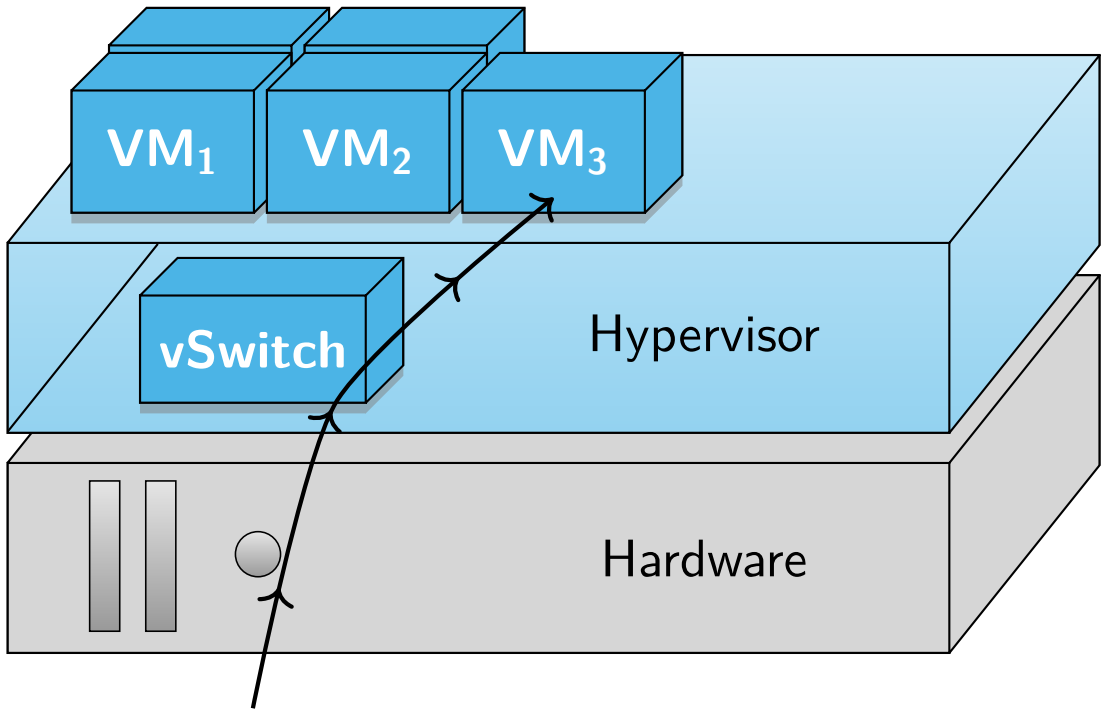
Oko: Open vSwitch Extensions with BPF

Paul Chaignon

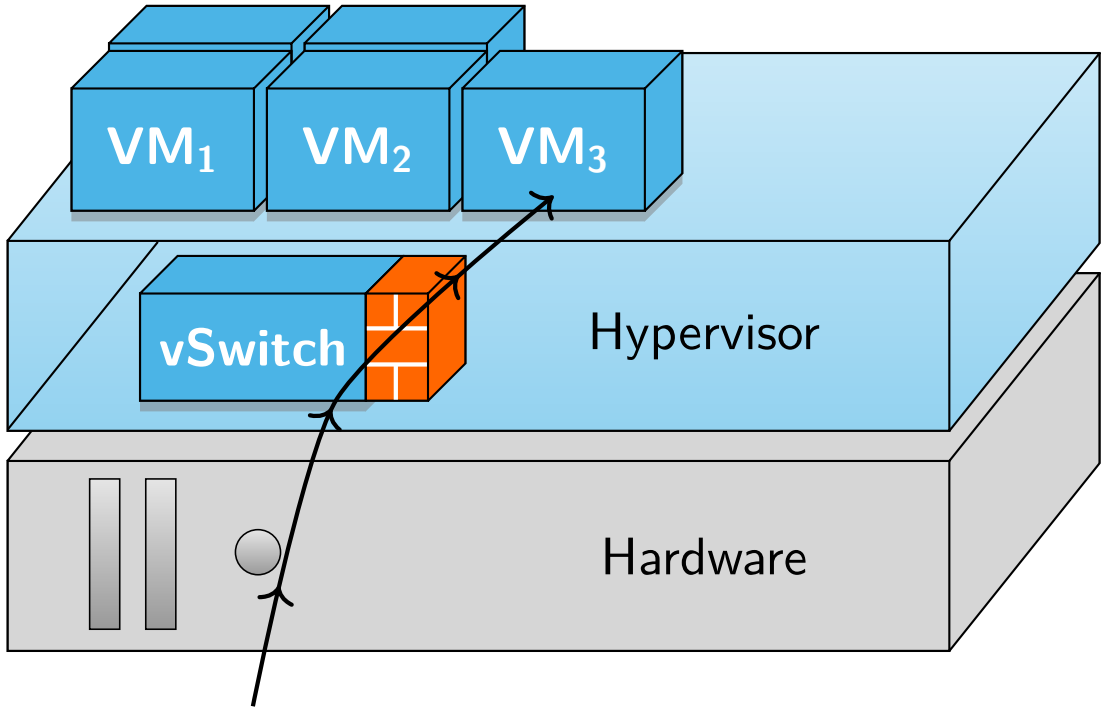
Orange Labs, France

February 2, 2019

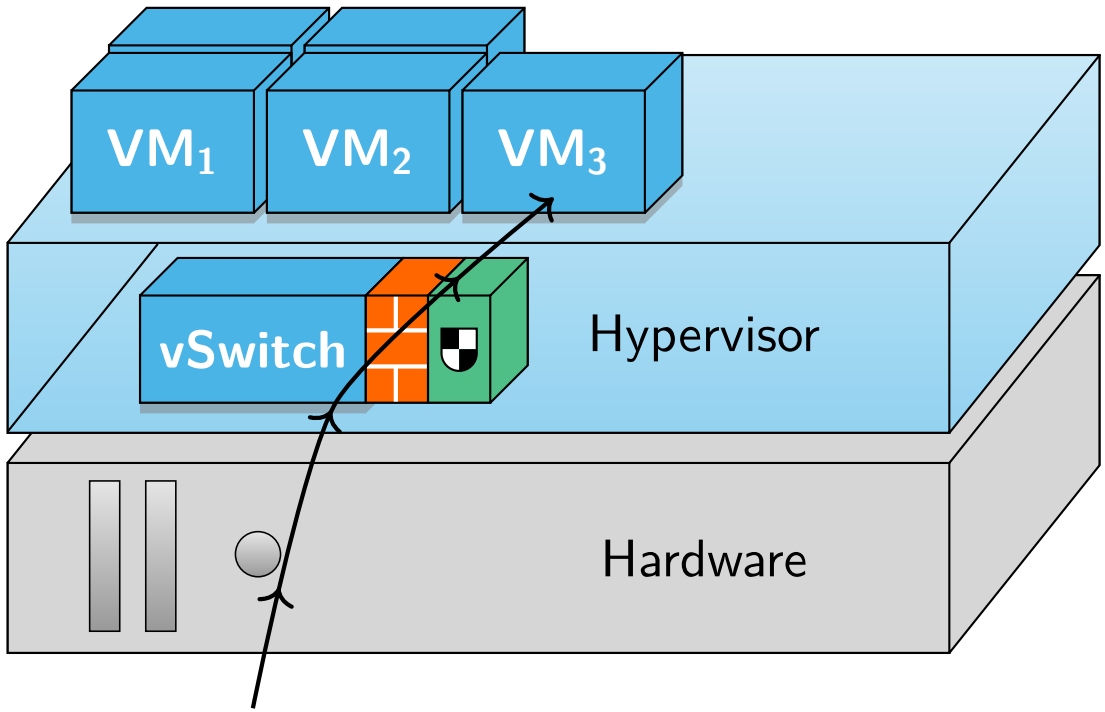
Extending Open vSwitch



