NetControl

Johanna Amann
johanna@icir.org

NetControl

Push rules to networking hard and software

Based on traffic observed by Bro

Simple to use but flexible API

Uses for NetControl

Traffic Shunting

Block attacks at network boundary

Redirecting high traffic flows to different interfaces

Quarantine hosts

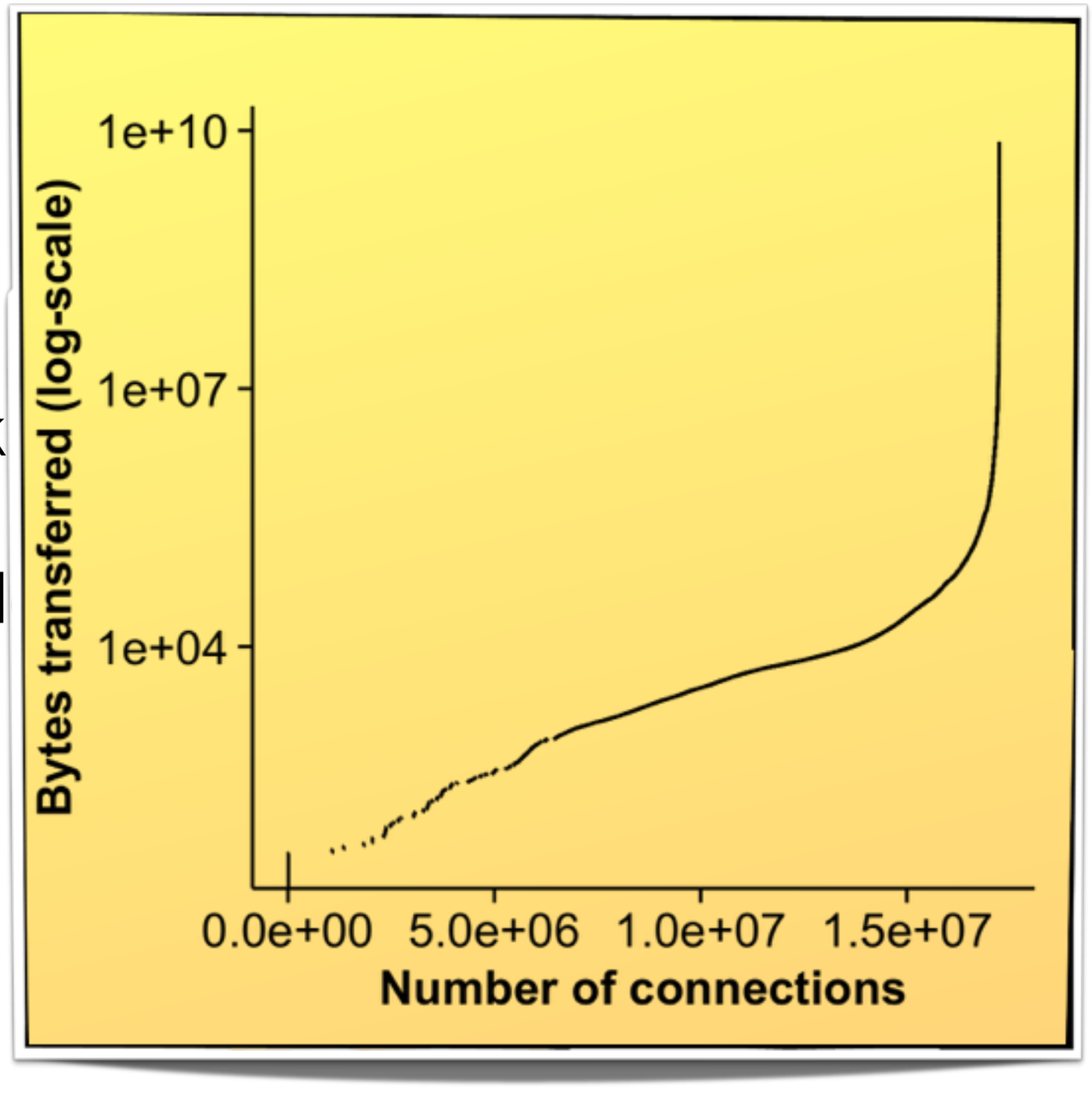
Uses for NetControl

Traffic Shunting

Block attacks at network

Redirecting high traffic fl

Quarantine hosts



Uses for NetControl

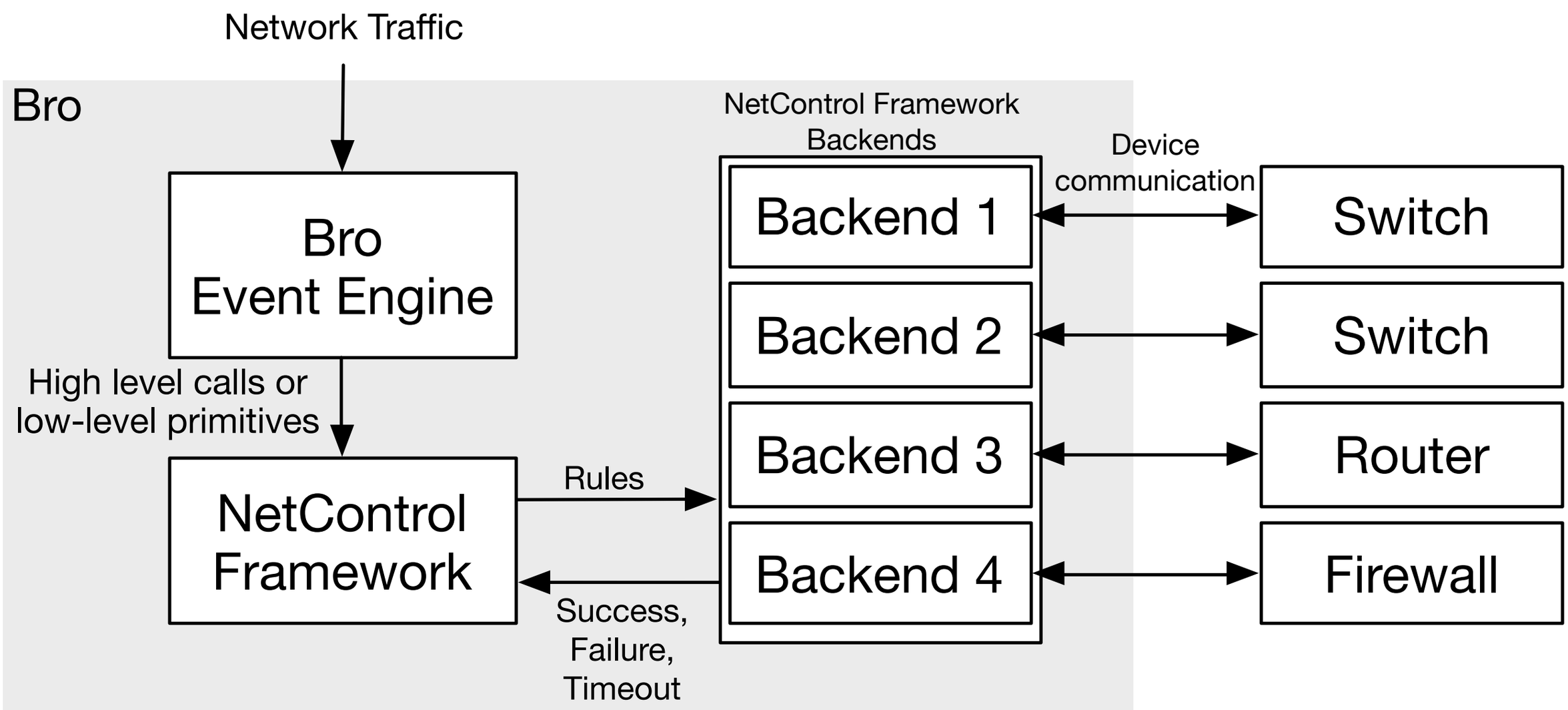
Traffic Shunting

Block attacks at network boundary

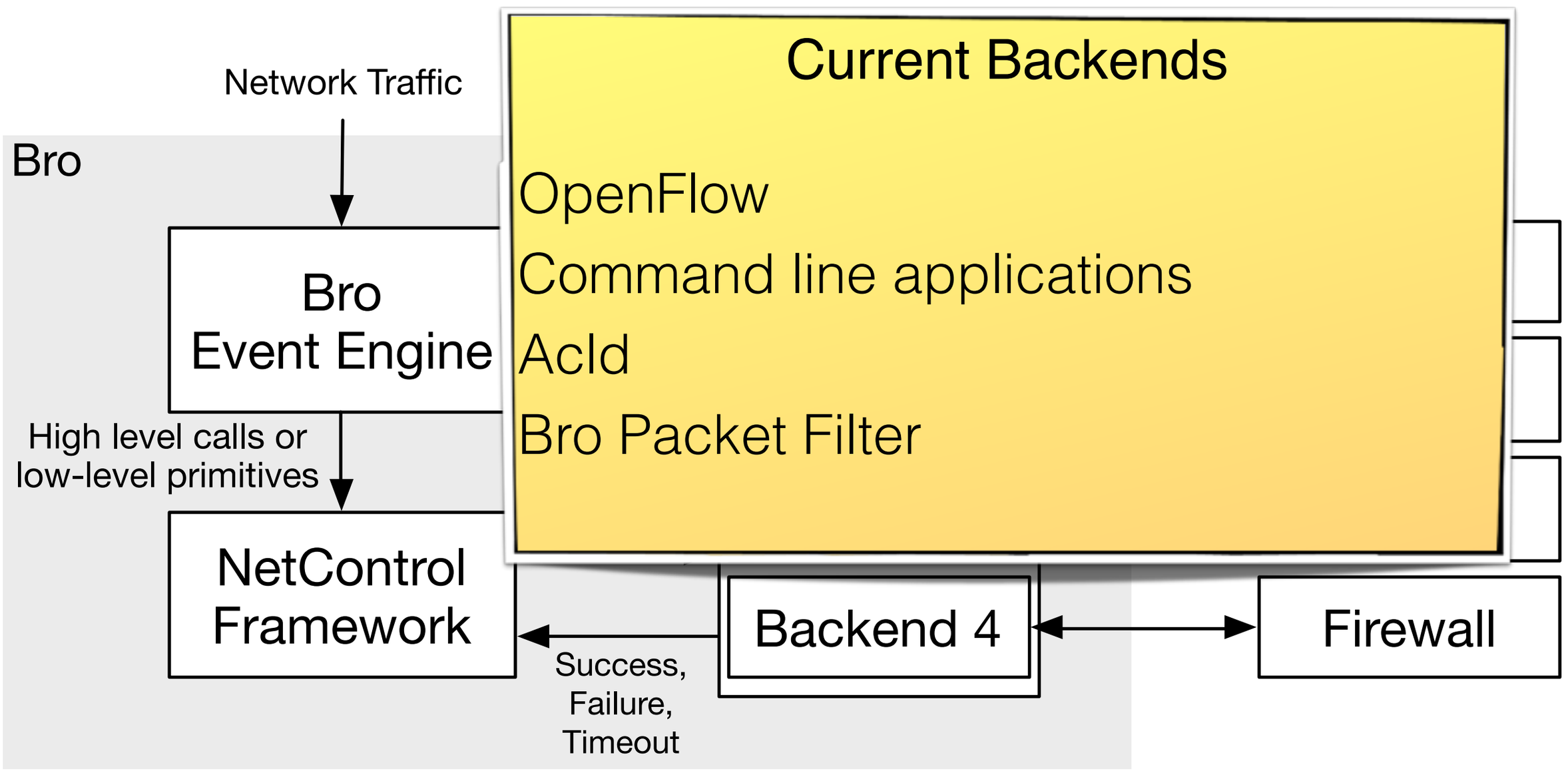
Redirecting high traffic flows to different interfaces

Quarantine hosts

Architecture



Architecture



Bro PacketFilter

`install_dst_addr_filter`: `function`	Installs a filter to drop packets destined to a given IP address with a certain probability if none of a given set of TCP flags are set.
