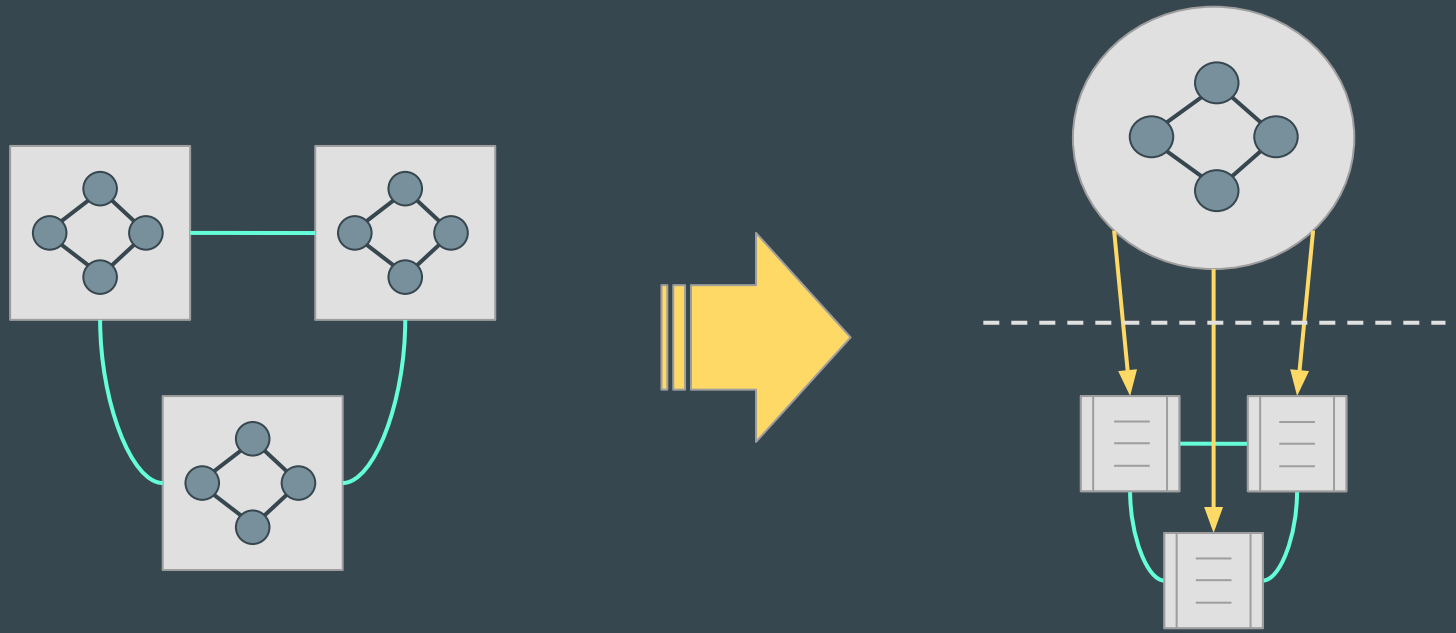
openvswitch.ko minus Open vSwitch

...

Joe Stringer, VMware

http://garfieldminusgarfield.net/post/26843739

Software-Defined Networking



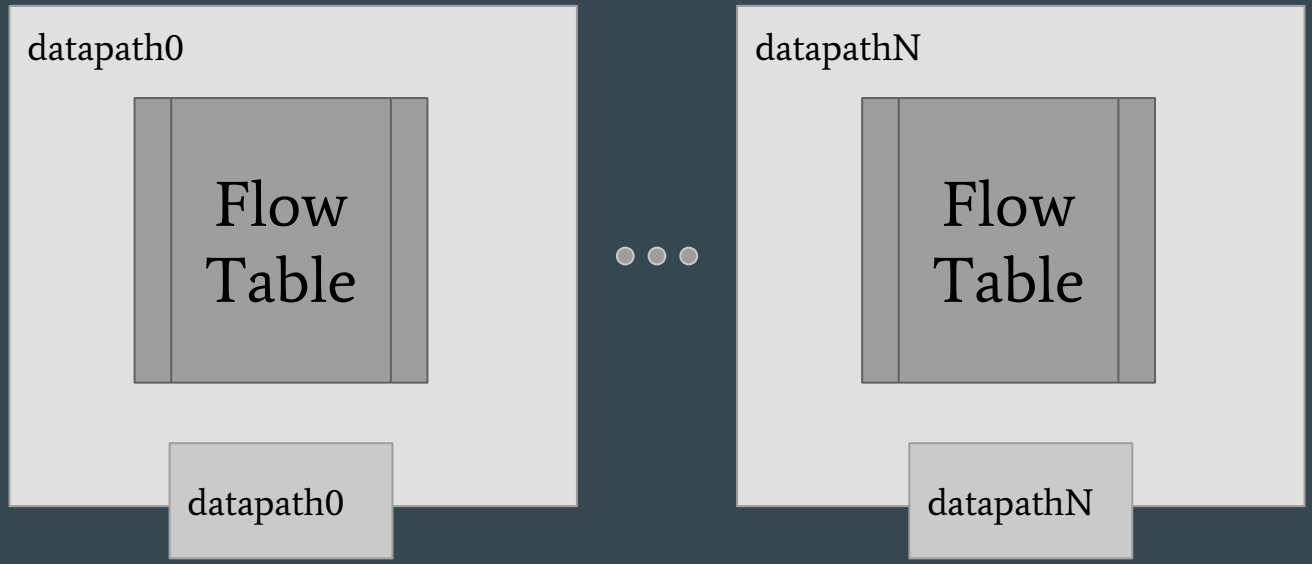
Flows

- Classify a set of packets that have some common criteria
- Not all flows are created equal
- Granularity => Power
  - => Performance?
- If possible, one lookup