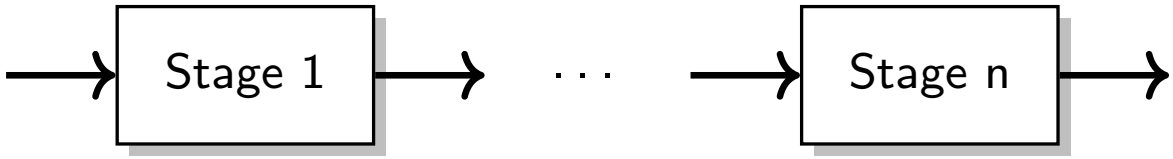
Extending Open vSwitch



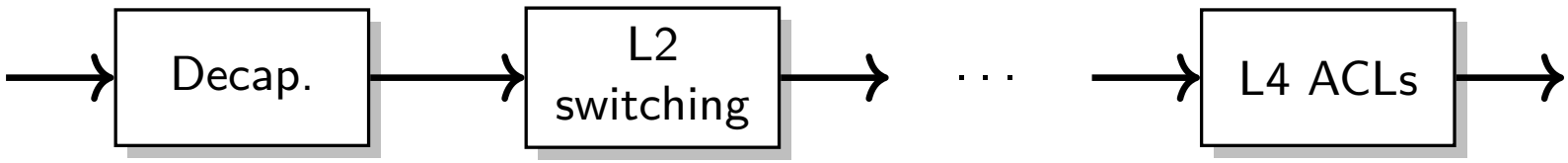
Extending Open vSwitch



Packet Processing Pipelines



Packet Processing Pipelines



Packet Processing

Flow Table

