Packet-Level Analytics in Software without Compromises

HotCloud '18, July 9th, 2018, Boston, MA

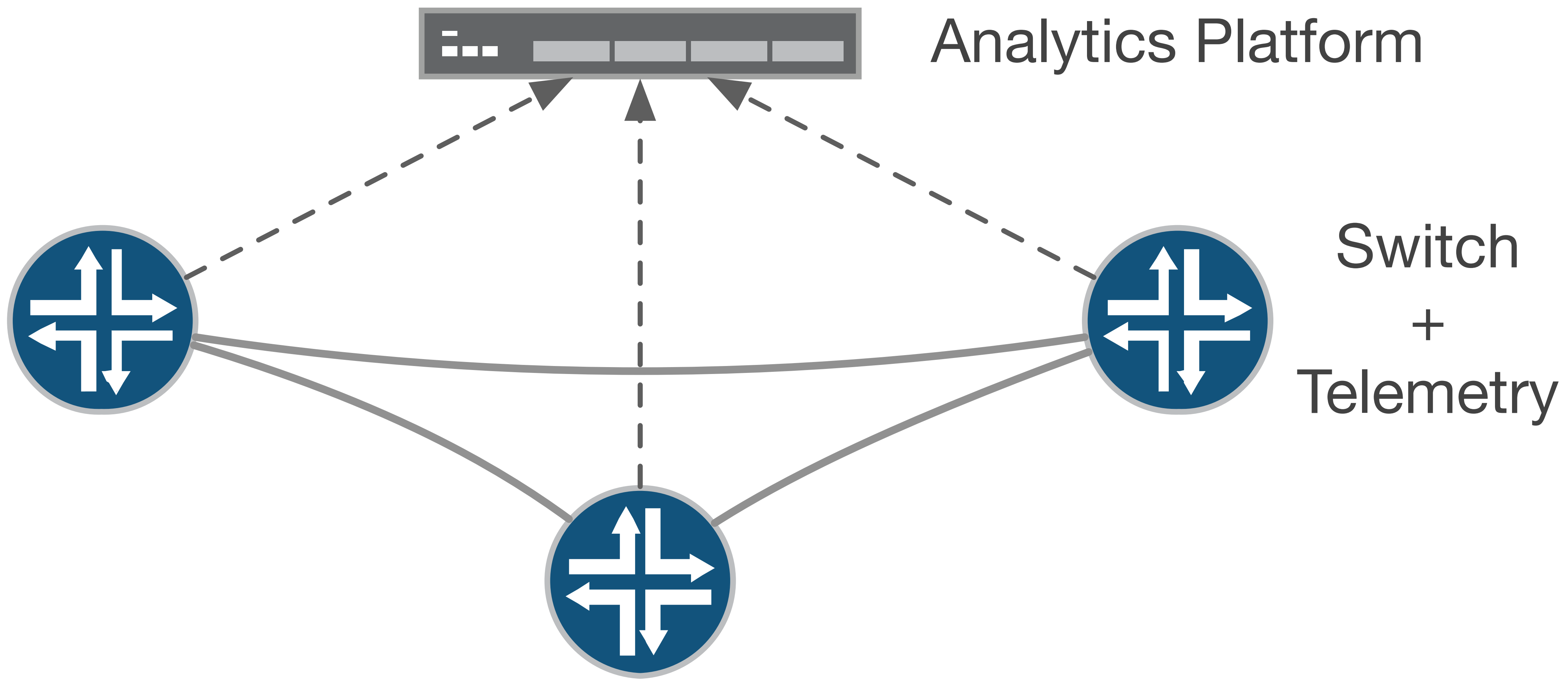
Oliver Michel
John Sonchack
Eric Keller
Jonathan M. Smith