How we described flow-based policy in Linux

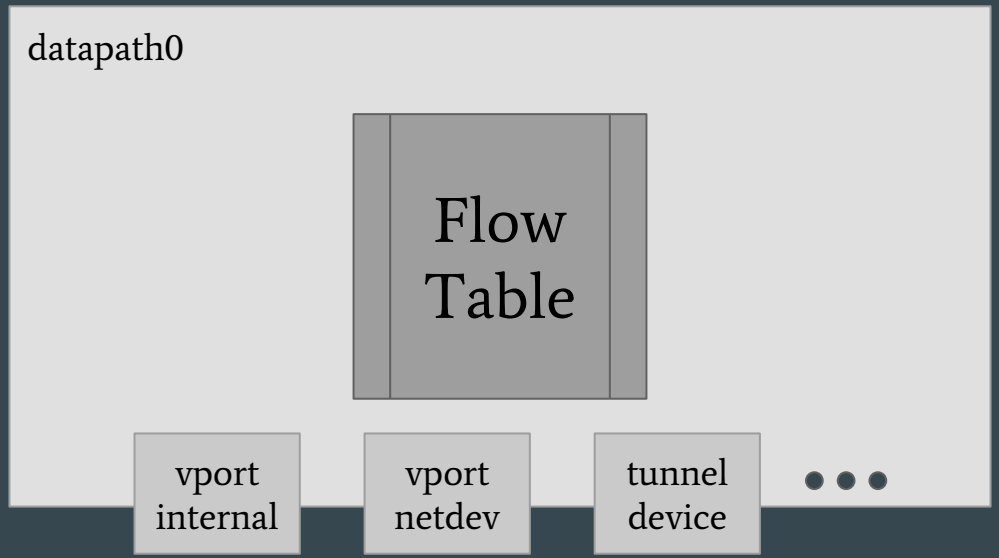
- Generic Netlink Families
- Shared flow table resource (**datapath**)
  - Need a bounding box for which set of flows apply
- Associate rx/tx **ports**
- Define the **flow**
  - Packet fields, metadata that can be matched on
- Describe how to handle **packets** when flow table empty