- Implemented as packet classifiers
- E.g., Open vSwitch, iptables

Code

- Executed or interpreted
- E.g., BPF, Cilium

Packet Processing

Flow Table

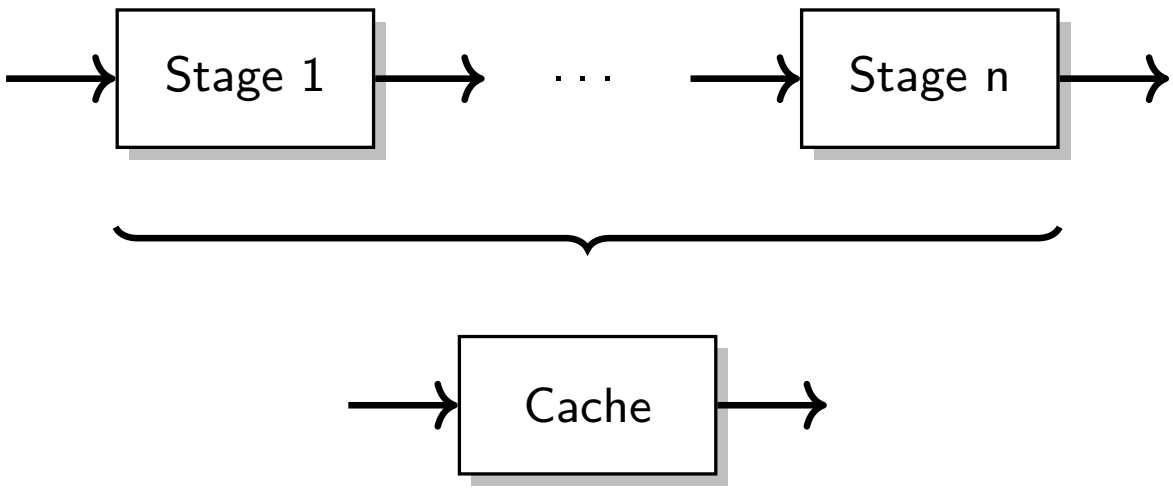
- Implemented as packet classifiers
- E.g., Open vSwitch, iptables