University of Colorado Boulder

Penn

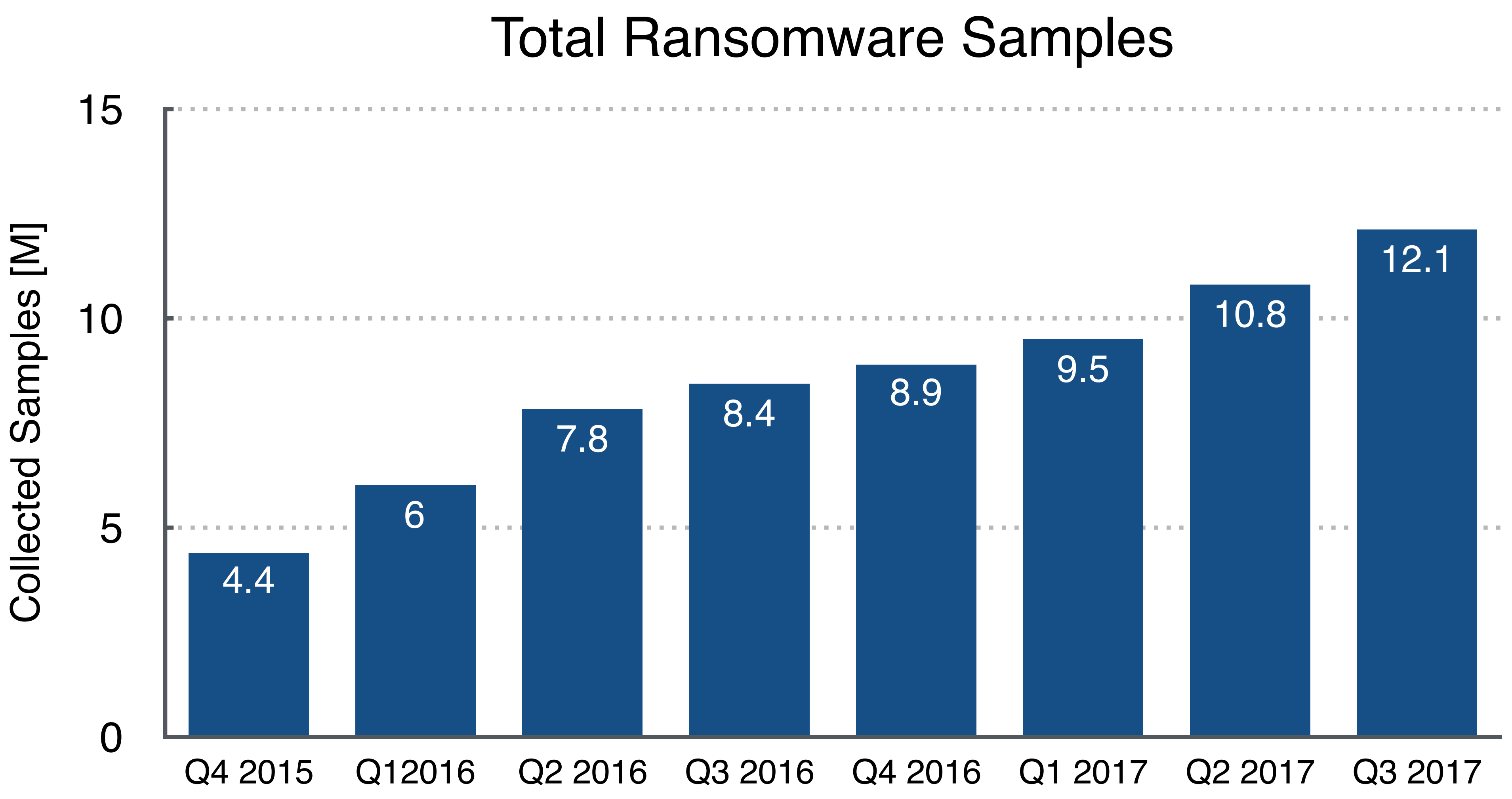
Network monitoring is important

- Security issues
- Performance issues
- Equipment failure
- Misconfiguration

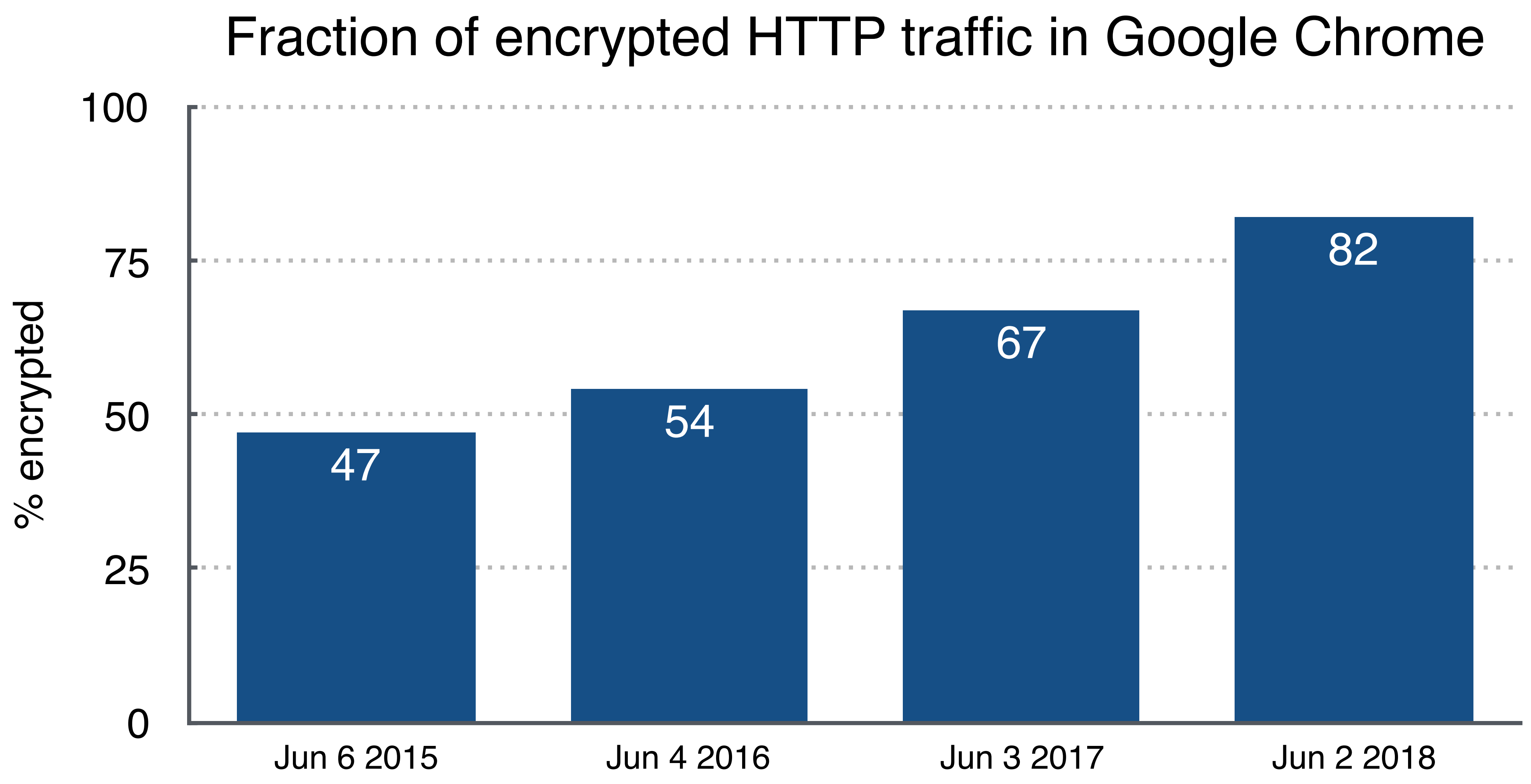


Challenging environment

- more traffic
- more threats
- encrypted traffic

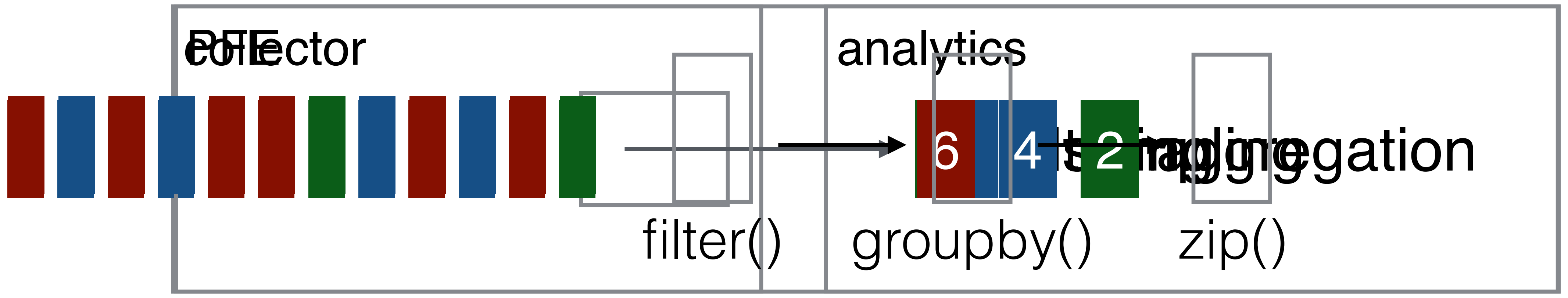


[McAfee Labs Thread Report Dec. 2017]



[Google Transparency Report 2018]

Existing systems make compromises

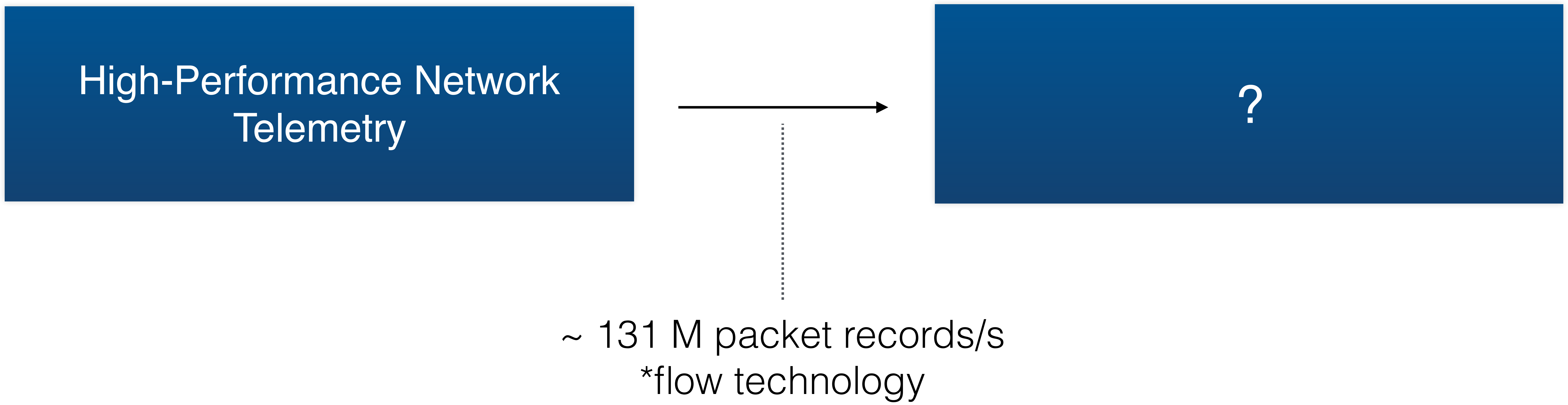


loss of information

loss of capability

Programmable Forwarding Engines

- Programmable Forwarding Engines
  - Marple [SIGCOMM 2017]
  - *flow [ATC 2018]

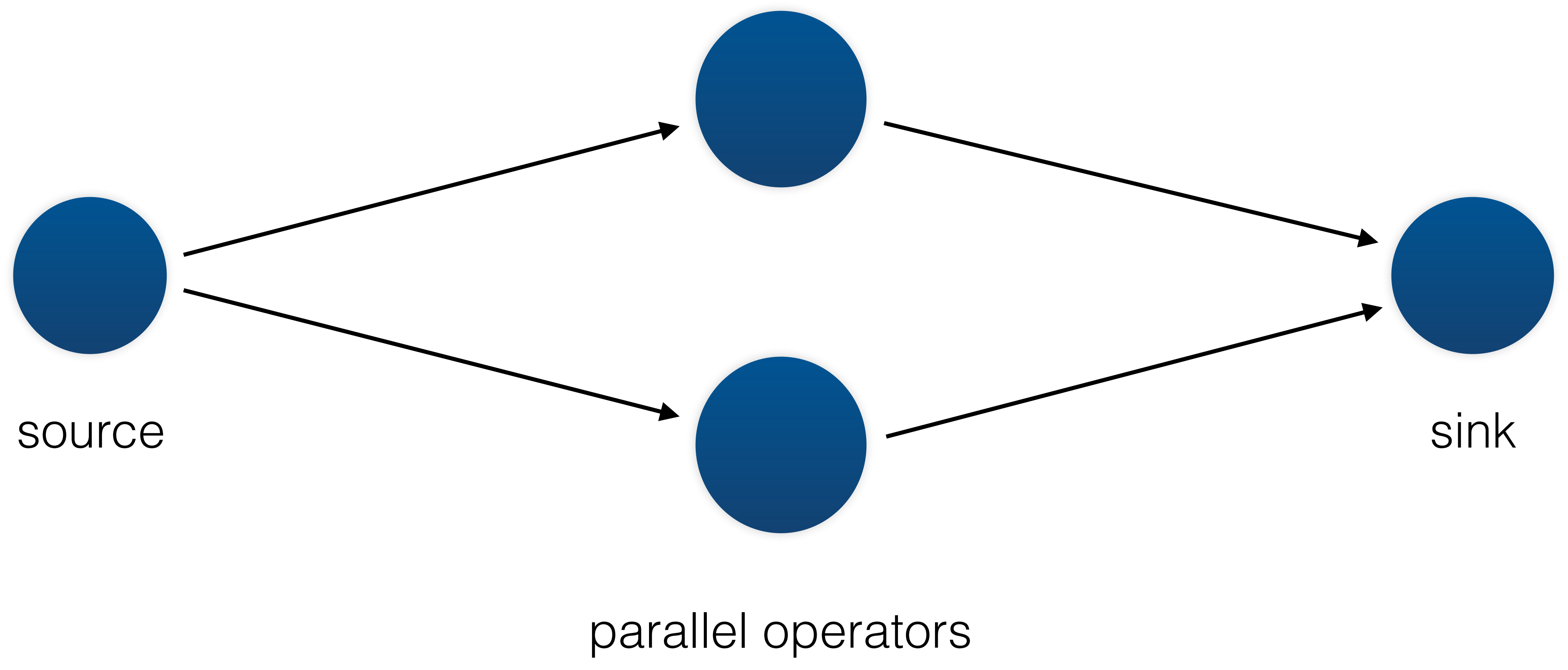


The ideal network analytics system

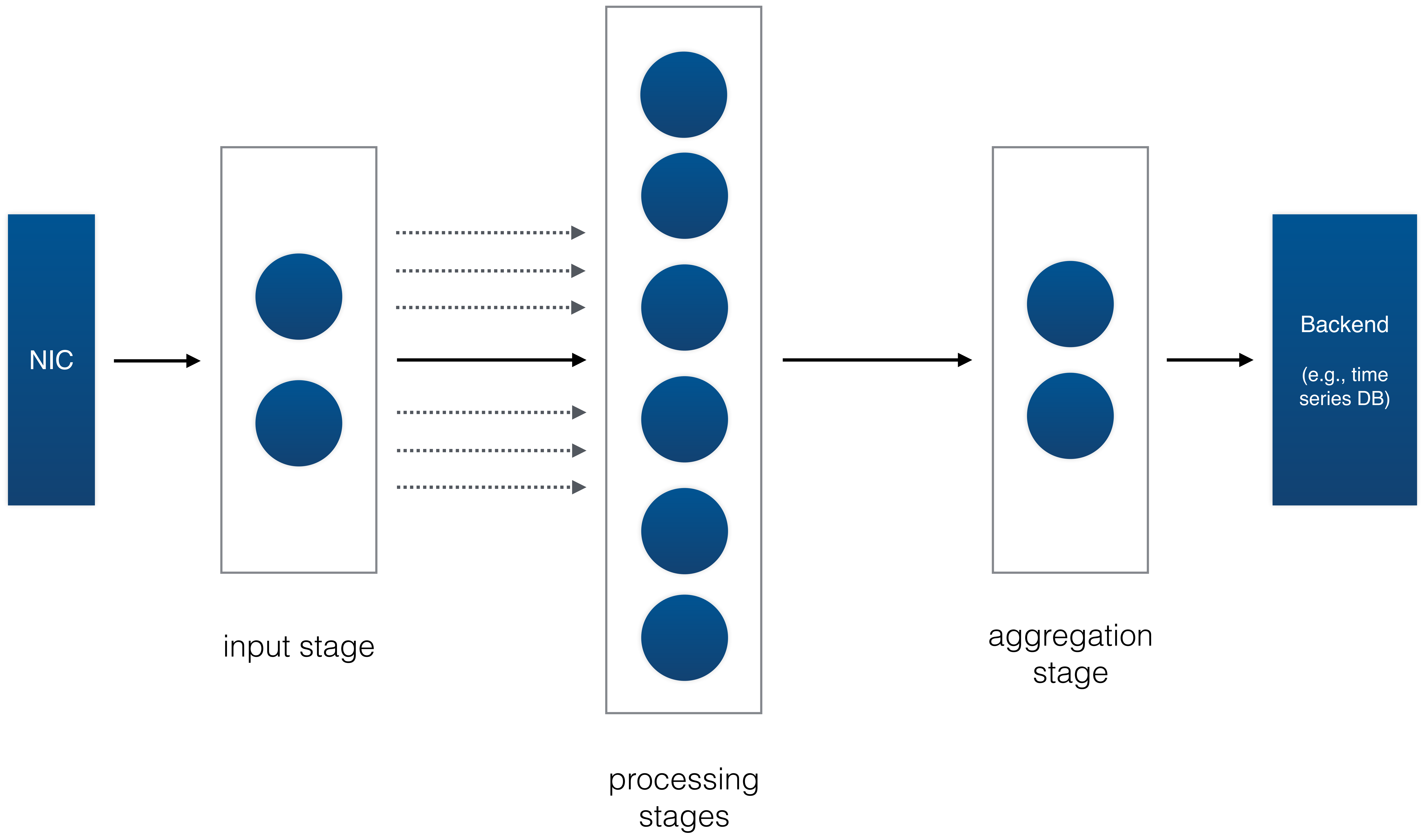
Is it possible to perform packet-level analytics on cloud-scale infrastructures without compromises?