`install_dst_net_filter`: `function`	Installs a filter to drop packets destined to a given subnet with a certain probability if none of a given set of TCP flags are set.
`install_src_addr_filter`: `function`	Installs a filter to drop packets from a given IP source address with a certain probability if none of a given set of TCP flags are set.
`install_src_net_filter`: `function`	Installs a filter to drop packets originating from a given subnet with a certain probability if none of a given set of TCP flags are set.

High level API

drop_connection (*connection*, *timeout*)

drop_address (*host*, *timeout*)

drop_address_catch_release (*host*)

shunt flow (*flow*, *timeout*)

quarantine (*infected host*, *dns host*, *q. server*, *timeout*)

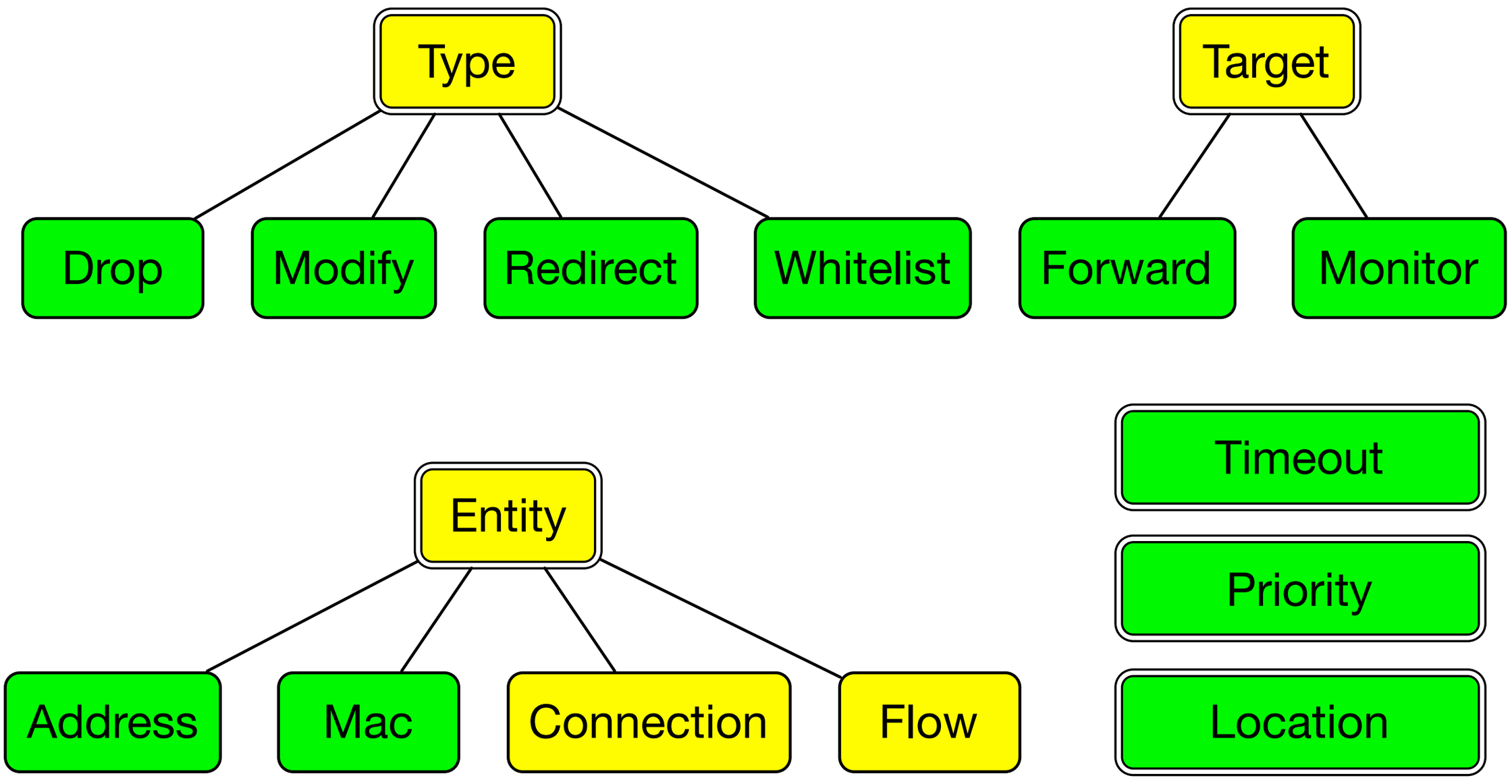
whitelist (*prefix*, *timeout*)