Code

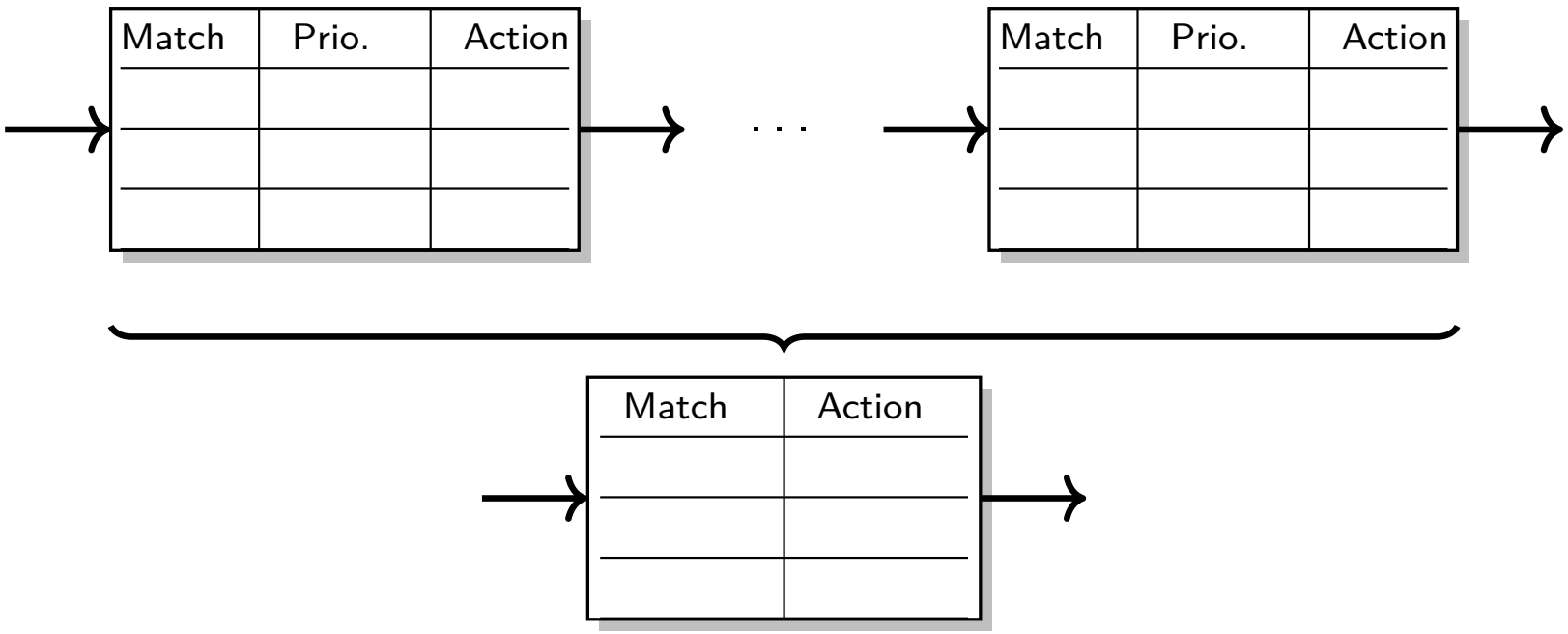
- Executed or interpreted
- E.g., BPF, Cilium

Why limit ourselves to flow tables when running on a CPU?

Flow Caching



Flow Caching in Open vSwitch



Oko's implementation

- Implementation on Open vSwitch-DPDK, in userspace
  - Better performance than kernel module
  - Easier to prototype
- Userspace BPF VM based on ubpf

`https://github.com/Orange-OpenSource/oko`

Oko's implementation

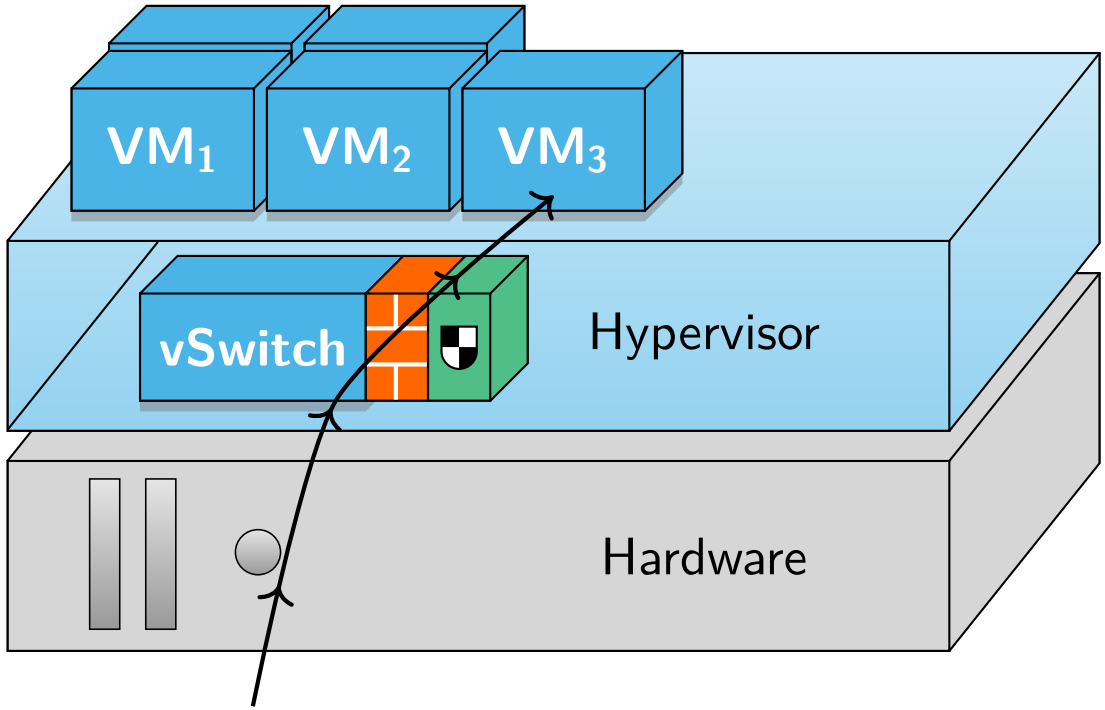
Implementation includes a partial verifier:

- Prohibits back-edges in CFG
- Checks bounds in memory accesses
- Prevents potential null memory accesses
- Invalidates divisions by zero

Todo:

- Prevent pointer leaks
- Check for unaligned memory accesses
- Optimizations

Extending Software Switches



Evaluations

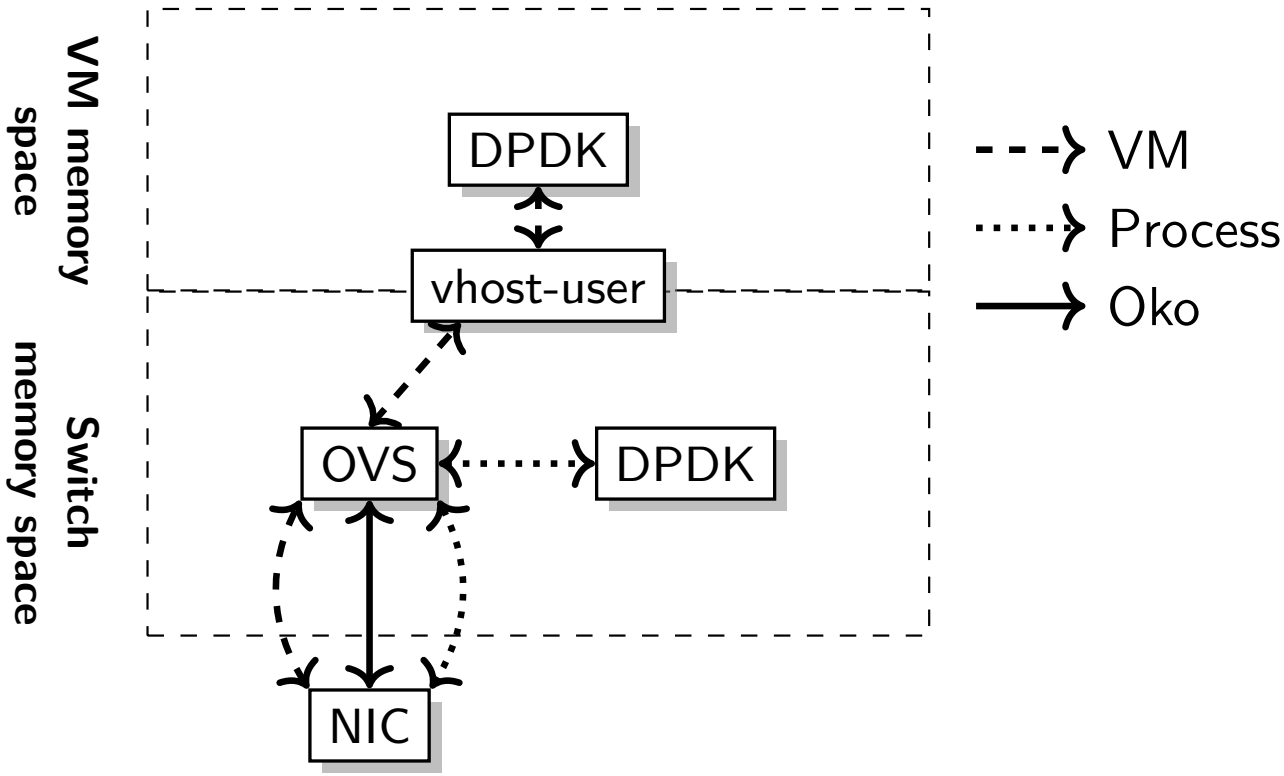


Figure: Evaluation setup. Packet copies are only necessary when crossing memory space boundaries.