- per-packet records
- x86 / general purpose programming language
- ~5M pps per core

Leveraging parallel architectures



Leveraging parallel architectures

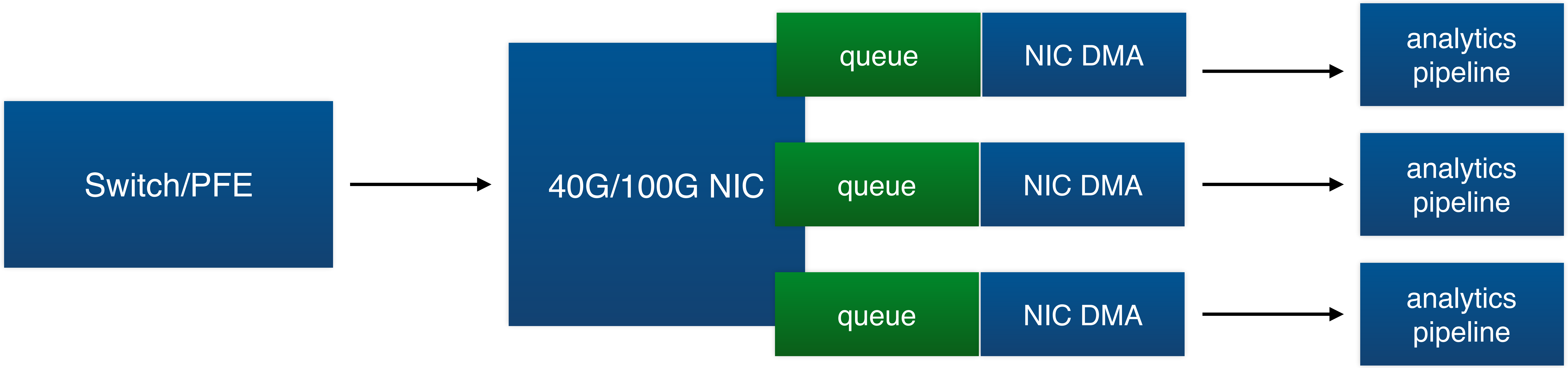


Characteristics of packet record workloads

Can we use properties of packet analytics workloads to our advantage?

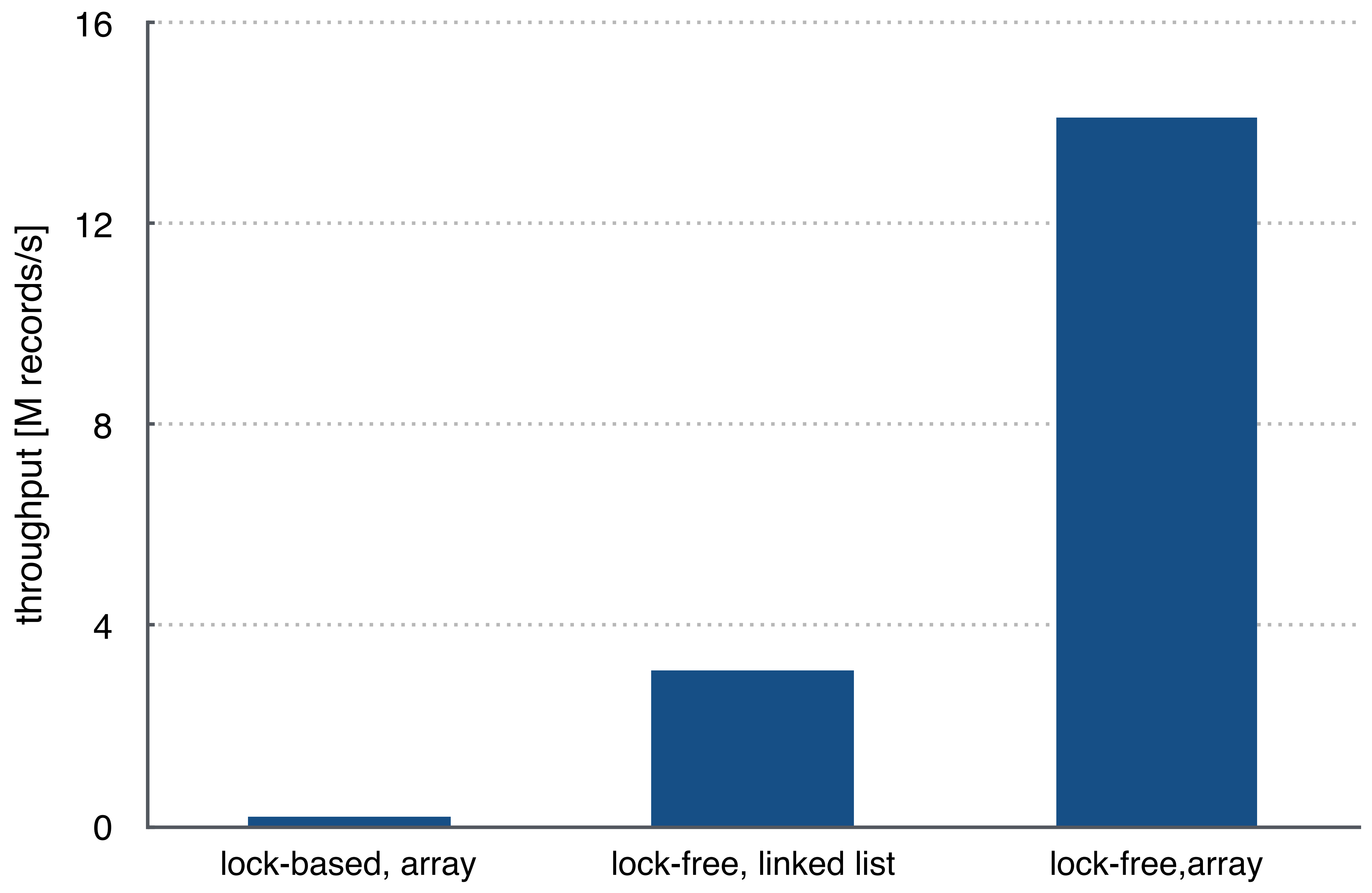
- Network attached input
- Partitionability/aggregation
- High rates, small, well-formed records

Network attached input

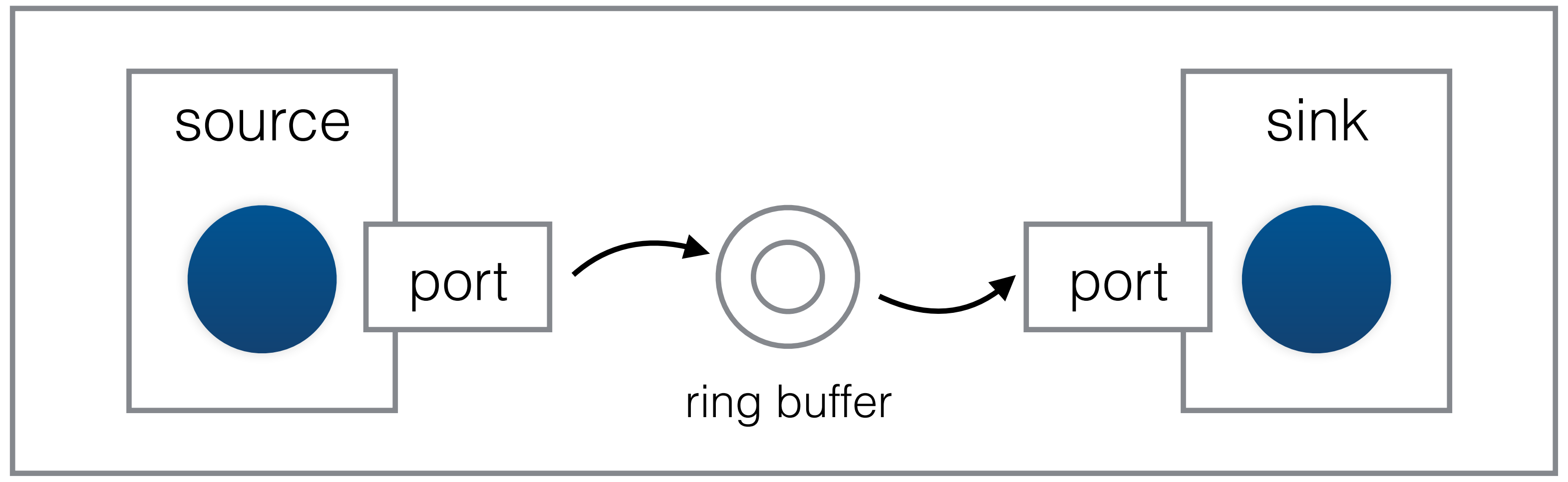


Many small records

- Array vs. linked list
- Lock-free design
- Wait-free design
- Zero-copy operations