API Examples

```
event GridFTP::data_channel_detected(c: connection) {
    NetControl::shunt_flow(
      [$src_h=c$id$orig_h, $src_p=c$id$orig_p,
       $dst_h=c$id$resp_h, $resp_p=c$id$resp_p],
      1hr);
}
```

```
event log_notice(n: Notice::Info) {
  if ( n$note == Address_Scan || n$note == Port_Scan )
    NetControl::drop_address(n$src, 10min);
}
```

What do Rules look like?

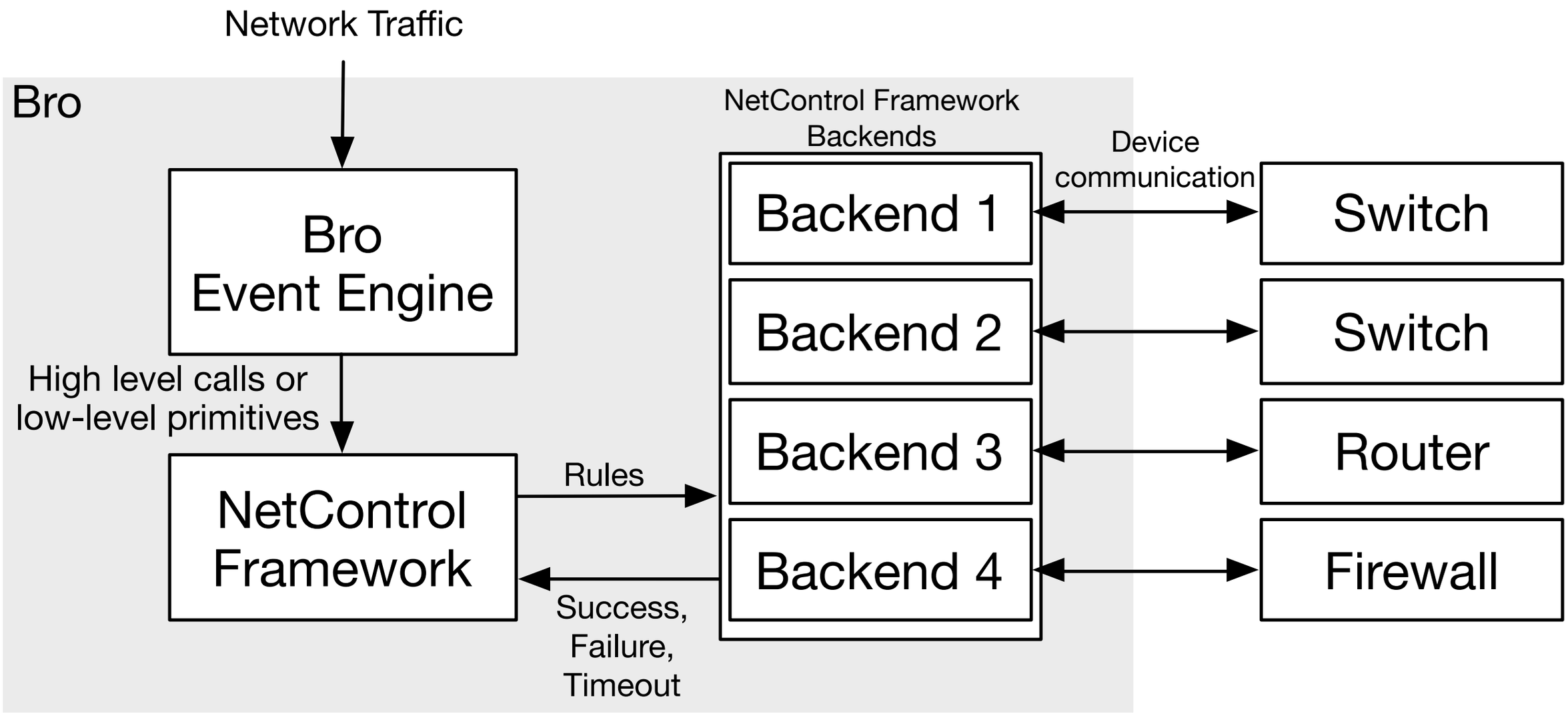


Example

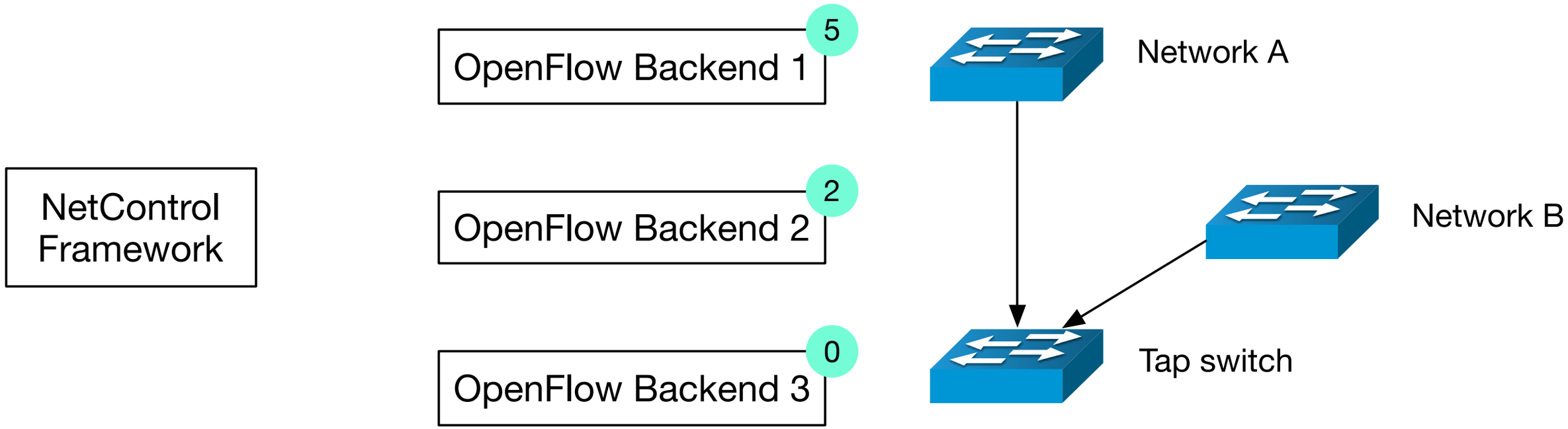
Rule(Type=Drop, Entity=Flow([5-tuple]), Target=Monitor)

```
function shunt_flow(f: flow_id, t: interval) : string {
  local flow = Flow(
      $src_h=addr_to_subnet(f$src_h), $src_p=f$src_p,
      $dst_h=addr_to_subnet(f$dst_h), $dst_p=f$dst_p
      );
  local e: Entity = [$ty=FLOW, $flow=flow];
  local r: Rule = [
      $ty=DROP, $target=MONITOR, $entity=e, $expire=t
      ];
  return add_rule(r);
}
```

