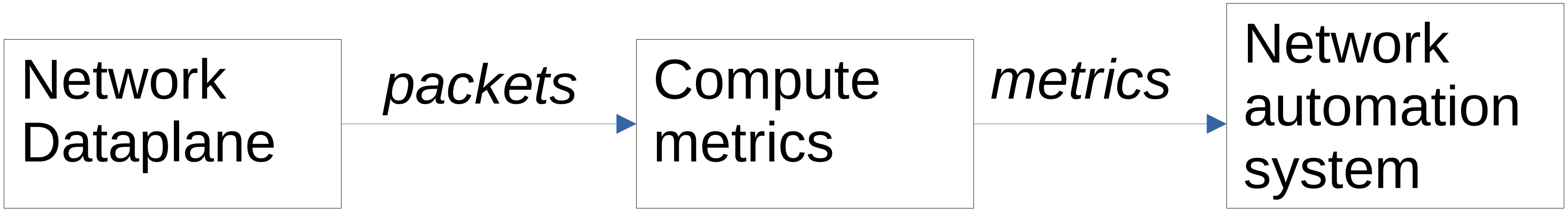
Designing Traffic Monitoring Systems for Self-driving Networks

Chris Misa, PhD Candidate
University of Oregon
2023-06-19

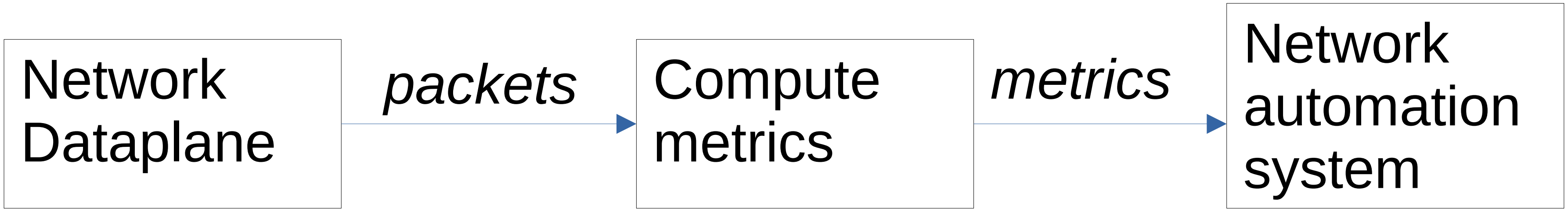
Traffic Monitoring

- *Traffic monitoring is **observing** packets in network...*
- *...and **computing** metrics for a particular goal.*



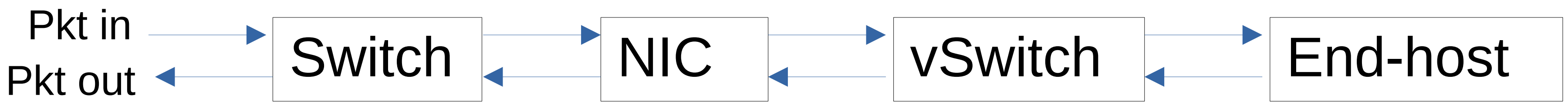
Traffic Monitoring

- *Traffic monitoring is **observing** packets in network...*
  - Single links: 400G, Switches: 2-3T.
- *...and **computing** metrics for a particular goal.*
  - Details for lots of traffic entities (flows).



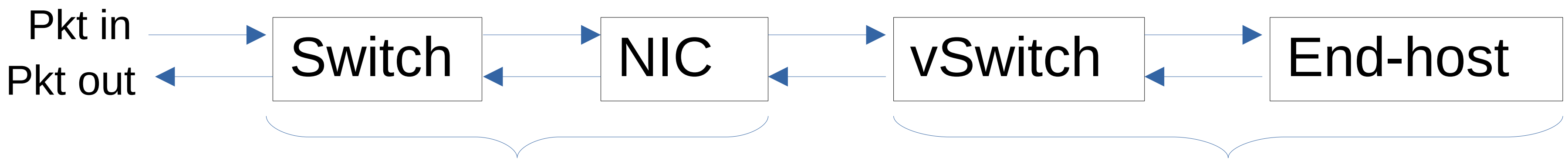
Traffic Monitoring **+Systems**

- Wide range of options for where to **compute.**
  - End-host CPU, NIC hardware, Switch hardware, etc.