Datapath family



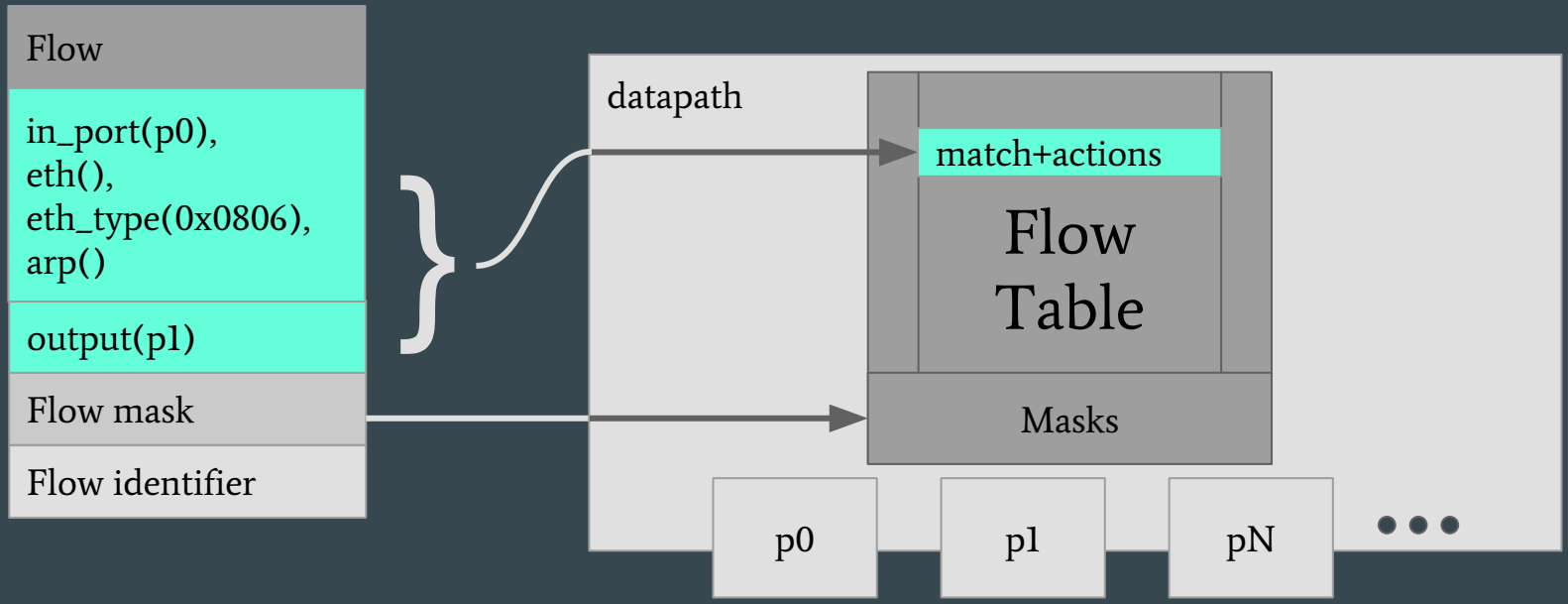
ovs-dpctl add-dp datapath0

Virtual port (vport) family



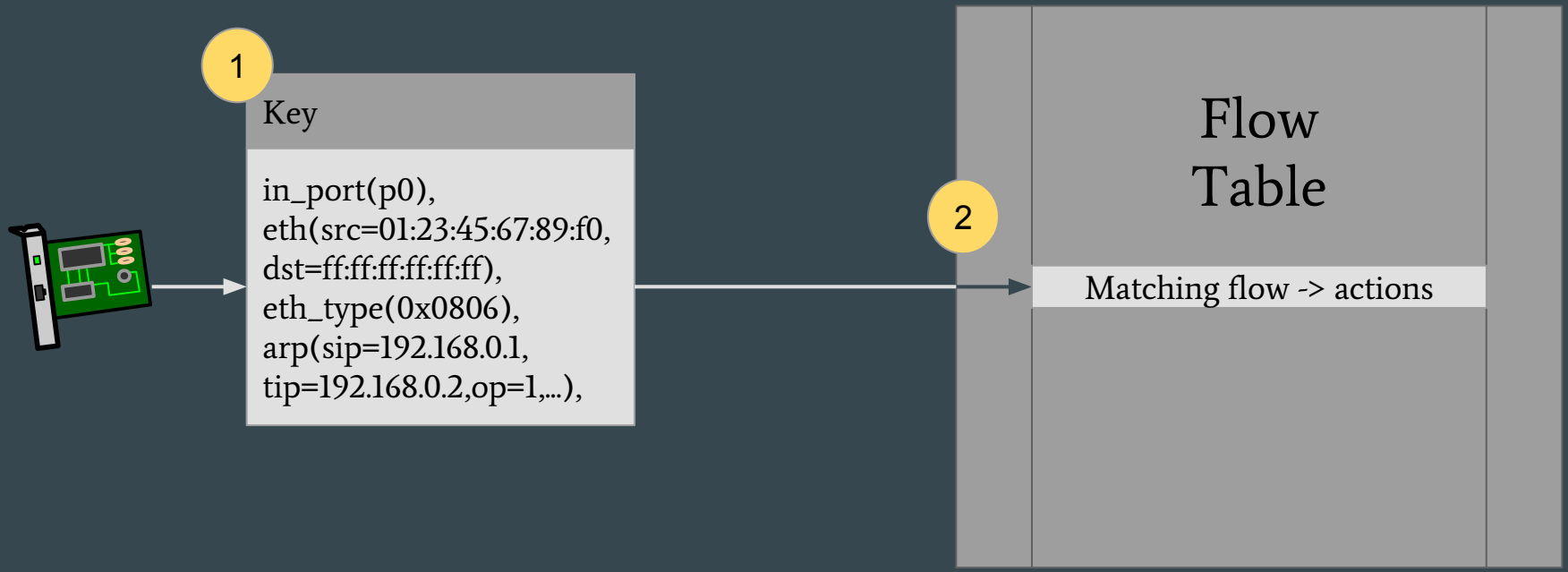
ovs-dpctl add-if datapath0 <netdev>

Flow family



ovs-dpctl add-flow datapath0 “in_port(0),eth(),eth_type(0x0806),arp()”, 1

Flow family: lookup hit

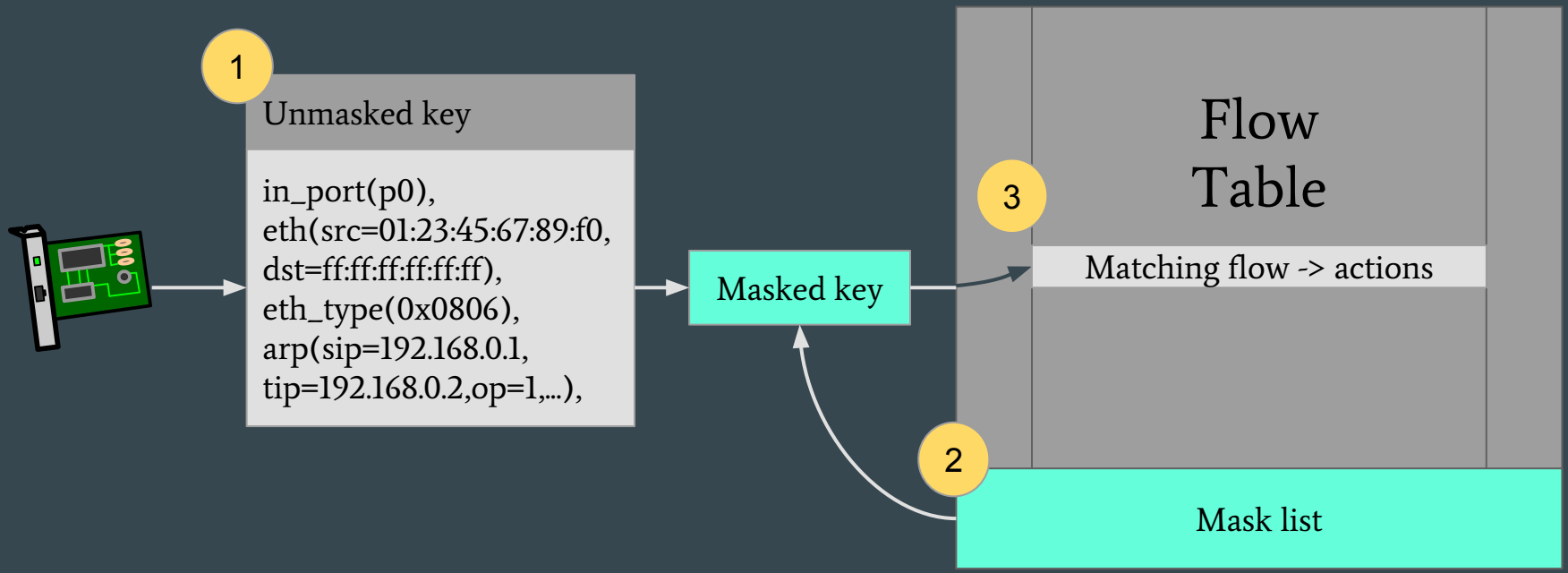


Masked tuple matching (megaflow)