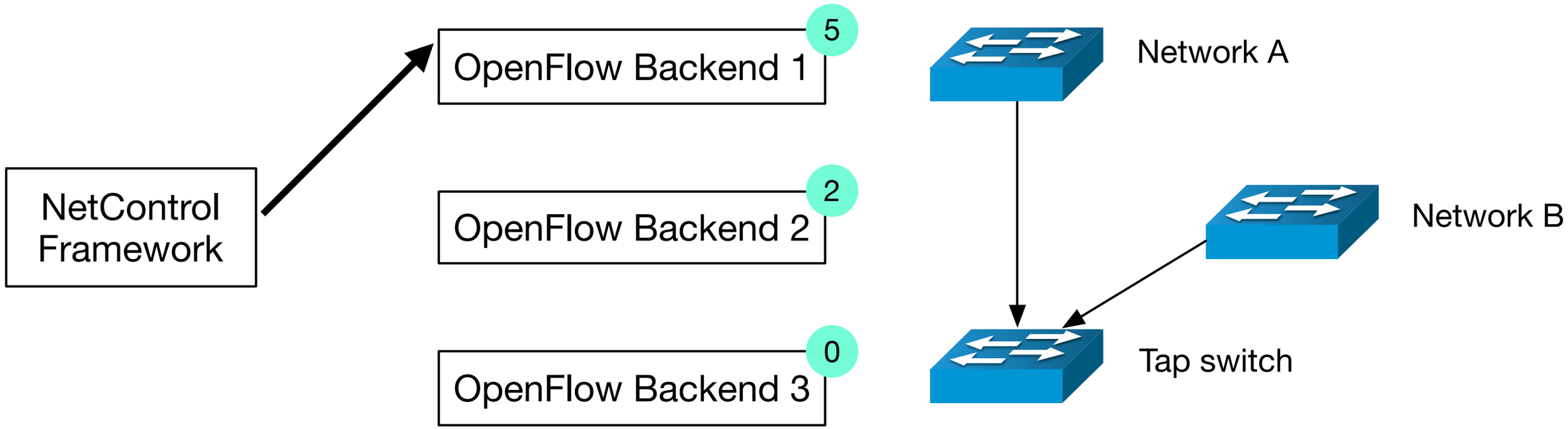
Choosing Backends



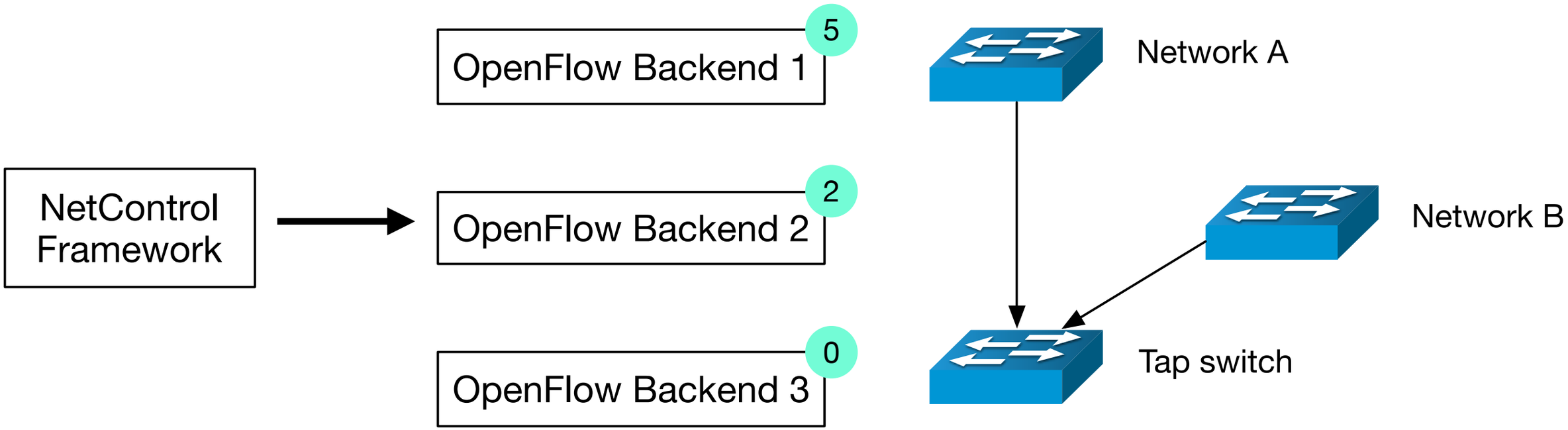
Choosing Backends



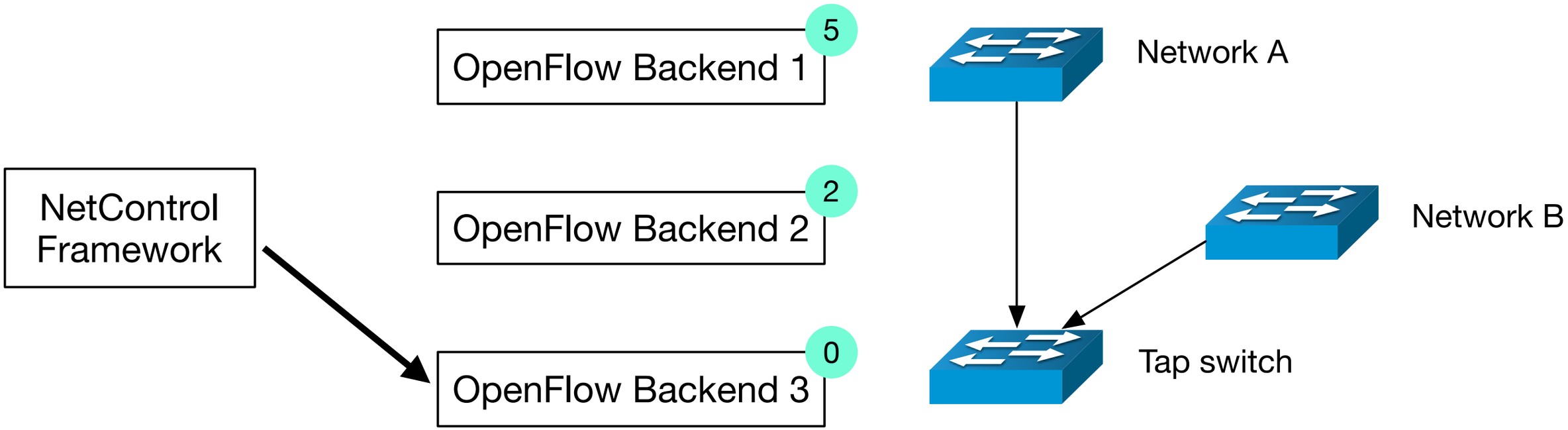
Choosing Backends



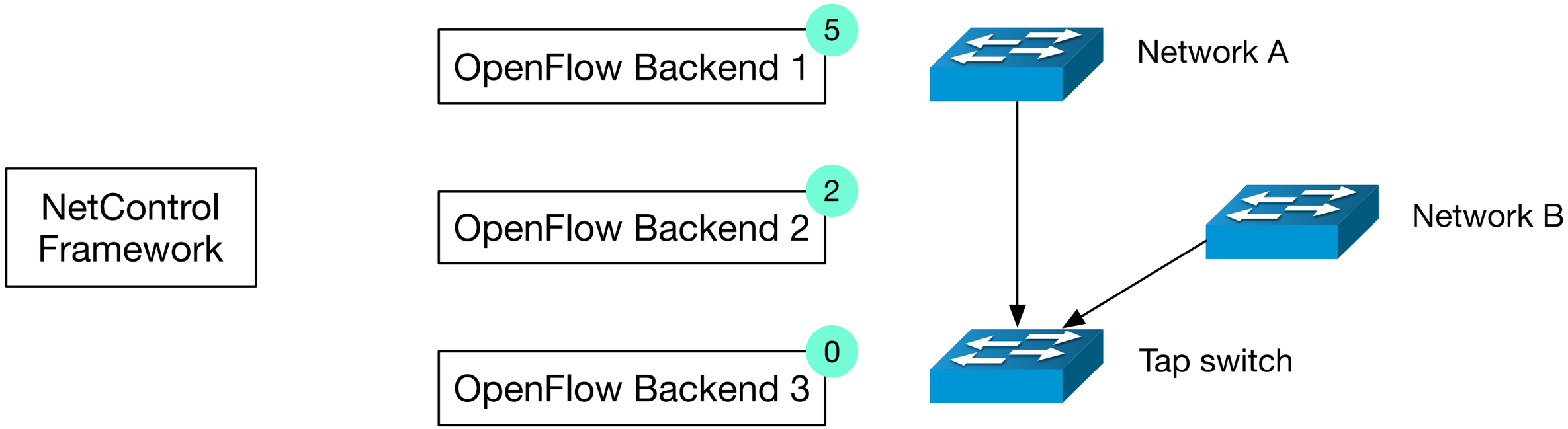
Choosing Backends



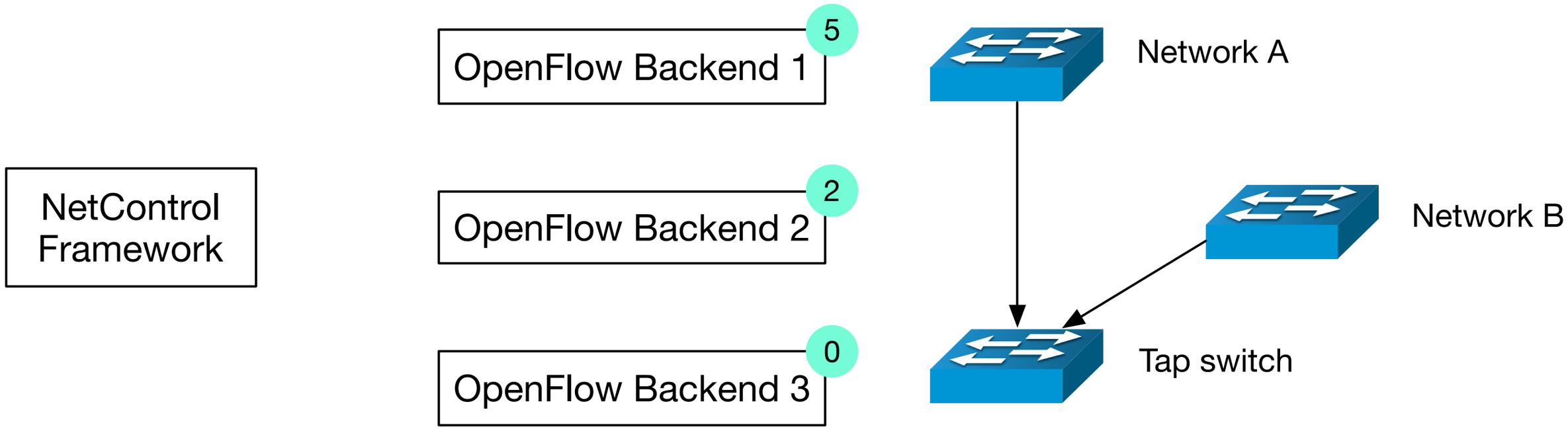
Choosing Backends