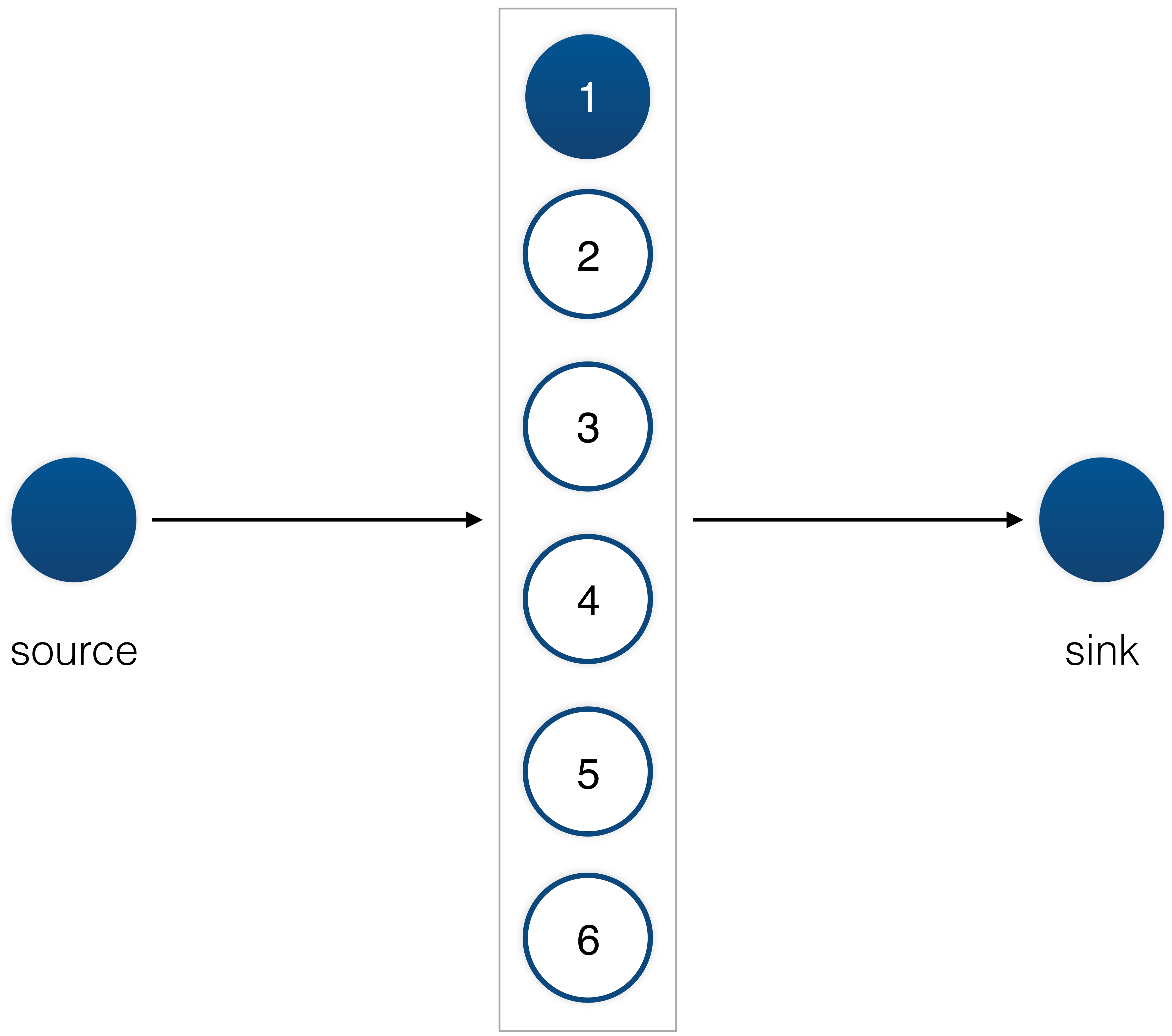


Programming Abstraction

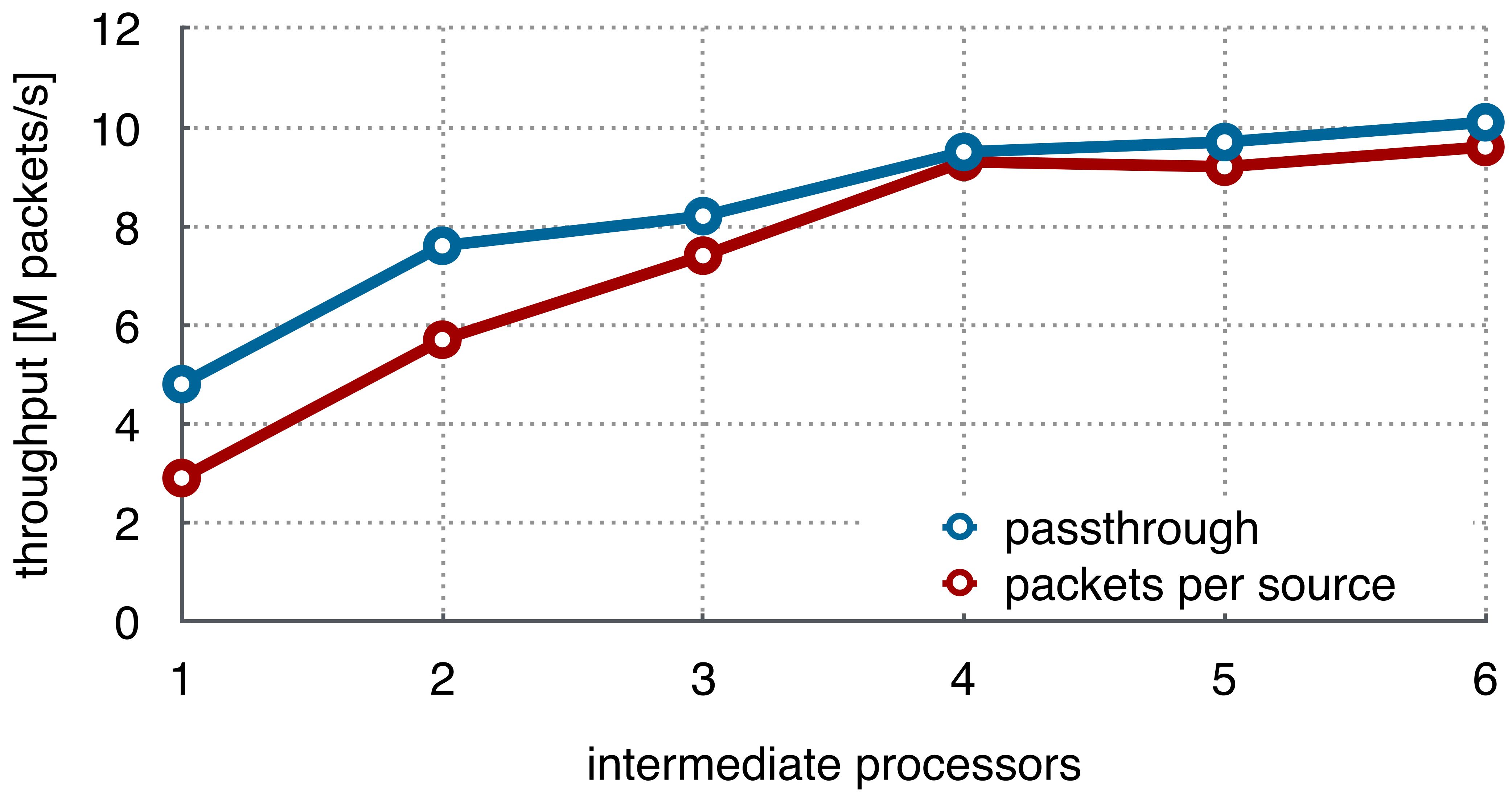



```cpp
int main(int argc, char** argv)
{
    jetstream::app app;
    auto source = app.add_stage<source>(1, “enp6s0f0”);
    auto sink   = app.add_stage<sink>(1, std::cout);
    app.connect<jetstream::pkt_t>(source, sink);
    app();
    return 0;
}
```

Performance

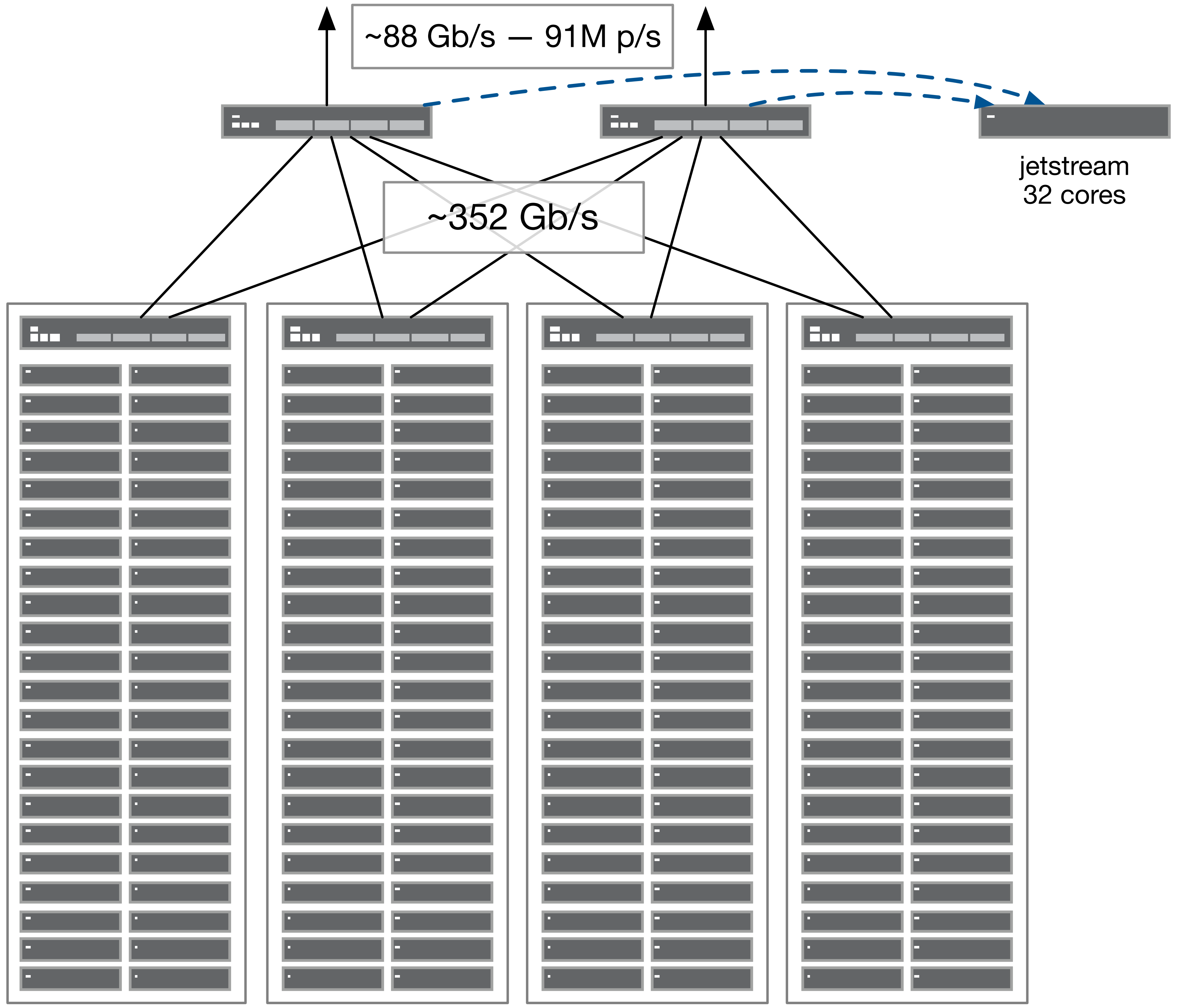


parallel operators



Performance

- Facebook web cluster: ~ 91M egress pps
- ~32 cores for basic packet-level insight
- 176 web servers — 1 analytics server: ~0.5% of cluster capacity



[Arjun Roy, Hongyi Zeng, Jasmeet Bagga, George Porter, and Alex C. Snoeren. 2015. Inside the Social Network's (Datacenter) Network. SIGCOMM Comput. Commun. Rev. 45, 4 (August 2015), 123-137]

Conclusion / Discussion

Is it possible to perform packet-level analytics on cloud-scale infrastructures without compromises?