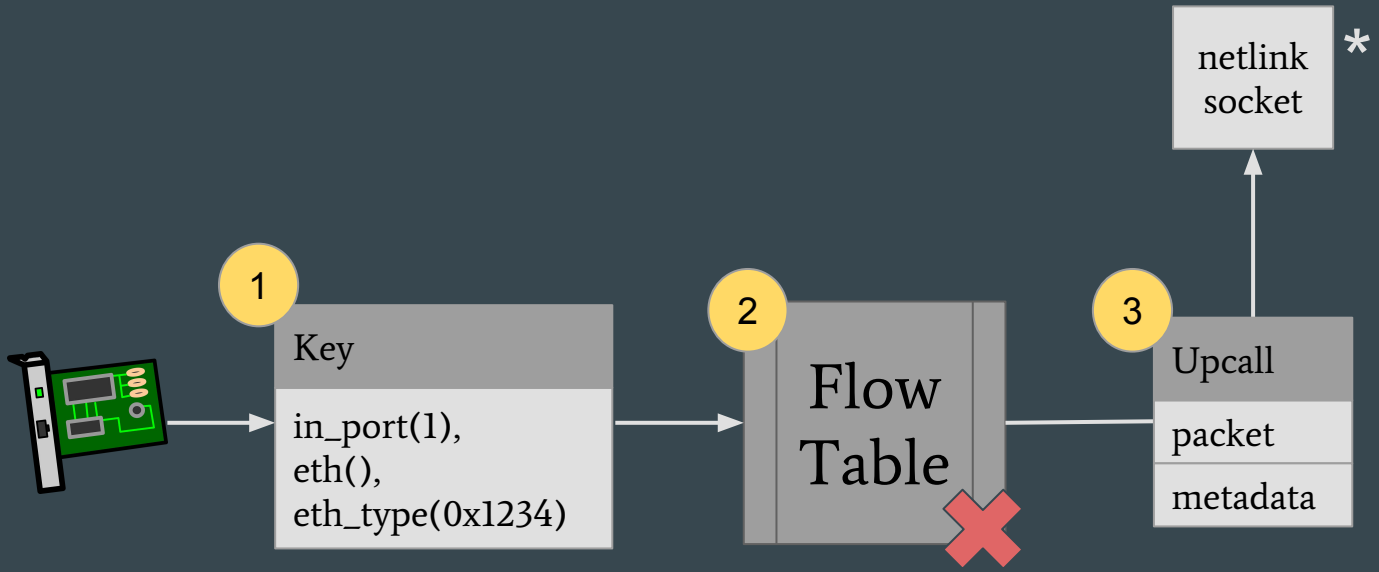
eth(src=x,dst=y),ip(dst=1.2.3.0)
eth(src=x,dst=y),ip(dst=1.2.3.1)
eth(src=x,dst=y),ip(dst=1.2.3.2)
eth(src=x,dst=y),ip(dst=1.2.3.3)
eth(src=x,dst=y),ip(dst=1.2.3.4)
eth(src=x,dst=y),ip(dst=1.2.3.5)
eth(src=x,dst=y),ip(dst=1.2.3.6)
eth(src=x,dst=y),ip(dst=1.2.3.7)

eth(src=x/ff:ff:ff:ff:ff:ff,dst=y/ff:ff:ff:ff:ff:ff),
ip(dst=1.2.3.0/255.255.255.248)

Flow family: lookup hit (megaflow)

