PacketCount

FlowCount

FlowXor

Bloom filter

Counting table

Encode flowset

Encoded structure

Encoded

Stage1. Single Decode

Stage2. Network-wide Decode

Analyze

Flow&counter

Encoded flowset
Stage 1. SingleDecode

Input: a single encoded flowset

Output: per-flow counters

Flow filter

Bloom filter

FlowXor ...
FlowCount ...
PacketCount ...

Counting table

<table>
<thead>
<tr>
<th>Flow</th>
<th>#packet</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>S(a)</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
Stage 1. SingleDecode

- Find a **pure cell**: a cell with one flow
- Remove the flow from all cells

### FlowXor

<table>
<thead>
<tr>
<th>Flow</th>
<th>#packet</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>1</td>
</tr>
</tbody>
</table>

### Flow filter

<table>
<thead>
<tr>
<th>FlowXor</th>
<th>FlowCount</th>
<th>PacketCount</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>a ⊕ b</td>
<td>2</td>
<td>12</td>
</tr>
<tr>
<td>b ⊕ c ⊕ d</td>
<td>3</td>
<td>13</td>
</tr>
<tr>
<td>b ⊕ c ⊕ d</td>
<td>3</td>
<td>13</td>
</tr>
<tr>
<td>a ⊕ b</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>c ⊕ d</td>
<td>2</td>
<td>6</td>
</tr>
</tbody>
</table>

**Decoded:**

- Pure cell
- Flow XOR
- Flow Count
- Packet Count
Stage 1. SingleDecode

- Find a cell with one flow (pure cell)
- Remove the flow from all cells
  - Create more pure cells
- Iterate until no pure cells

Decoded:

<table>
<thead>
<tr>
<th>Flow</th>
<th>#packet</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>5</td>
</tr>
</tbody>
</table>

Flow filter

<table>
<thead>
<tr>
<th>FlowXor</th>
<th>0</th>
<th>b</th>
<th>b⊕c⊕d</th>
<th>b⊕c⊕d</th>
<th>0</th>
<th>c⊕d</th>
</tr>
</thead>
<tbody>
<tr>
<td>FlowCount</td>
<td>0</td>
<td>1</td>
<td>3</td>
<td>3</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>PacketCount</td>
<td>0</td>
<td>7</td>
<td>13</td>
<td>13</td>
<td>0</td>
<td>6</td>
</tr>
</tbody>
</table>
Stage 1. SingleDecode

Decoded:

<table>
<thead>
<tr>
<th>Flow</th>
<th>#packet</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>5</td>
</tr>
<tr>
<td>b</td>
<td>7</td>
</tr>
</tbody>
</table>

We want to leverage the network-wide info to decode more flows.
FlowRadar architecture

Collector

Stage1. Single Decode

Stage2. Network-wide Decode

Network

Encoded flowset

Encoded flowset

Encoded flowset

Periodic report

Analyze

Flow & counter
Key insight: overlapping sets of flows

- The sets of flows overlap across hops
  - We can use the redundancy to decode more flows
- Use different hash functions across hops
- Provision memory based on $\text{avg}(\#\text{flows})$, not $\text{max}(\#\text{flows})$
  - SingleDecode for normal case
  - Network-wide decoding for bursts of flows

Collector can leverage flow sets from all switches to decode more
Challenge 1: sets of flows not fully overlapped

- Flows from one switch may go to different next hops
- One switch receive flow from multiple hops
Challenge 1 solution: use flow filter to check

- Generalize to network
  - No need for routing info
  - Incremental deployment
Challenge 2: counters are different across hops

• The counter of a flow may be different across hops
  – Some packets may get lost
  – On-the-fly packets
Challenge 2 solution: solve linear equations

• We got full list of flows
• Combine with counting table
• Construct and solve a linear equation system for each switch
• Speed up by using counter’s properties to stop solver earlier

<table>
<thead>
<tr>
<th>FlowXor</th>
<th>FlowCount</th>
<th>PktCount</th>
</tr>
</thead>
<tbody>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
<th>c</th>
<th>d</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flow</td>
<td>#pkt</td>
<td></td>
<td></td>
</tr>
<tr>
<td>a</td>
<td>5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>b</td>
<td>7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>c</td>
<td>4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>d</td>
<td>2</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
FlowRadar architecture

Collector

Stage1.SingleDecode → Stage2.1 FlowDecode → Stage2.2 CounterDecode

Analyze

Flow&counter

Network

Encoded flowset

Encoded flowset

Encoded flowset

Periodic report
Evaluations

SingleDecode vs. Network-wide Decode

Collector

Stage1. SingleDecode

Stage2.1 FlowDecode

Stage2.2 CounterDecode

Analyze

Flow & counter

Bandwidth usage

Network

Memory efficiency

Periodic report

Encoded flowset

Encoded flowset

Encoded flowset
Evaluation

• Simulation of k=8 FatTree (80 switches, 128 hosts) in ns3
• Config the memory base on avg(#flow),
  – when burst of flows happens, use network-wide decode
• The worst case is all switches are pushed to max(#flow)
  – Traffic: each switch has same number of flows, and thus same memory
• Each switch reports the flowset every 10 ms.
Memory efficiency

FlowRadar: 24.8MB

FlowRadar: 2.36MB

#cell=#flow (Impractical)

Log scale
Other results

• Bandwidth usage
  – Only 0.52% based on topology and traffic of Facebook data centers (sigcomm’15)

• NetDecode improvement over SingleDecode
  – SingleDecode 100K flow, which takes 10ms
  – NetDecode 26.8% more flows with the same memory, which takes around 3 sec
FlowRadar analyzer

Collector

Stage1. SingleDecode  Stage2. Network-wide Decode

Network

Periodic report

Encoded flowset  Encoded flowset  Encoded flowset
Analysis applications

• Flow coverage
  – Transient loop/blackhole
  – Error in match-action table
  – Fine-grained traffic analysis

• Temporal coverage
  – Short time-scale per-flow loss rate
  – ECMP load imbalance
  – Timely attack detection
Per-flow loss map: better temporal coverage

- Detect loss faster than NetFlow

Switch 1
- 15 packets detected
- FlowRadar detects loss

Switch 2
- 14 packets detected
- NetFlow detects loss

35 packets detected
Conclusion

• Report counters for all flows in fine-grained time granularity
• Fully leverage the capability of both the switches and the collector
  – Switch: fixed per-packet processing time, memory-efficient
  – Collector: Network-wide decoding