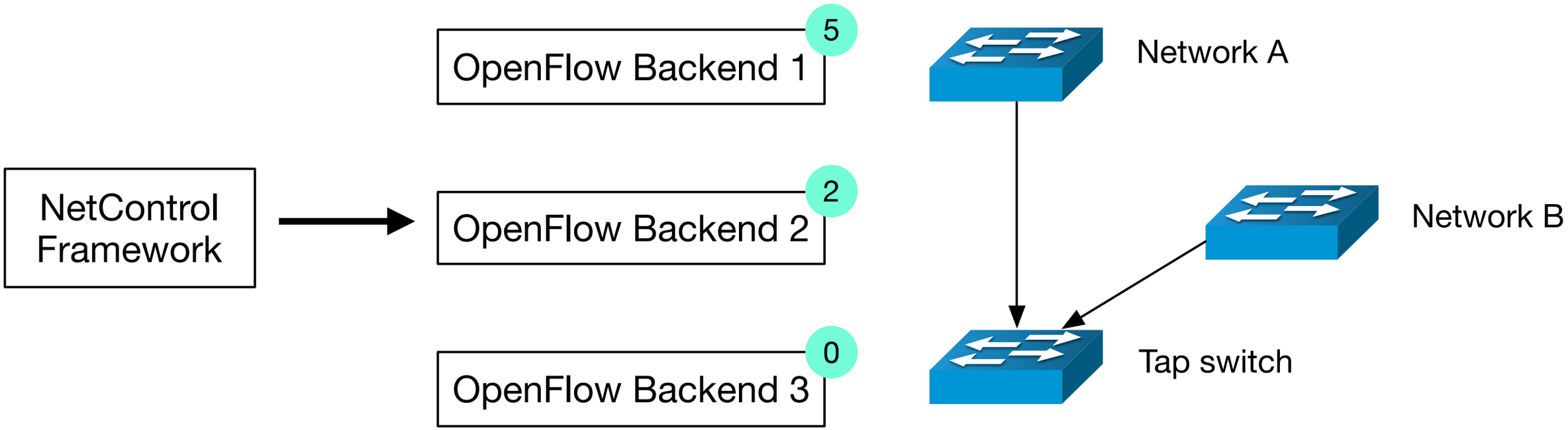
Choosing Backends



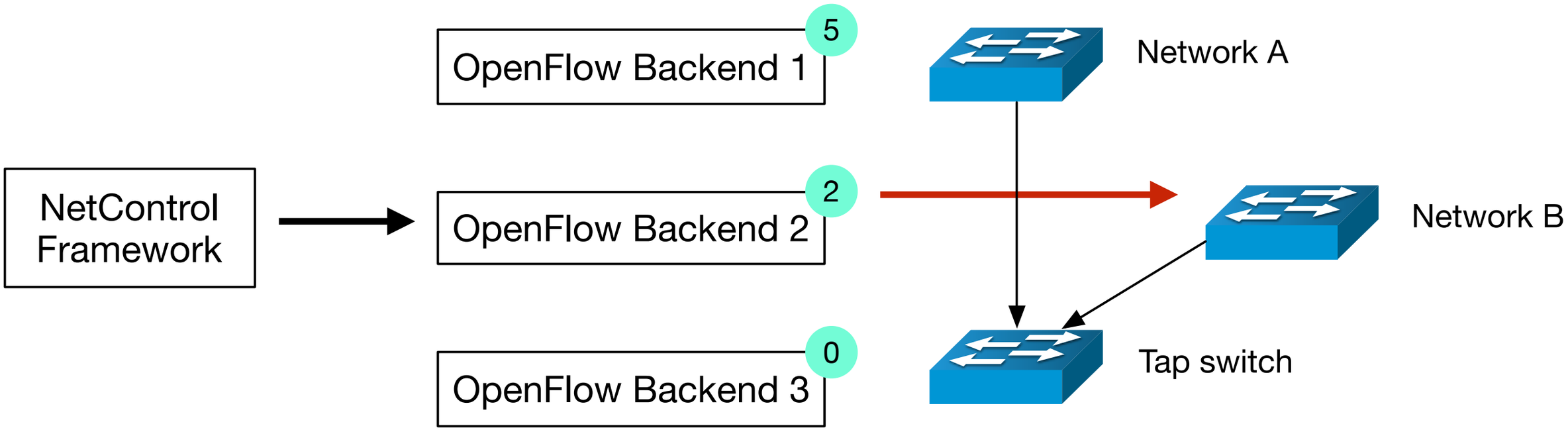
Choosing Backends



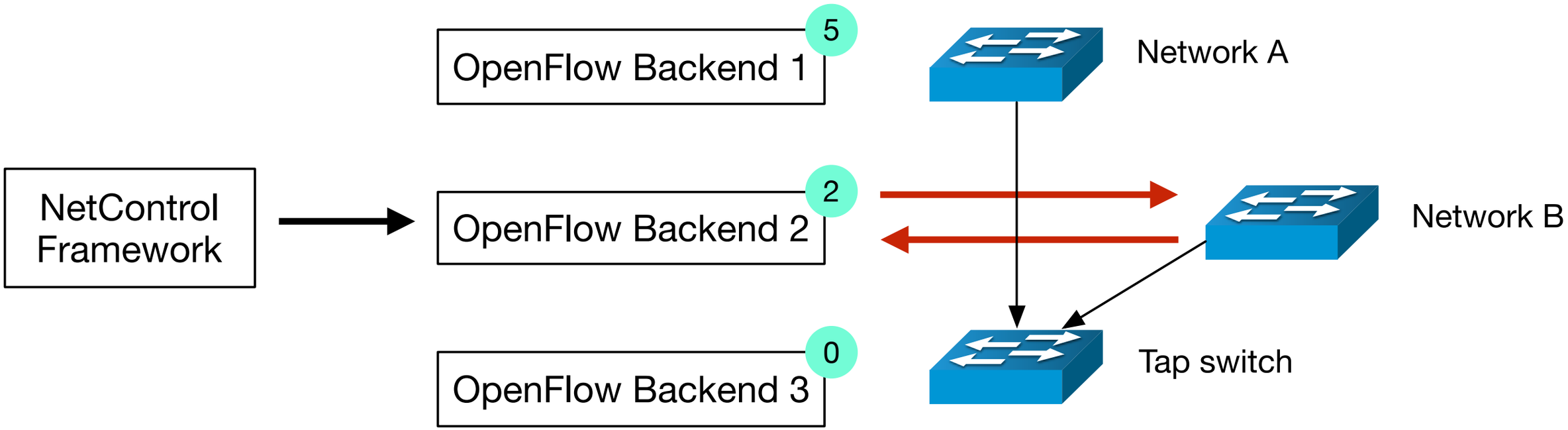
Choosing Backends



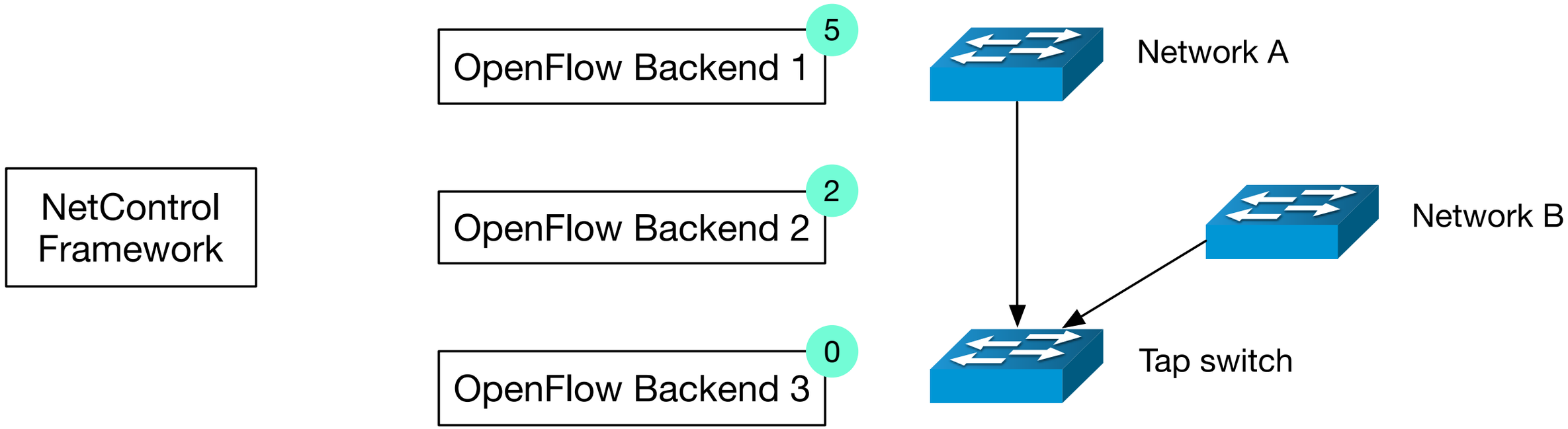
Choosing Backends



Choosing Backends



Choosing Backends



Adding Backends

```
local backend = NetControl::create_backend_Foo([...]);