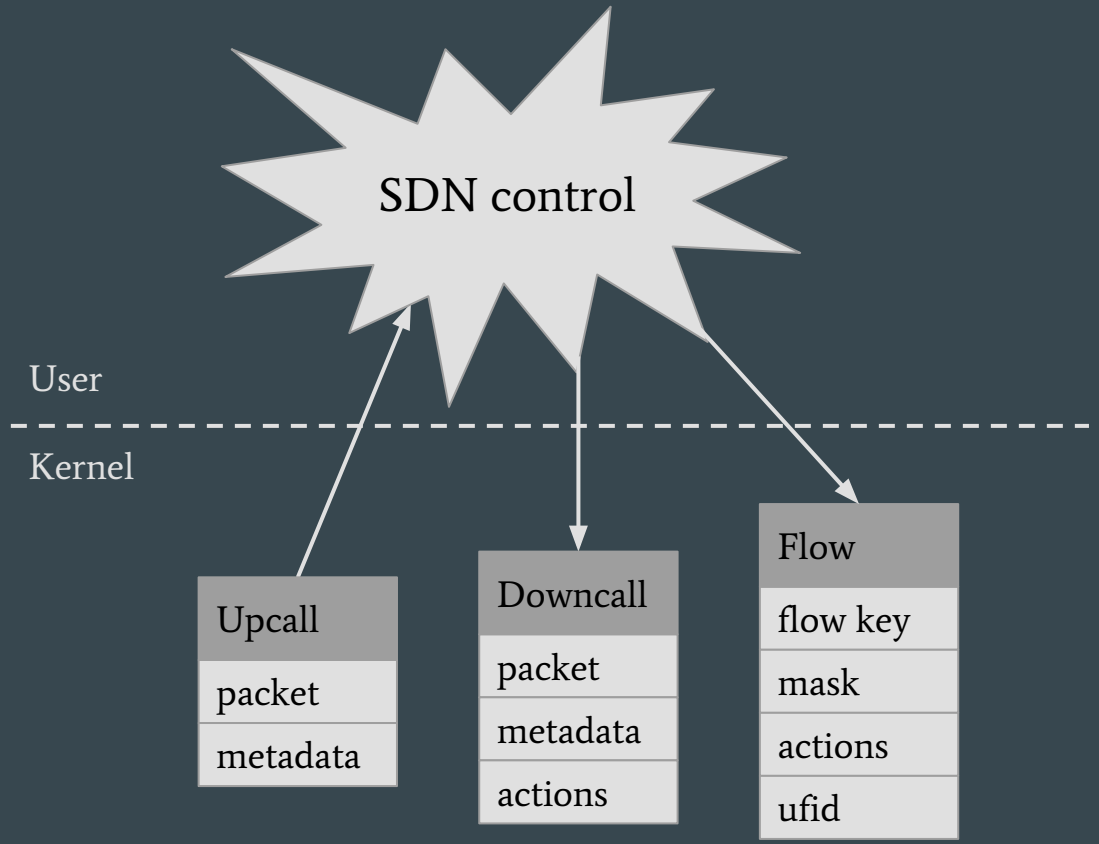


Flow family: Lookup miss

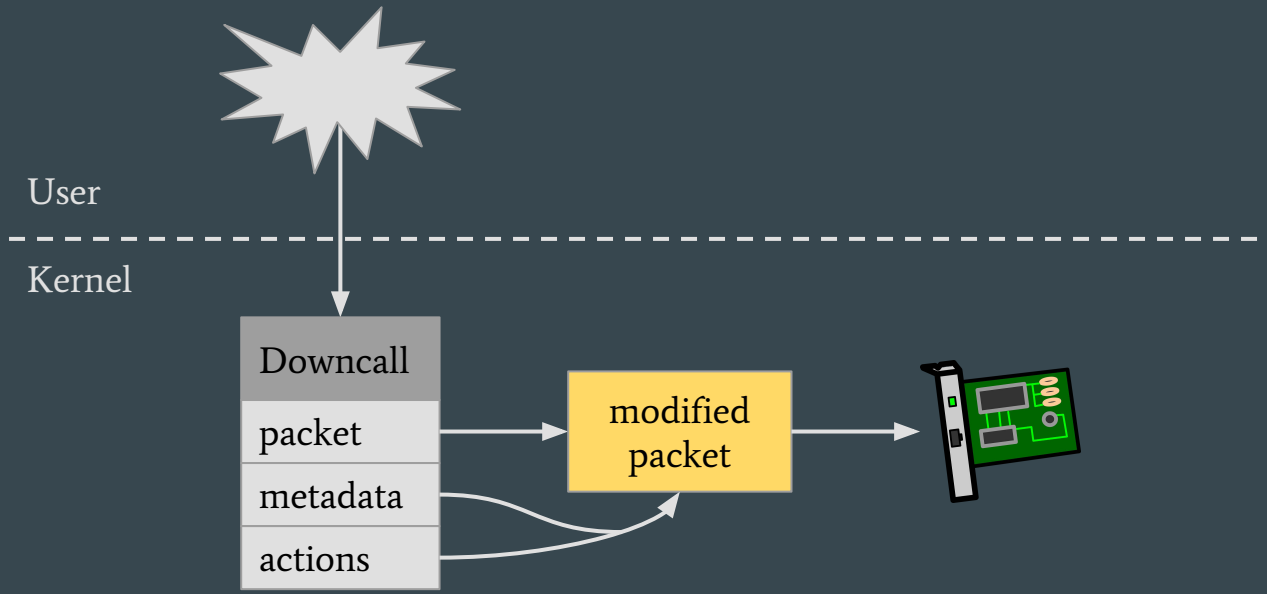


* netlink socket may be set to ‘0’, indicating default drop

Packet family: userspace upcall



Packet family: Execute



OVS Netlink API Summary

- Datapath family
  - Shared flow table
  - Access to stack
  - Place to hang ports
- Virtual port (vport) family
  - Access for rx/tx with the datapath
- Flow family
  - Describe forwarding behavior
- Packet family
  - Handle packet+metadata to/from userspace

Notable Improvements

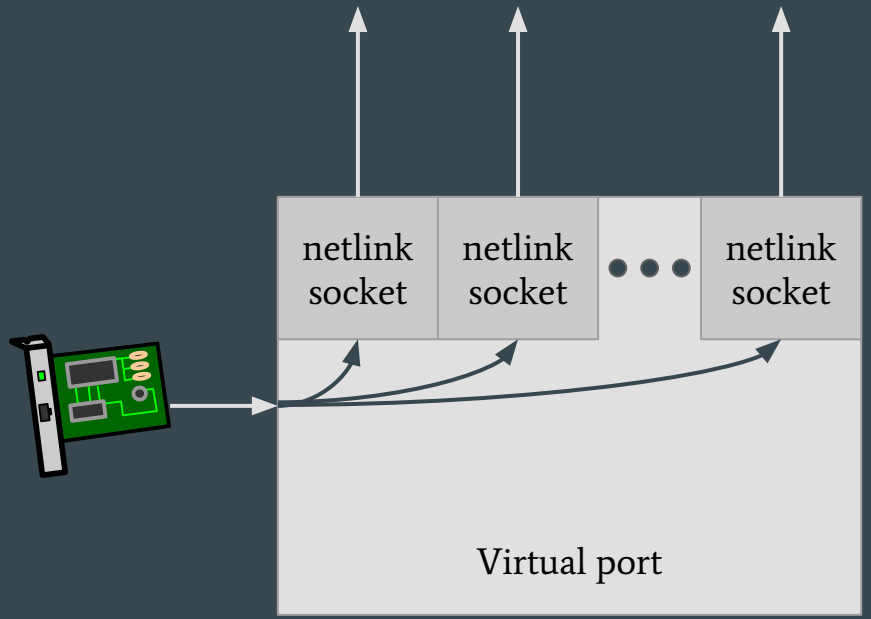
- Megaflows
- Traffic Isolation
- NetFilter integration
- Recirculation

Megaflows

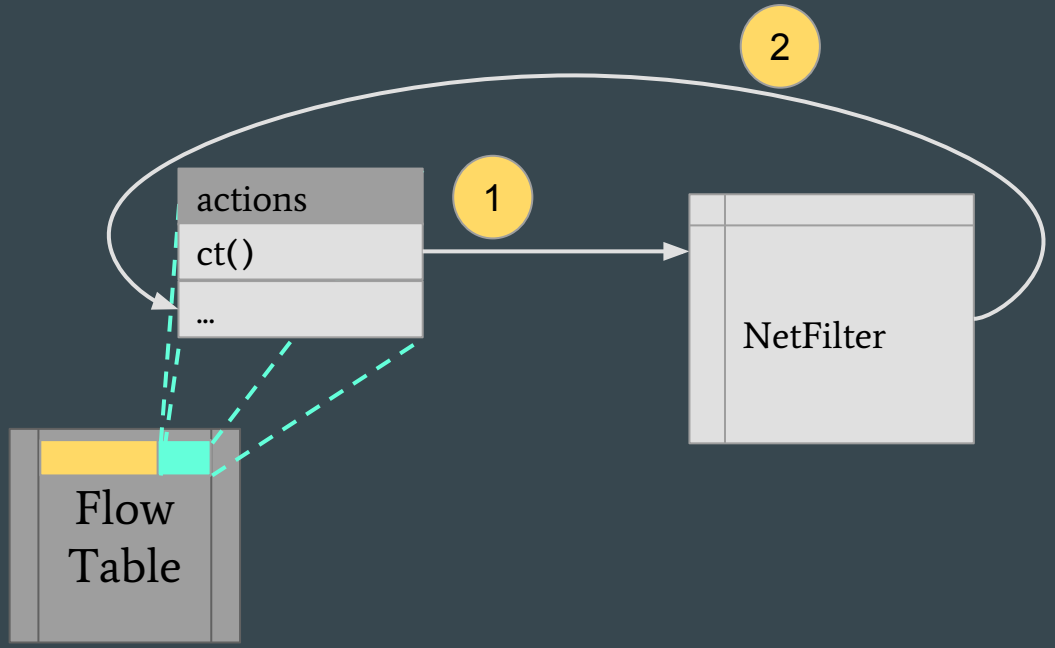
Optimizations	Ktps (TCP_CRR)	Flows	Masks	CPU % (user / kernel)
Megaflows disabled	37	1,051,884	1	45 / 40
No optimizations	56	905,758	3	37 / 40
With priority sorting	57	785,124	4	39 / 45
With prefix tracking	95	13	10	0 / 15
With staged lookup	115	14	13	0 / 15
All optimizations	117	15	14	0 / 20