Traffic Monitoring **+Systems**

- Wide range of options for where to **compute.**
  - End-host CPU, NIC hardware, Switch hardware, etc.

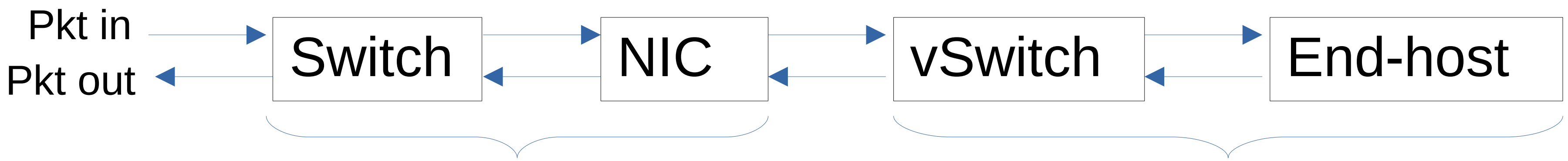


Hardware programming:
- (+) Fast per-packet processing
- (-) Limited memory
- (-) Limited operations

CPU programming:
- (+) Lots of memory
- (+) Lots of flexible ops.
- (-) Slow per-packet processing

Traffic Monitoring **+Systems**

- Wide range of options for where to **compute.**
  - End-host CPU, NIC hardware, Switch hardware, etc.



Hardware programming:
- (+) Fast per-packet processing
- (-) Limited memory
- (-) Limited operations

CPU programming:
- (+) Lots of memory
- (+) Lots of flexible ops.
- (-) Slow per-packet processing

...most systems are (actually) hardware + CPU hybrid.

Reqs. for **Self-Driving Networks**

- **R1:** *Set of monitored metrics changes at runtime.*
  - Monitoring is a *service* for automation.

Reqs. for **Self-Driving Networks**

- **R1:** *Set of monitored metrics changes at runtime.*
  - Monitoring is a *service* for automation.
- **R2:** *Resource efficiency for wide range of metrics.*
  - Including potentially non-linear feature vectors.

Reqs. for **Self-Driving Networks**

- **R1:** *Set of monitored metrics changes at runtime.*
  - Monitoring is a *service* for automation.
- **R2:** *Resource efficiency for wide range of metrics.*
  - Including potentially non-linear feature vectors.
- **R3:** *Remain robust in face of changing traffic.*
  - Changes in traffic cannot impact accuracy of results.

Reqs. for **Self-Driving Networks**

- **R1:** *Set of monitored metrics changes at runtime.*
  - Monitoring is a *service* for automation.

?

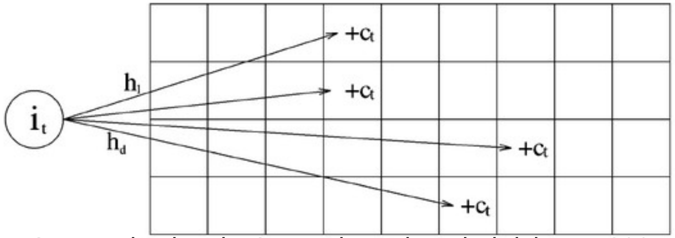
- **R2:** *Resource efficiency for wide range of metrics.*
  - Including potentially non-linear feature vectors.
- **R3:** *Remain robust in face of changing traffic.*
  - Changes in traffic cannot impact accuracy of results.

?