NetControl::activate(backend, 10);
```

State management

Rules often only needed for limited time

NetControl supports timeouts

…but respects hard/software that don’t need them

OpenFlow

Open Specification

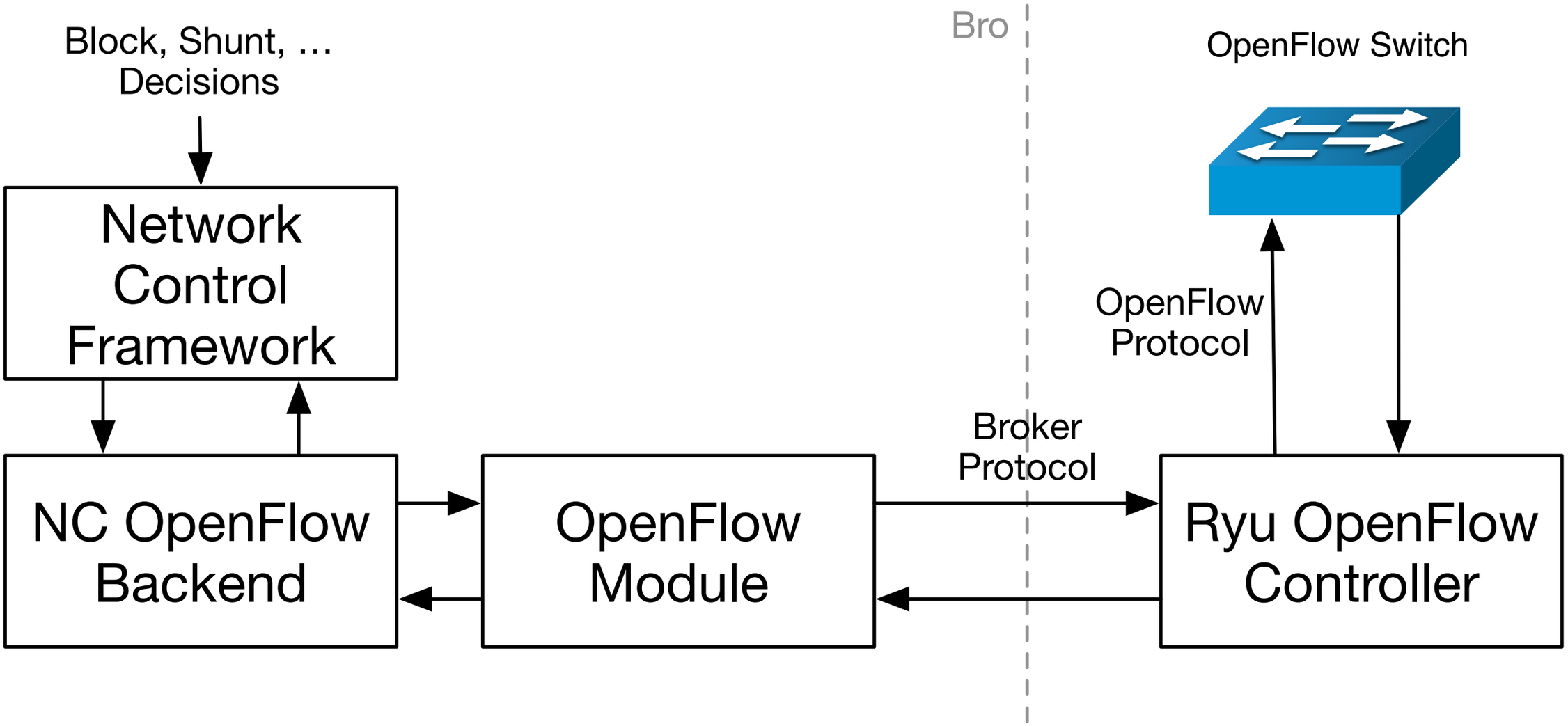
Allows Software to insert rules into switch flow tables