From “The Design and Implementation of OVS”, Ben Pfaff et al., NSDI ’15

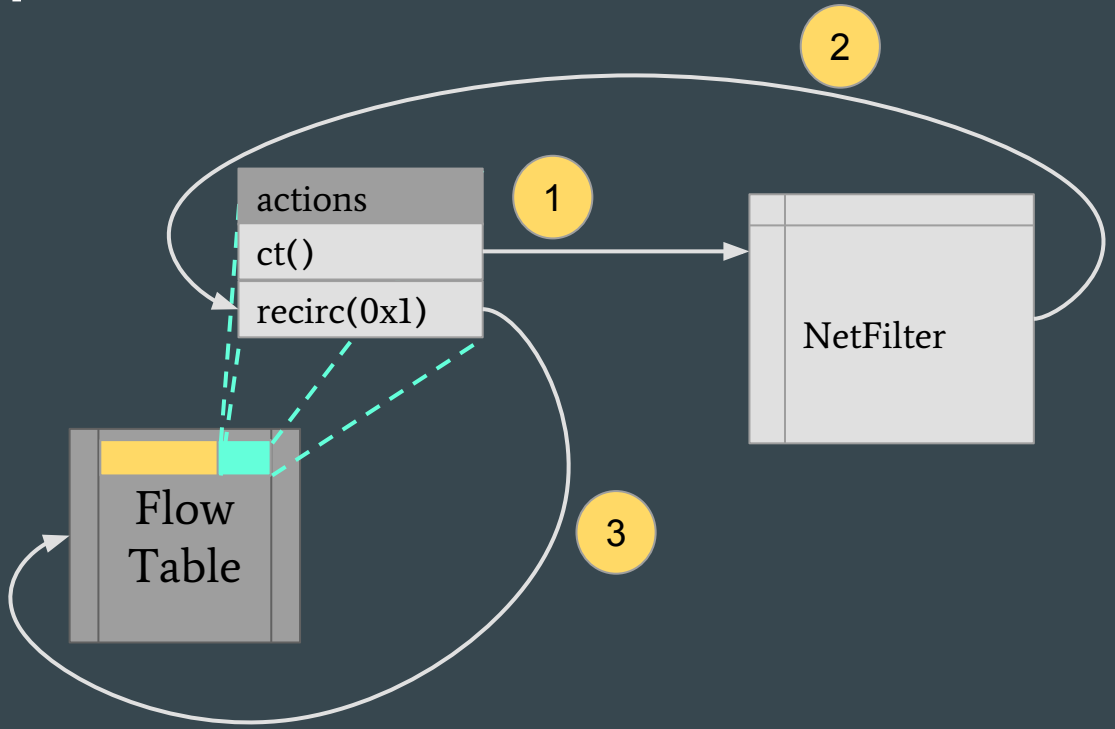
Notable improvements: Upcall hashing



Notable improvements: conntrack



Notable improvements: recirculate



openvswitch.ko

Kernel API users

- CLI tools
- Open vSwitch (ovs-vswitchd)
- MidoNet
- Weave Net
- Indigo Virtual Switch

CLI tools - datapath / vport

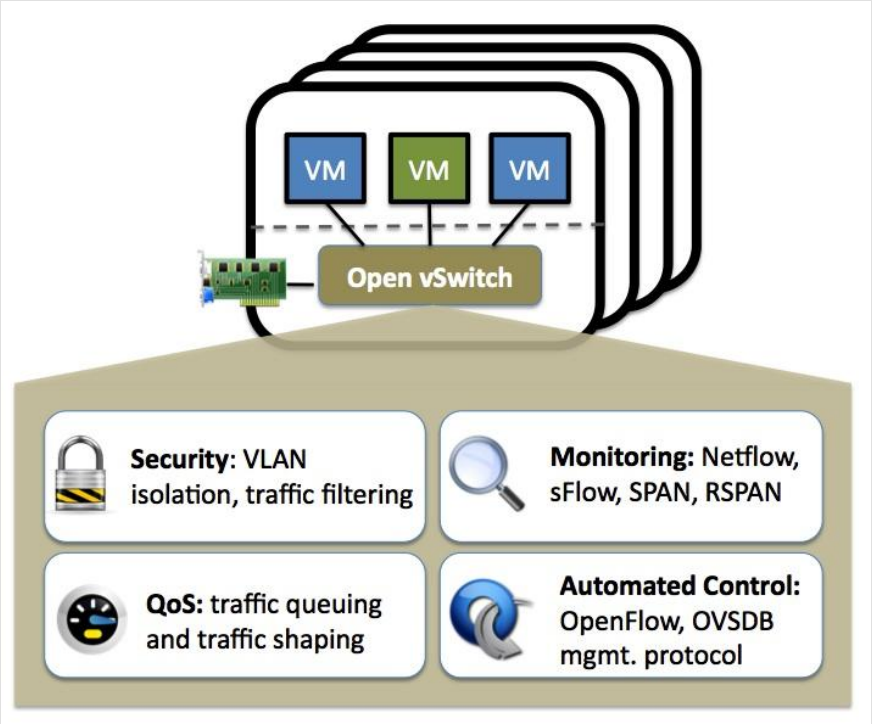
```
# modprobe openvswitch
# ovs-dpctl add-dp myDP
# ip li add dev dummy0 type dummy
# ovs-dpctl add-if myDP dummy0
# ip li add dev dummy1 type dummy
# ovs-dpctl add-if myDP dummy1
```

```
# ovs-dpctl show
system@myDP:
        lookups: hit:0 missed:177 lost:177
        flows: 0
        masks: hit:0 total:0 hit/pkt:0.00
        port 0: myDP (internal)
        port 1: dummy0
        port 2: dummy1
```