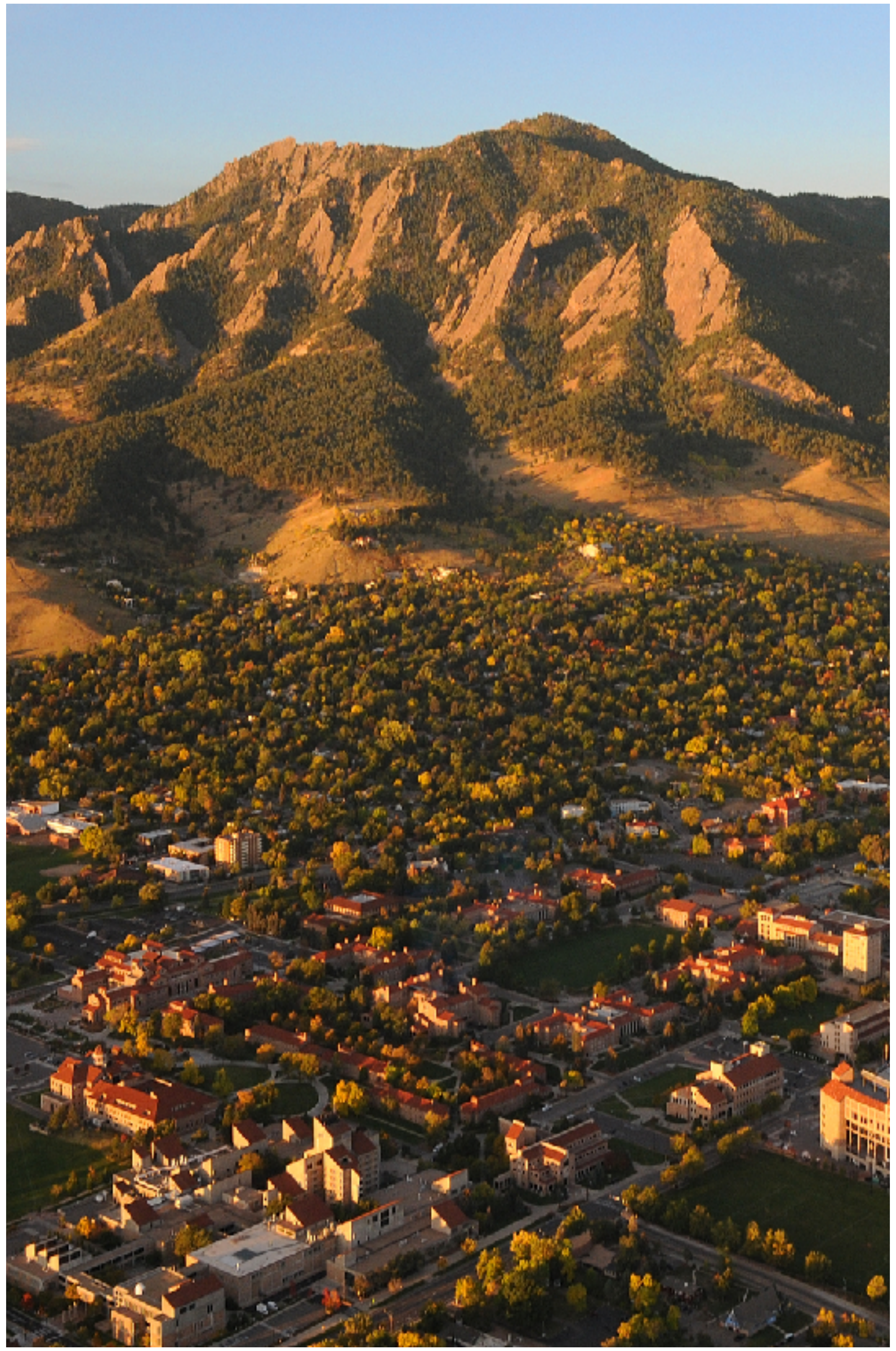
jetstream → high-performance, software network analytics platform

Q&A / Discussion

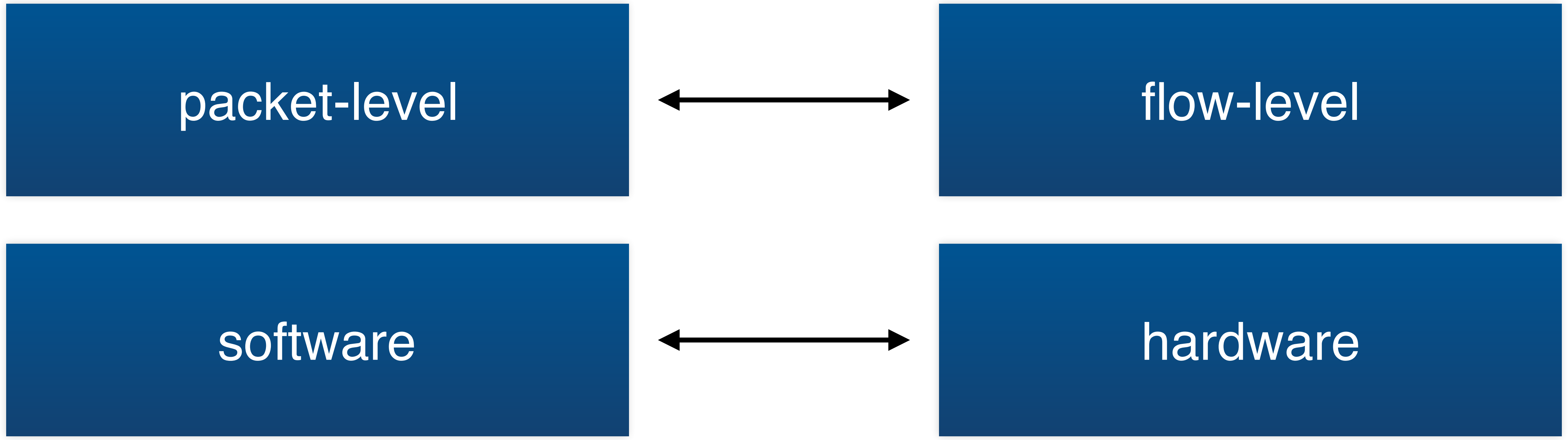
Oliver Michel

oliver.michel@colorado.edu
http://nsr.colorado.edu/oliver

University of Colorado Boulder



The *right* approach for network monitoring and analytics?



What data do we *need* for monitoring/debugging?

Panel Opening Slide

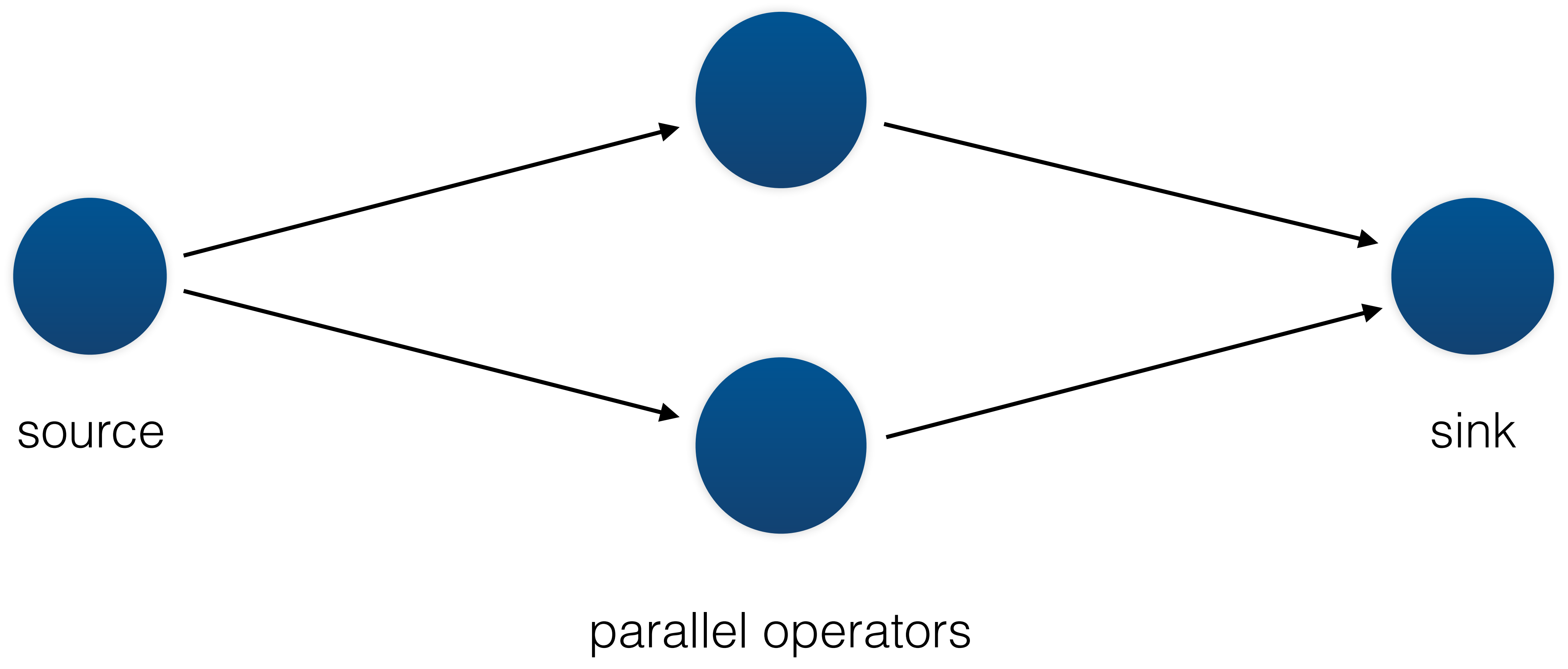
Packet-Level Analytics in Software without Compromises