Match (and change) characteristics like

IPv4/6 addresses, ports, etc.

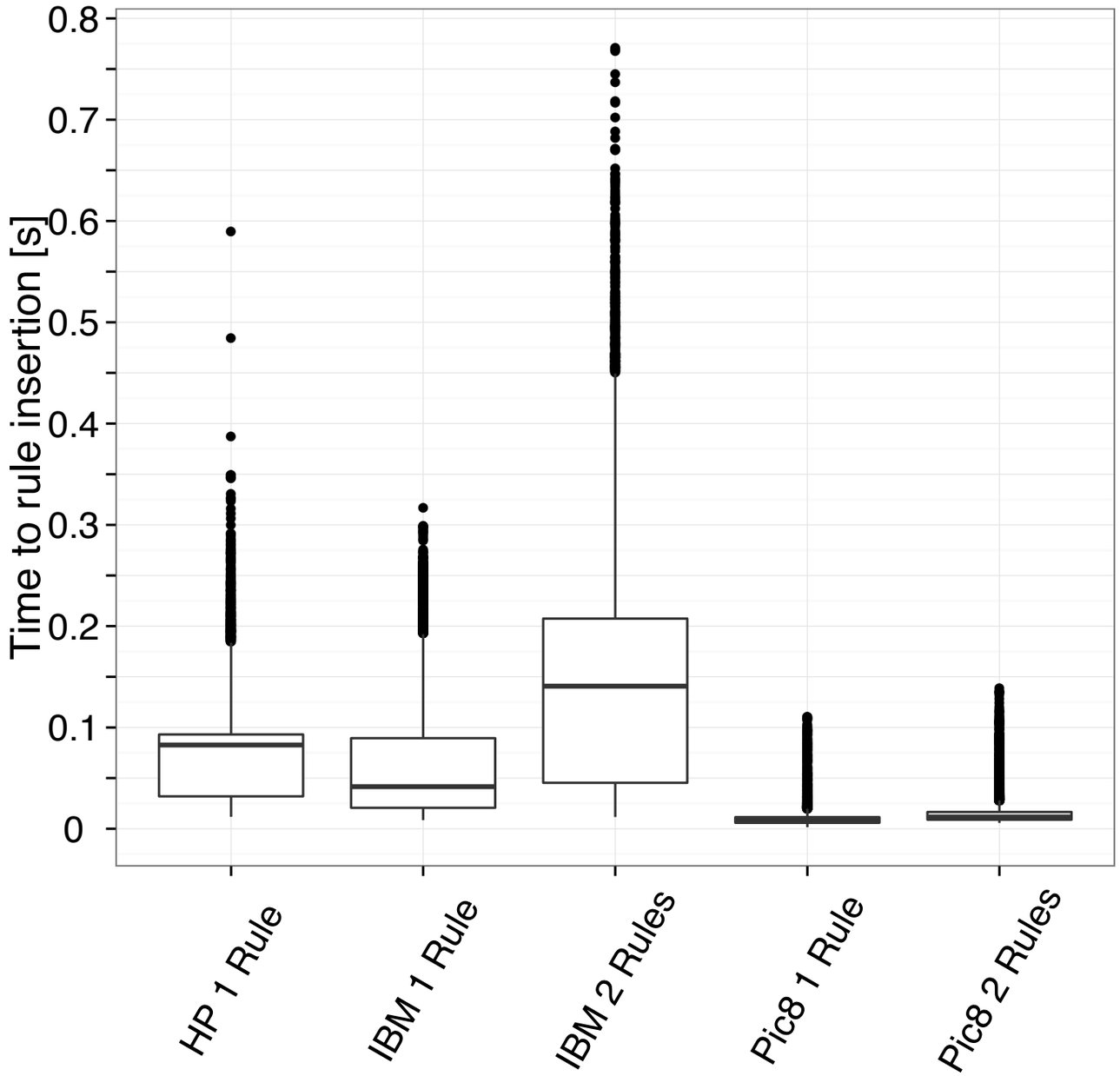
Vlans

NetControl & OpenFlow



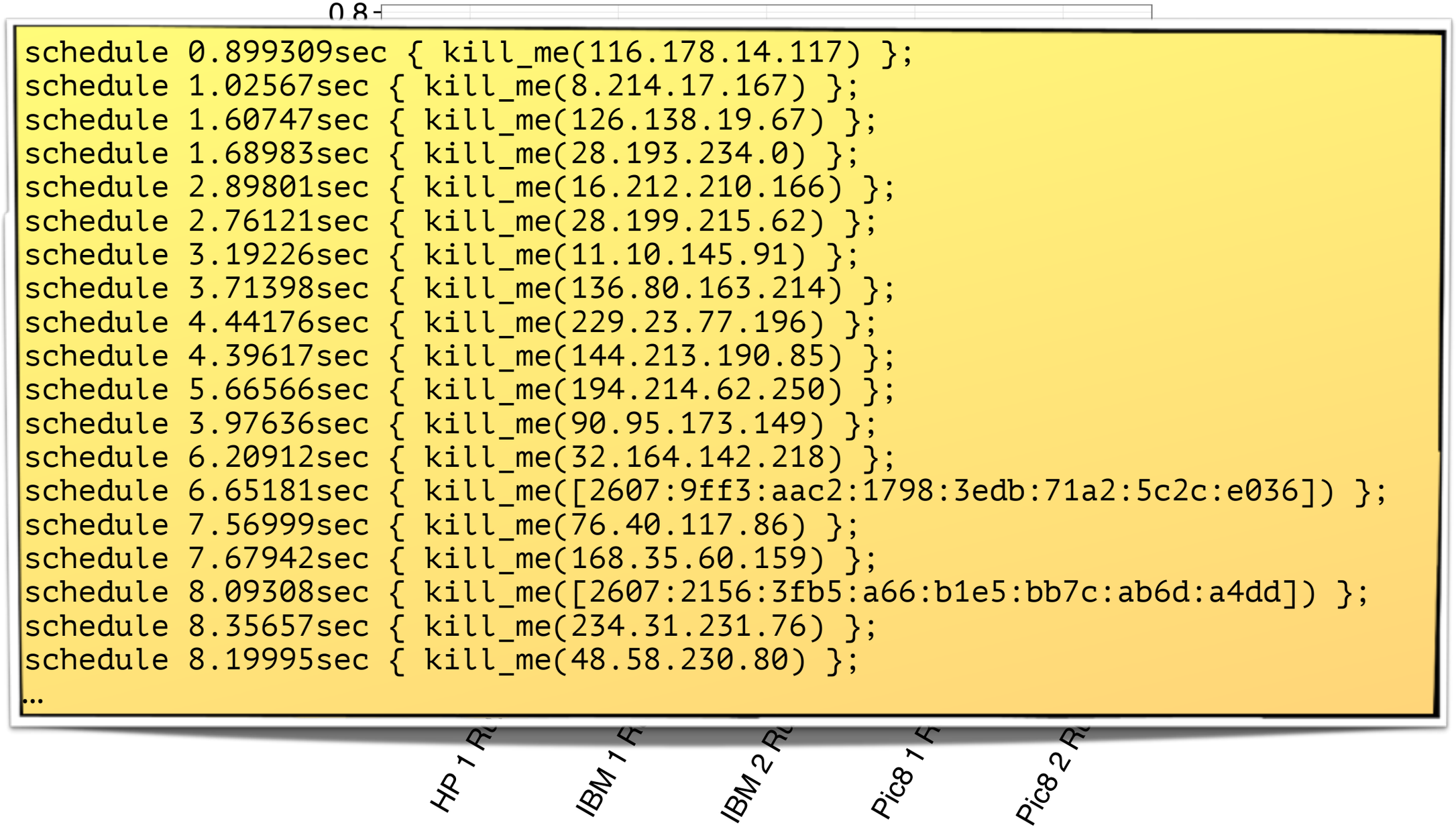
Demonstration

Rule Insertion Speed



Rule Insertion Speed

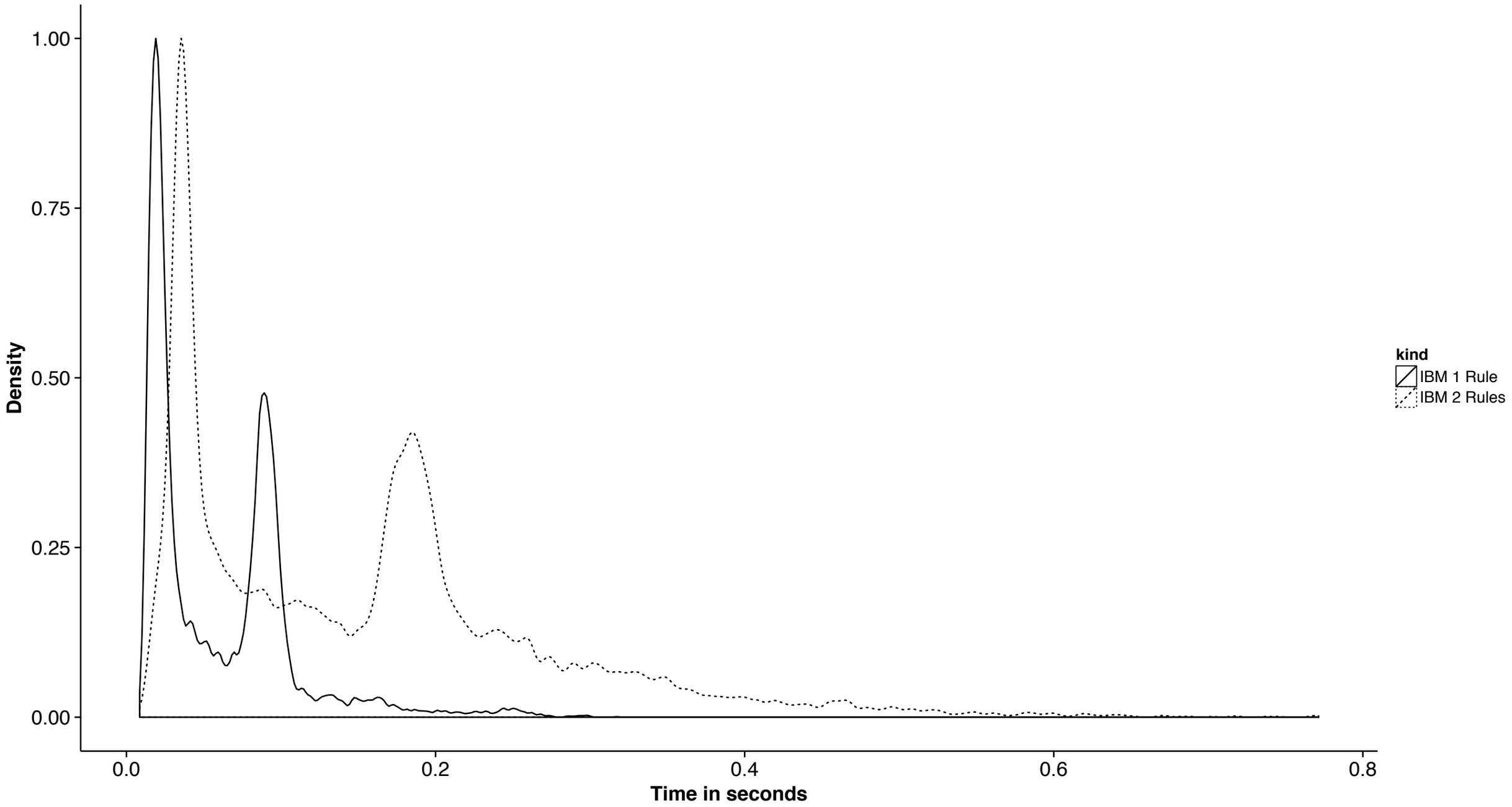
```
schedule 0.899309sec { kill_me(116.178.14.117) };
schedule 1.02567sec { kill_me(8.214.17.167) };
schedule 1.60747sec { kill_me(126.138.19.67) };
schedule 1.68983sec { kill_me(28.193.234.0) };
schedule 2.89801sec { kill_me(16.212.210.166) };
schedule 2.76121sec { kill_me(28.199.215.62) };
schedule 3.19226sec { kill_me(11.10.145.91) };
schedule 3.71398sec { kill_me(136.80.163.214) };
schedule 4.44176sec { kill_me(229.23.77.196) };
schedule 4.39617sec { kill_me(144.213.190.85) };
schedule 5.66566sec { kill_me(194.214.62.250) };
schedule 3.97636sec { kill_me(90.95.173.149) };
schedule 6.20912sec { kill_me(32.164.142.218) };
schedule 6.65181sec { kill_me([2607:9ff3:aac2:1798:3edb:71a2:5c2c:e036]) };