CLI tools - flow

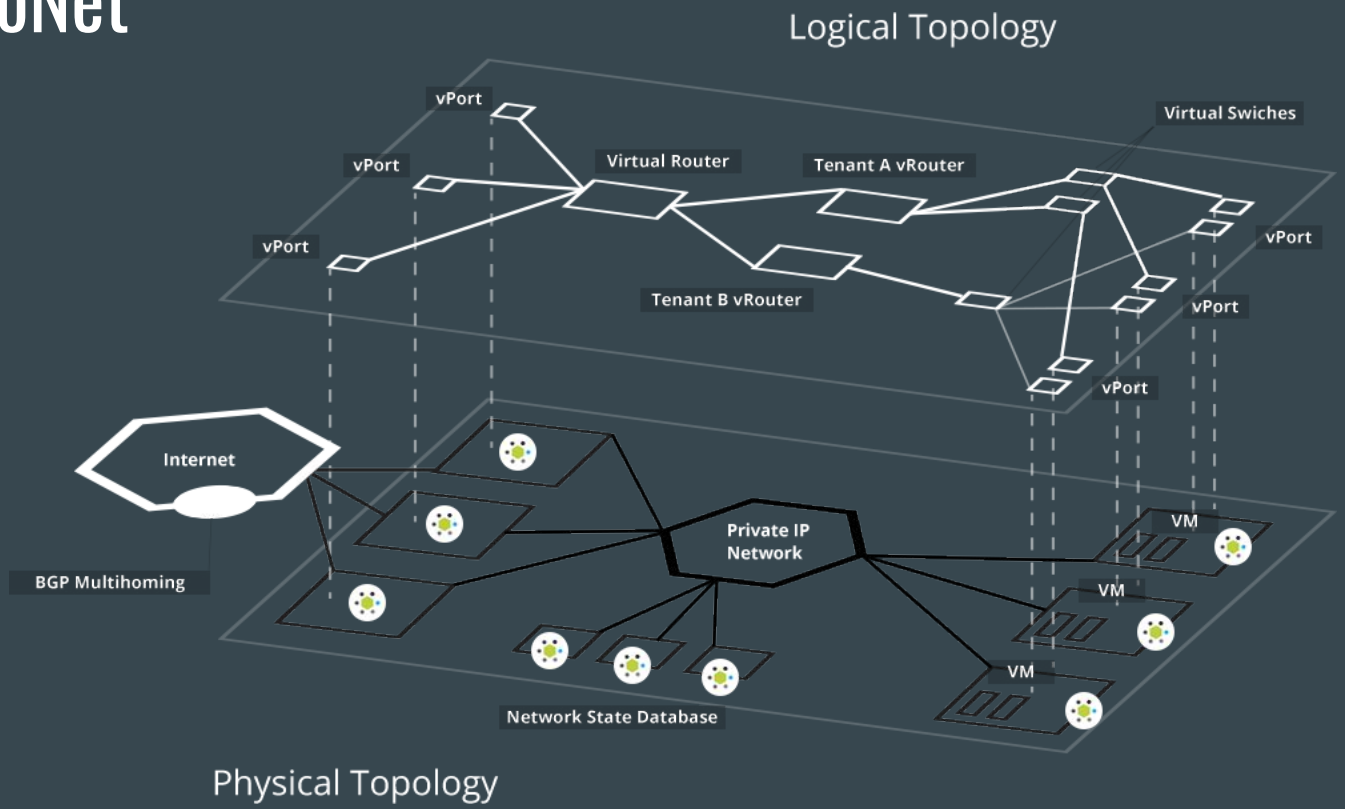
```
# ovs-dpctl add-flow "in_port(1),eth(),eth_type(0x806),arp()" 2
# ovs-dpctl add-flow "in_port(2),eth(),eth_type(0x806),arp()" 1
# ovs-dpctl add-flow "in_port(1),eth(),eth_type(0x800),ipv4(proto=1),icmp()" 2
# ovs-dpctl add-flow "in_port(2),eth(),eth_type(0x800),ipv4(proto=1),icmp()" 1
# ovs-dpctl dump-flows
in_port(2),eth_type(0x0806), packets:0, bytes:0, used:never, actions:1
in_port(1),eth_type(0x0806), packets:0, bytes:0, used:never, actions:2
in_port(2),eth_type(0x0800),ipv4(proto=1), packets:0, bytes:0, used:never, actions:1
in_port(1),eth_type(0x0800),ipv4(proto=1), packets:0, bytes:0, used:never, actions:2
```