Currently lots of focus on R2, just starting to focus on R1 and R3.

Designs Proposed in Research

- **Sketches for efficient approximation.**



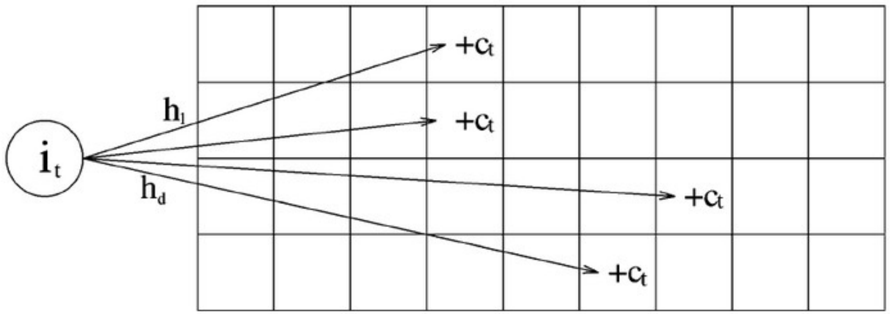
Count-min sketch: Cormode and Muthukrishnan, 2004.

- **“Map-reduce” model for flexible queries.**

```
ddos = PacketStream(1)
        .distinct(keys=('ipv4.dstIP', 'ipv4.srcIP'))
        .map(keys=('ipv4.dstIP',), map_values=('count',), func=('eq', 1,))
        .reduce(keys=('ipv4.dstIP',), func=('sum',))
        .filter(filter_vals=('count',), func=('geq', 45))
```

Sonata: Gupta et al., 2018.

Sketches for Efficient Approximation



- **Embrace hash collisions.**
- ***Adding* hash functions *multiplies* error.**

Pros:

- O(1) update.
- Several metrics can be computed.
  - Heavy hitters, cardinality, entropy, etc.

Cons:

- Typically fix flow key.
  - Hard to address **R1**.
- Error is function of (unknown) number of keys.
  - Hard to address **R3.**

“Map Reduce” for Flexibility

- **Language-based design.**
- **Partitioned across processors.**

```
ddos = PacketStream(1)
        .distinct(keys=('ipv4.dstIP', 'ipv4.srcIP'))
        .map(keys=('ipv4.dstIP',), map_values=('count', ...
        .reduce(keys=('ipv4.dstIP',), func=('sum',))
        .filter(filter_vals=('count',), func=('geq', 45))
```

… report destinations that receive from more than 45 distinct sources.

Pros:

- Unified interface for hardware and software platforms.
- Recent efforts also address **R1.**

Cons:

- Limited types of computations.
  - Simple “count” or “distinct” aggregations so far.
- Limited solutions for traffic dynamics (**R3**).

Some Recent Examples

- *Automatic DDoS defense:*[1]
  - Library of sketch-based detection and mitigation.
  - Compiled into switch + CPU policy implementation.

- *Automatic flow offloading:*[2]
  - Application of burst-based monitoring.

1. Jaqen, USENIX Sec. ’21.

2. Elixir, NSDI ’22.

Research Challenges

- **Define the role of traffic monitoring in network automation.**
  - What is produced? (Do ML models run in monitor?)[1]
  - How are computations specified? (Regular expressions?)[2]

1. FlowLens, NDSS ’21. 2. NetQRE, SIGCOMM ’17.

Research Challenges

- **Define the role of traffic monitoring in network automation.**
  - What is produced? (Do ML models run in monitor?)[1]
  - How are computations specified? (Regular expressions?)[2]
- **Address complex resource management problems.**
  - All kinds of dynamics?[3]
  - Contention with other data-plane applications?

1. FlowLens, NDSS ’21. 2. NetQRE, SIGCOMM ’17. 3. DynATOS, NSDI ’22.

Thanks!

(Questions and discussion later...)

cmisa@uoregon.edu