schedule 7.56999sec { kill_me(76.40.117.86) };
schedule 7.67942sec { kill_me(168.35.60.159) };
schedule 8.09308sec { kill_me([2607:2156:3fb5:a66:b1e5:bb7c:ab6d:a4dd]) };
schedule 8.35657sec { kill_me(234.31.231.76) };
schedule 8.19995sec { kill_me(48.58.230.80) };
…
```



Blocked Connections

Switch	Block time	Not blocked	Transferred Bytes		
			Med.	Mean	Max
Pica8 (Median)	8.5ms	4,229 (2.7%)	0	1.6k	68k
Pica8 (75 Percentile)	11ms	8,273 (5.1%)	12	2.3k	101k
IBM (Median)	41ms	27,848 (17.4%)	194	9.5k	1.1MB
IBM (75 Percentile)	89ms	41,965 (26.3%)	526	27k	4.0MB
HP (Median)	82ms	38,381 (24%)	454	23k	4.5MB
HP (75 Percentile)	93ms	43,128 (27%)	537	28k	5.0MB

IBM G8052


1.00
0.75
0.50
0.25
0.00
Density
0.0
0.2
0.4
0.6
0.8
Time in seconds
kind
IBM 1 Rule
IBM 2 Rules

NetControl Summary

Control switches and other hardware

Easy syntax and rules

Extensible (API & Backends)

Fast

Get NetControl

github.com/bro/bro-netcontrol