Oliver Michel, John Sonchack, Eric Keller, Jonathan M. Smith
University of Colorado Boulder, University of Pennsylvania

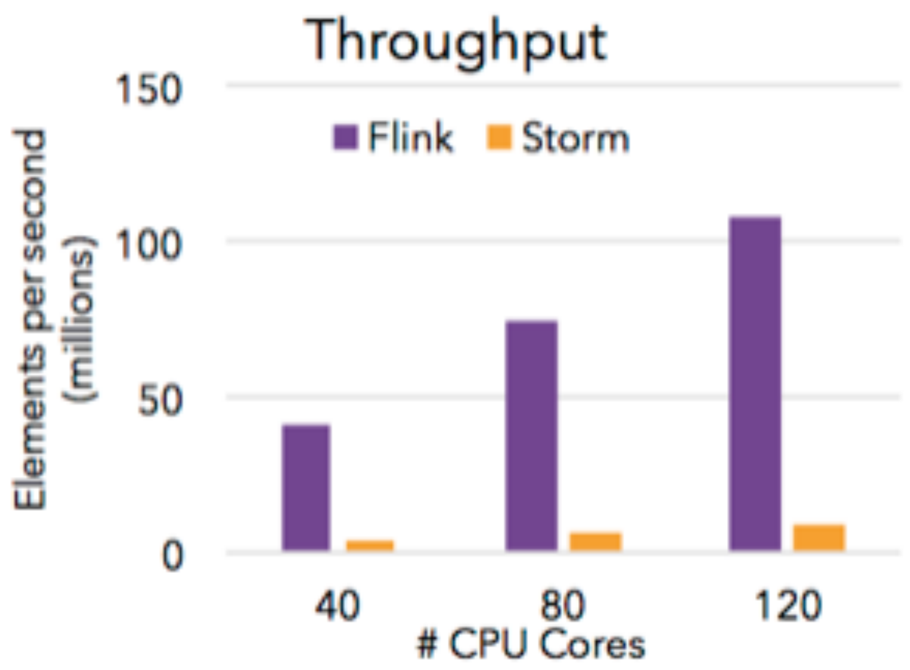
encrypted traffic → behavioral analysis

programmable forwarding engines → packet level record generation

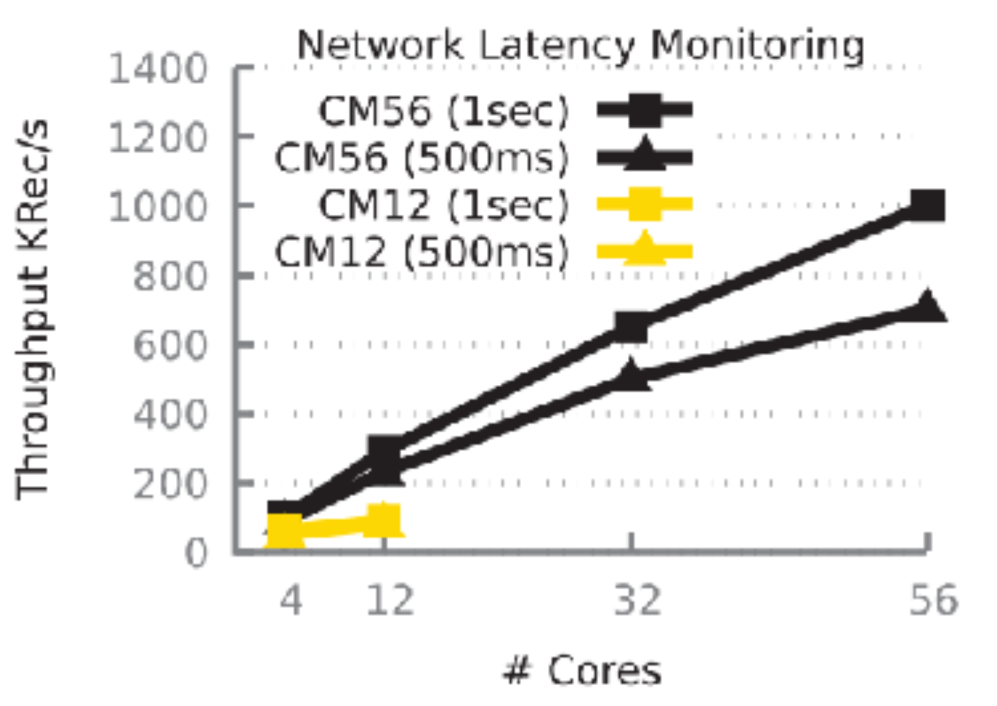
complex applications → software processing



Backup Slides



[Apache Flink]



[StreamBox Miao '18]

Programming abstraction

Processor definition

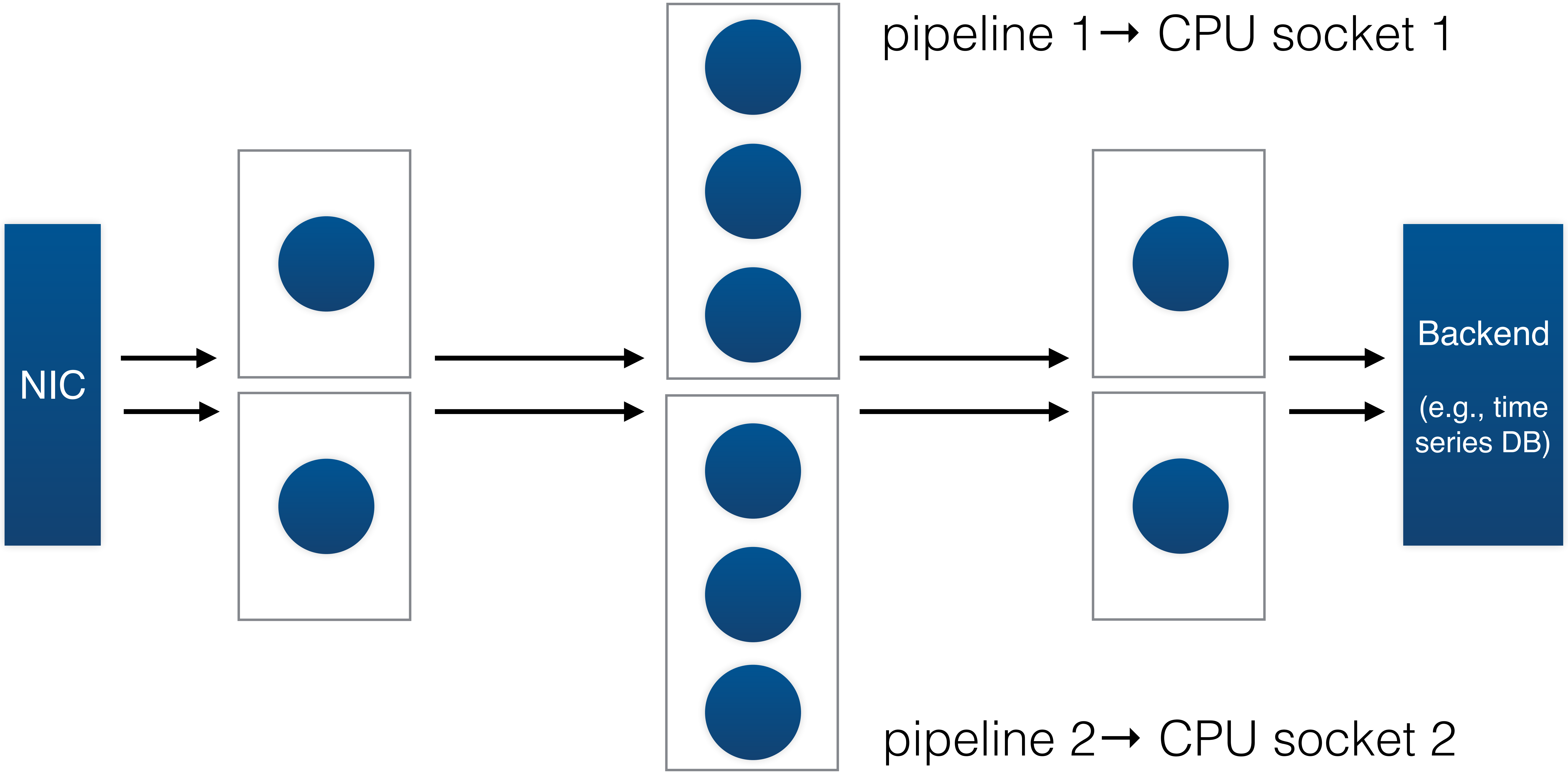
```
class source : public jetstream::proc {
  […]
};
```

```
explicit source(const std::string& iface_name_) : proc() {
   add_out_port<jetstream::pkt_t>(0);
  […]
}
```