Evaluations

1. Filtering using p0f TCP signatures
   - Compute a signature over TCP option fields
2. Stateful firewall
   - TCP state machine + hash table lookup over 4-tuple
3. Dapper: TCP performance analysis
   - Extract TCP options + hash table lookup over 4-tuple

Evaluations

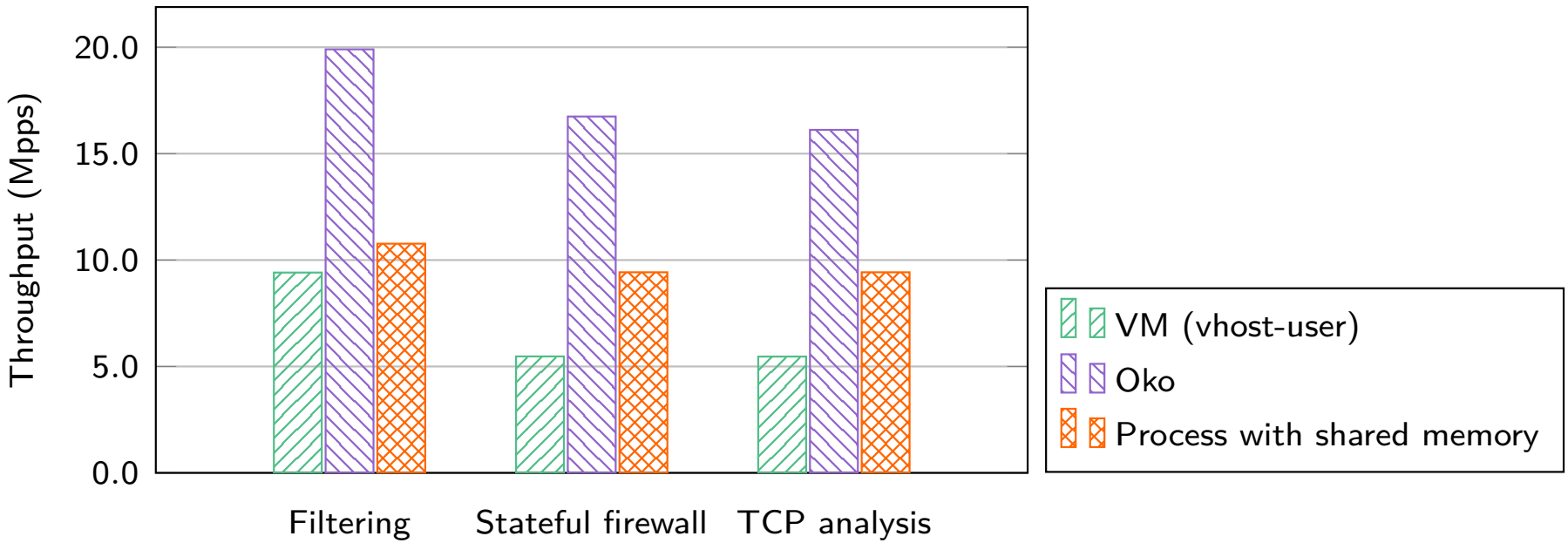


Figure: Throughput for different packet processing setups

Evaluations

Why is Oko faster than zero-copy process setup?

- No overhead from packet transfers across cores
- Better use of CPU caches
  - With process setup, packets need to be brought in L1 cache

Conclusion

- Oko: extensible Open vSwitch through BPF actions
- Best of code-based and flow table-based pipelines
- Started upstream of work on ubpf
  - Looking for help to review!
  - `https://github.com/iovisor/ubpf`
- Patch set to OVS with BPF actions (and no need to recirculate)

Thank you for listening!