Open vSwitch Daemon



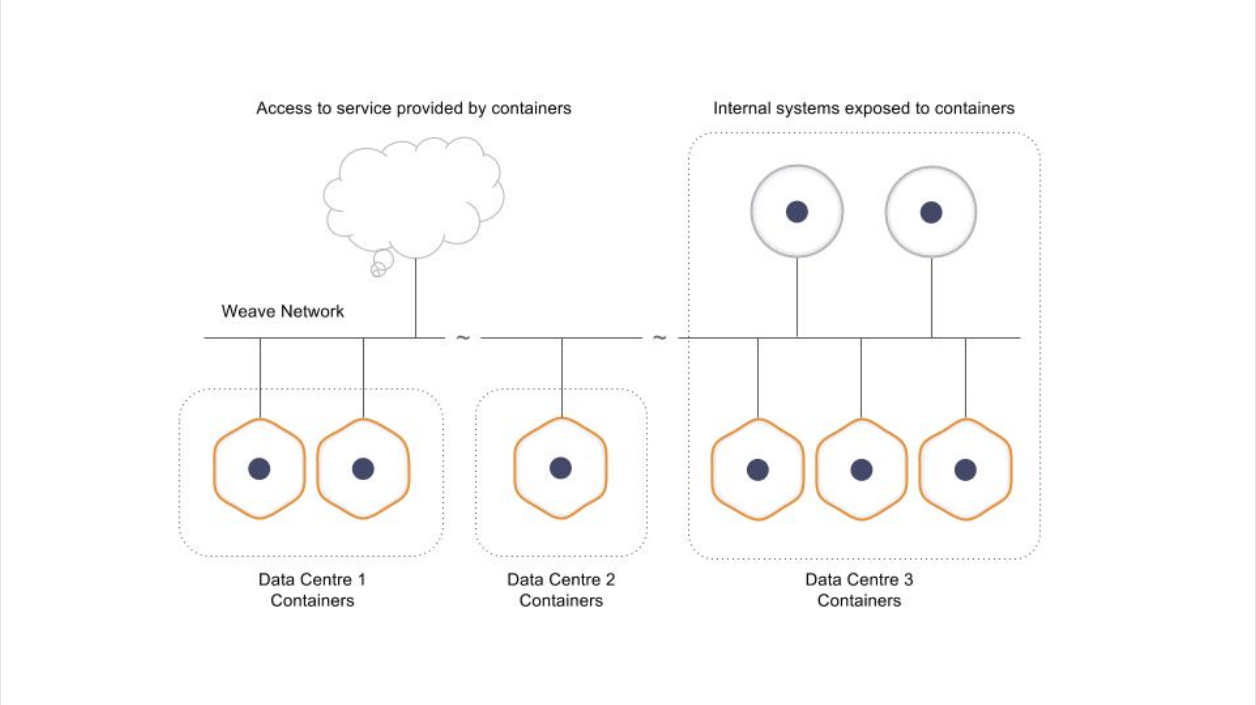
http://openvswitch.org/assets/featured-image.jpg

MidoNet



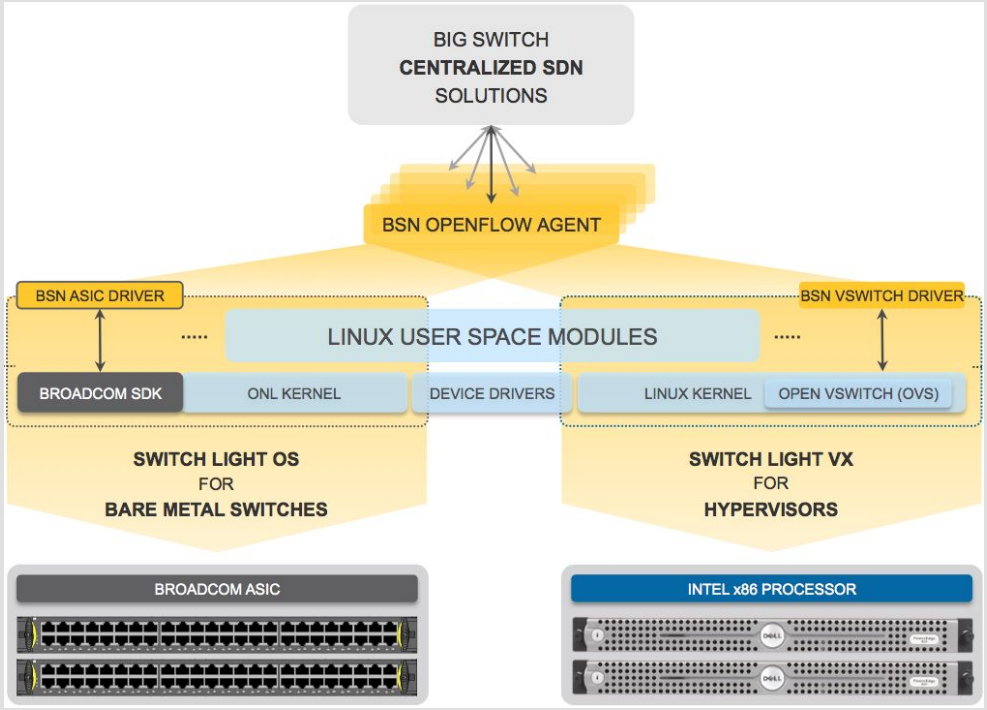
https://www.midonet.org/i/graphic.png

Weave Net



https://www.weave.works/wp-content/uploads/d989f137a913d15c6ab2afe14149d8acfd180db3.png

Indigo Virtual Switch



http://www.bigswitch.com/sites/default/files/_/switch_light_archictecture.png

Common threads: integration

- Lightweight Tunneling
- Netfilter
- XFRM
- QoS
- Hardware offloads

Common threads: complexity

- Desired configuration is orders of magnitude more complex than kernel API
  - Dozens of tables
  - Thousands of priorities
- Compile hundreds of lookups into a single* lookup
  - Lower per-packet costs for complex pipelines

* or small integer when subsystem input is required

Summary

- SDN has driven openvswitch.ko development
  - logically centralized packet forwarding behaviour
- OVS Netlink API provides generally useful primitives
- Variety of users
  - OVS, MidoNet, WeaveNet, IVS
- Allows userspace to integrate with other kernel functionality
- Minimize kernel code complexity

http://garfieldminusgarfield.net/post/37998316

fin

...

joe@ovn.org