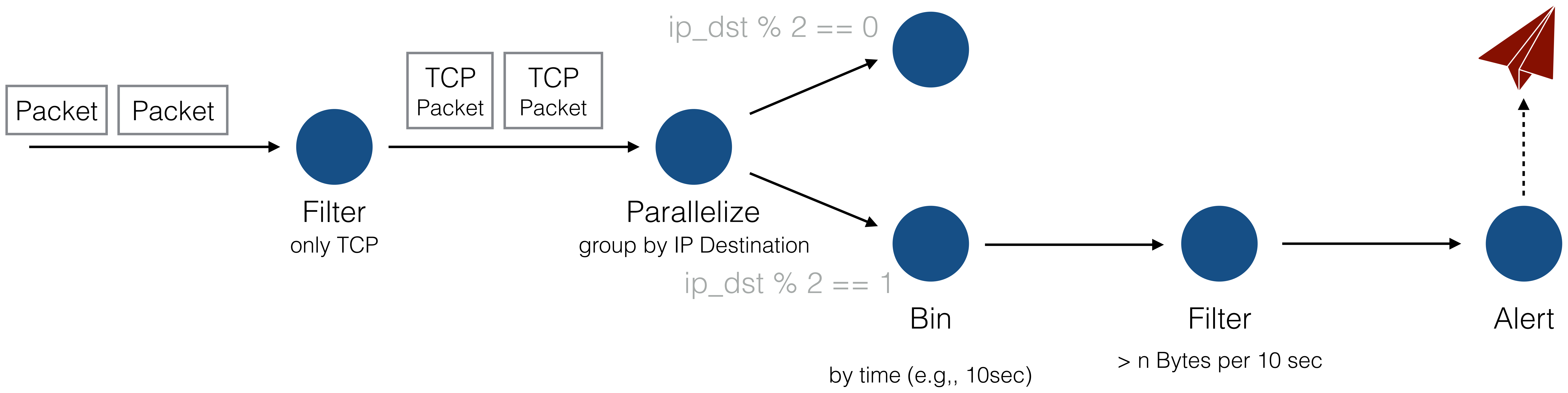
```
jetstream::signal operator()() override {
     out_port<pkt_t>(0)->enqueue(read_from_nic(_pkt),
                                 jetstream::signal::continue);
      return jetstream::signal::continue;
}
```

Jetstream architecture

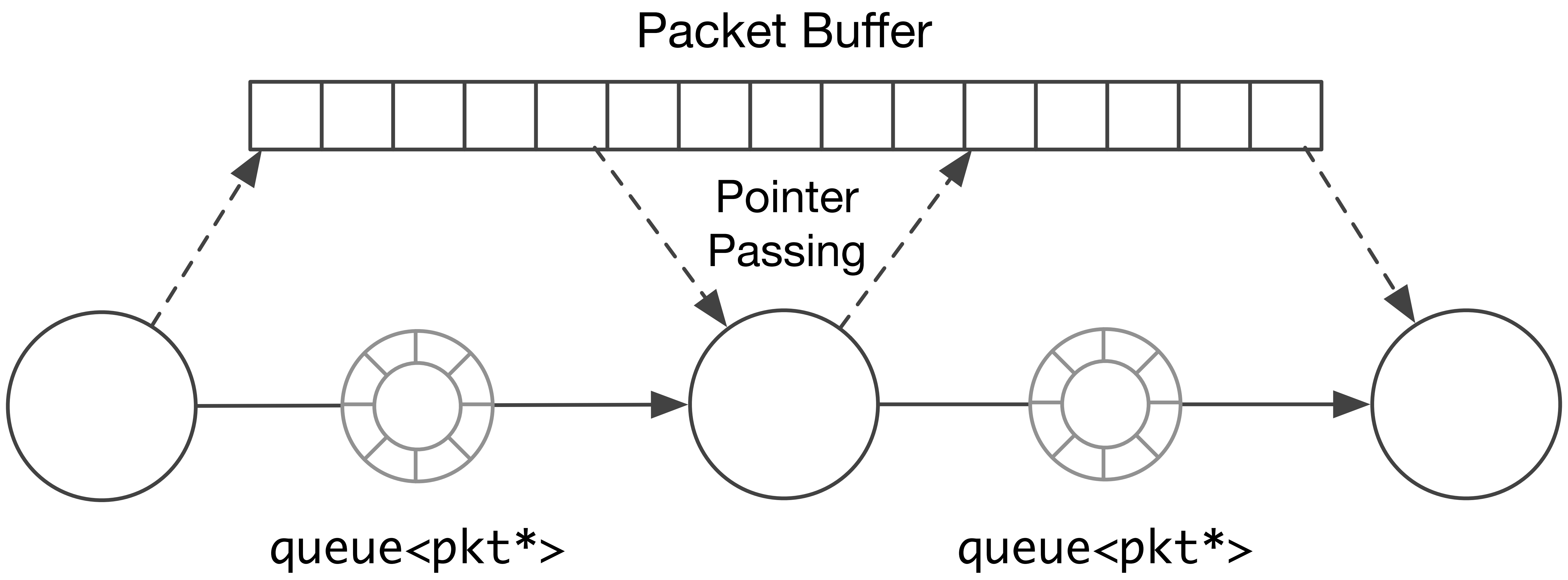
NUMA awareness



Stream Processing

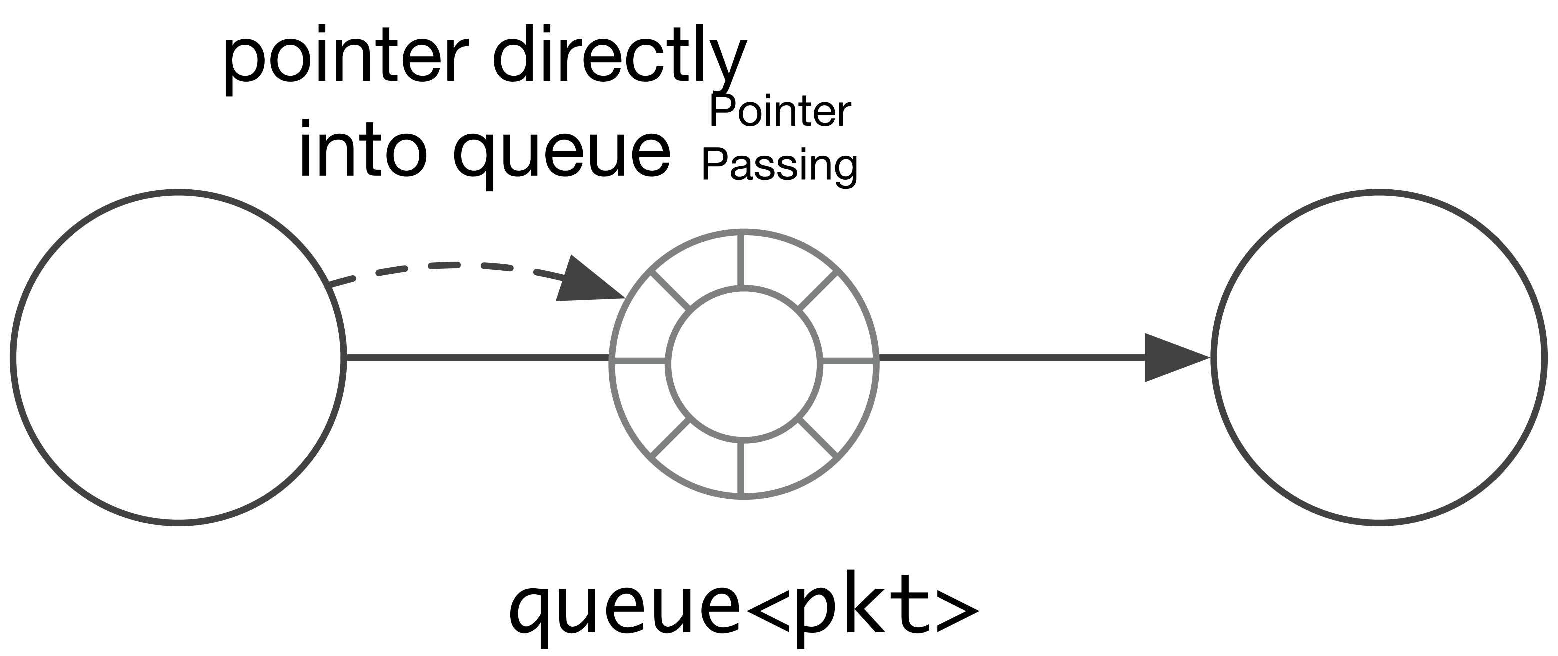


Reducing copy operations



Reducing copy operations

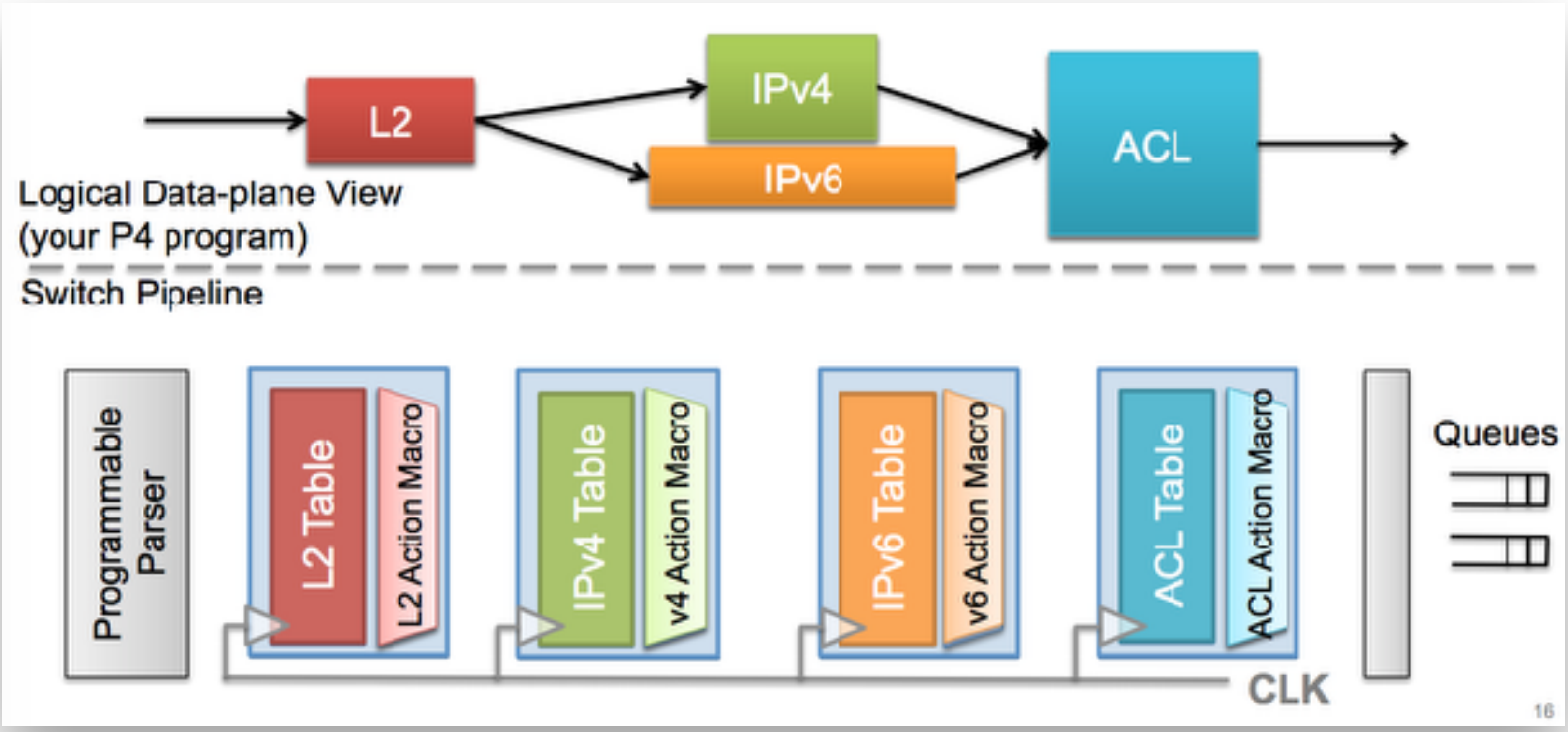
```
packet p;
p.ip_proto = 6;
q.enqueue(p);
```

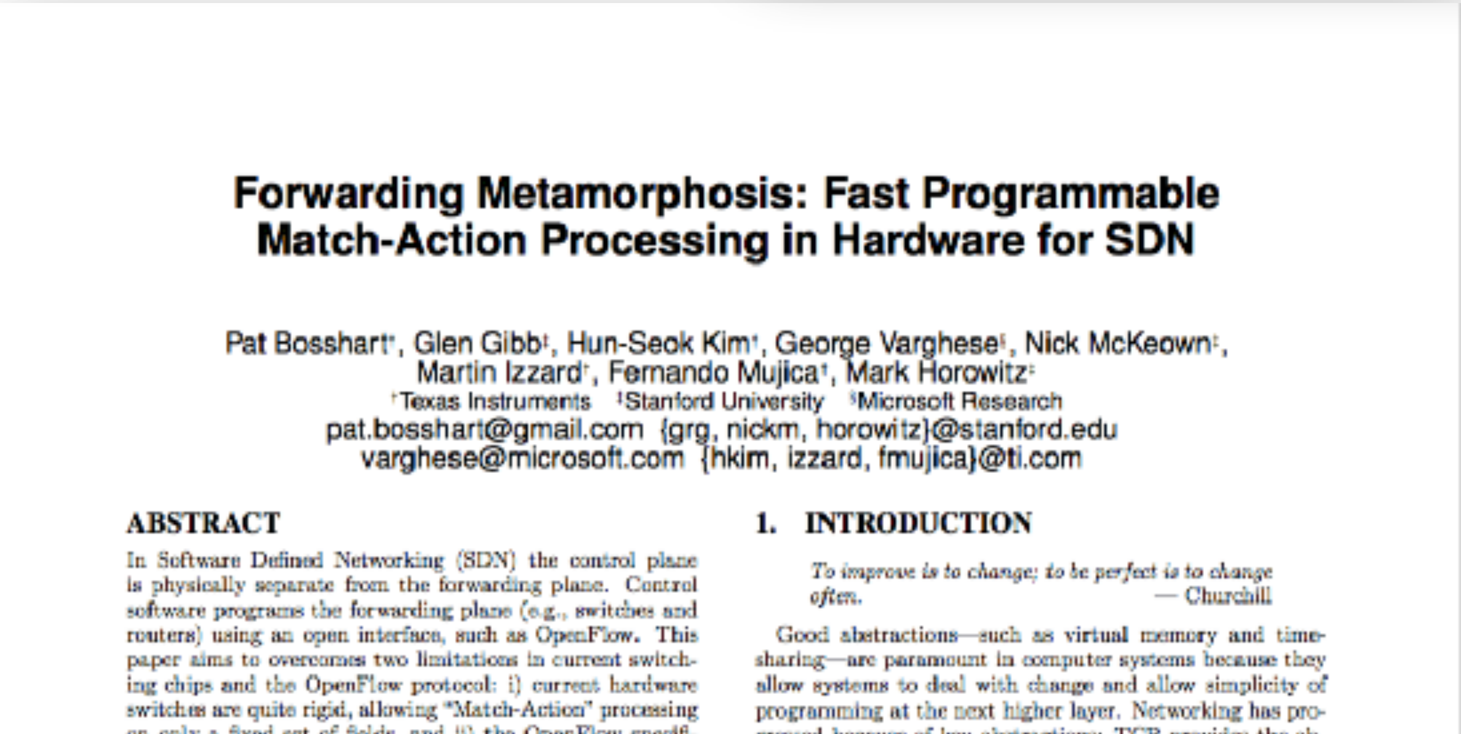



```
auto p = q.enqueue();
p->ip_proto = 6;
```

Technologies

- Programmable switches and PISA: Protocol Independent Switch Architecture
  - Reconfigurable match-action tables in hardware
  - multiple stages with TCAM/ALU pair, fixed processing time, guarantees line rate





Forwarding Metamorphosis: Fast Programmable Match-Action Processing in Hardware for SDN

Pat Bosshart[†], Glen Gibb[‡], Hun-Seok Kim[†], George Varghese[§], Nick McKeown[‡], Martin Izzard[†], Fernando Mujica[†], Mark Horowitz[‡]
[†]Texas Instruments [‡]Stanford University [§]Microsoft Research
pat.bosshart@gmail.com {grg, nickm, horowitz}@stanford.edu
varghese@microsoft.com {hkim, izzard, fmujica}@ti.com

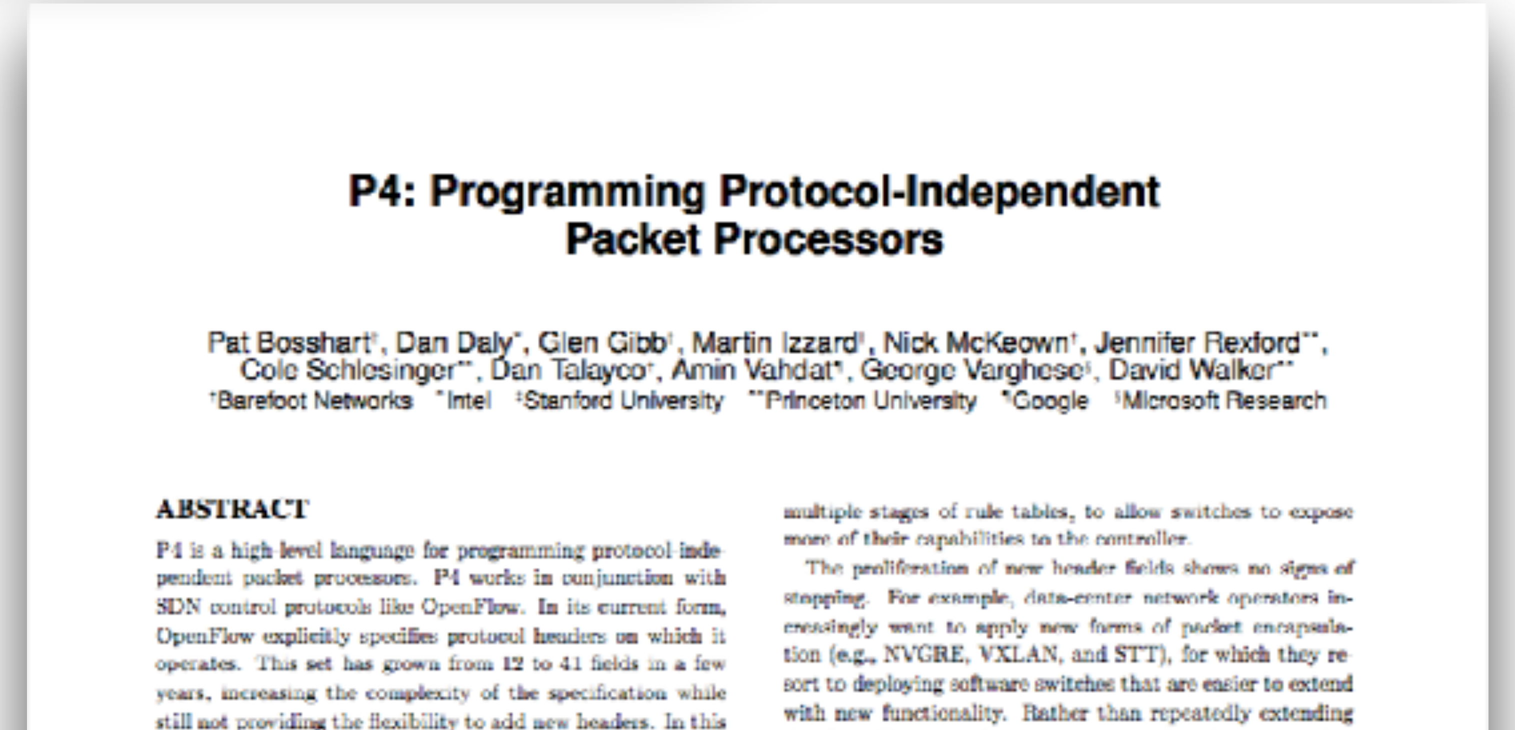
ABSTRACT

In Software Defined Networking (SDN) the control plane is physically separate from the forwarding plane. Control software programs the forwarding plane (e.g., switches and routers) using an open interface, such as OpenFlow. This paper aims to overcomes two limitations in current switching chips and the OpenFlow protocol: i) current hardware switches are quite rigid, allowing "Match-Action" processing

1. INTRODUCTION

To improve is to change; to be perfect is to change often. — Churchill

Good abstractions—such as virtual memory and time-sharing—are paramount in computer systems because they allow systems to deal with change and allow simplicity of programming at the next higher layer. Networking has pro-



P4: Programming Protocol-Independent Packet Processors

Pat Bosshart[†], Dan Daly[*], Glen Gibb[‡], Martin Izzard[†], Nick McKeown[‡], Jennifer Rexford[**], Cole Schlesinger[**], Dan Talayco[†], Amin Vahdat[¶], George Varghese[§], David Walker[**]
[†]Barefoot Networks [*]Intel [‡]Stanford University [**]Princeton University [¶]Google [§]Microsoft Research

ABSTRACT

P4 is a high-level language for programming protocol-independent packet processors. P4 works in conjunction with SDN control protocols like OpenFlow. In its current form, OpenFlow explicitly specifies protocol headers on which it operates. This set has grown from 12 to 41 fields in a few years, increasing the complexity of the specification while still not providing the flexibility to add new headers. In this

multiple stages of rule tables, to allow switches to expose more of their capabilities to the controller.

The proliferation of new header fields shows no signs of stopping. For example, data-center network operators increasingly want to apply new forms of packet encapsulation (e.g., NVGRE, VXLAN, and STT), for which they resort to deploying software switches that are easier to extend with new functionality. Rather than repeatedly extending