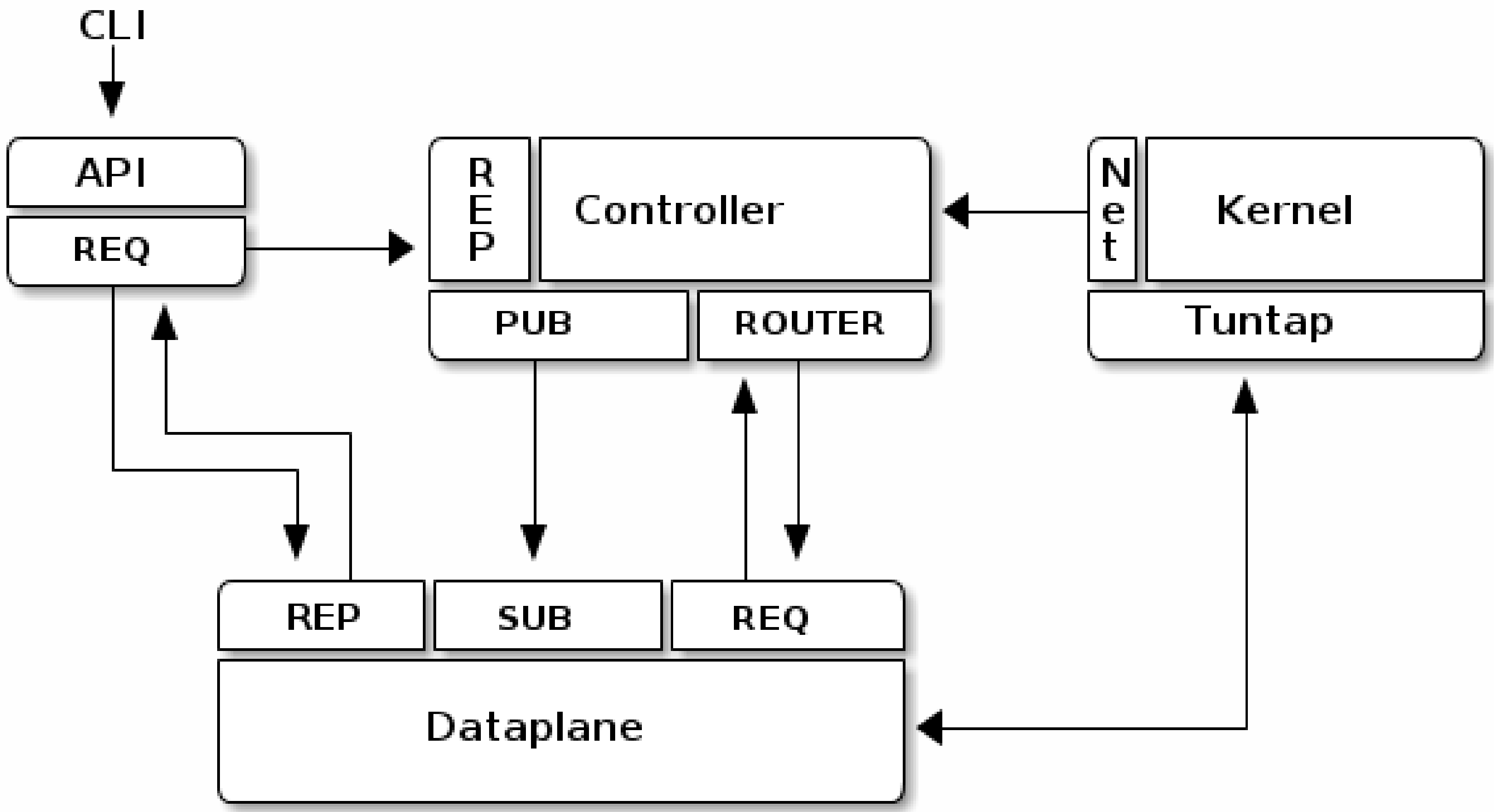
BROCADE

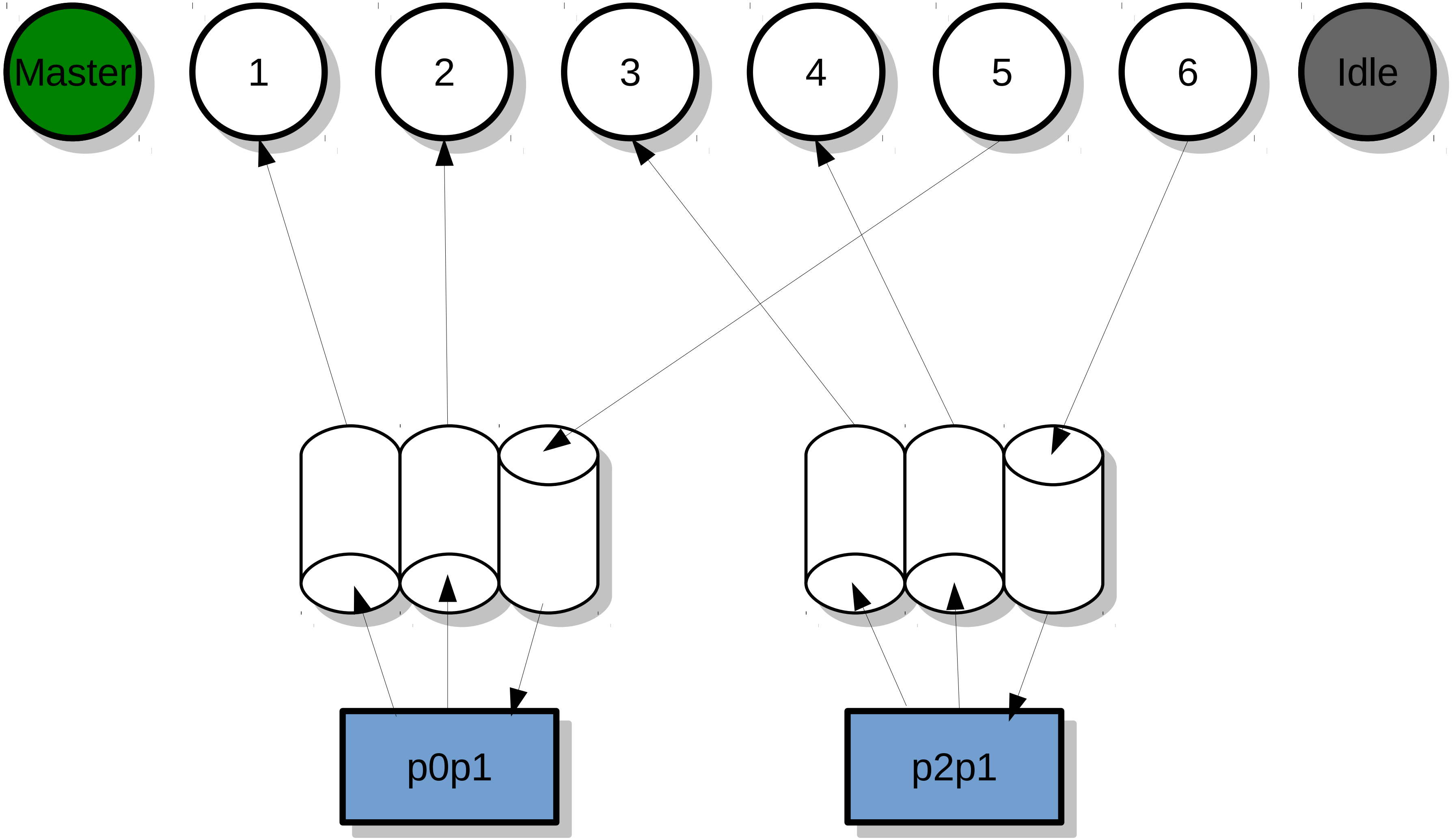
DPDK performance
Lessons learned in vRouter

Stephen Hemminger
stephen@networkplumber.org
@networkplumber

Architecture



Lcore Assignment



Internal Instrumentation

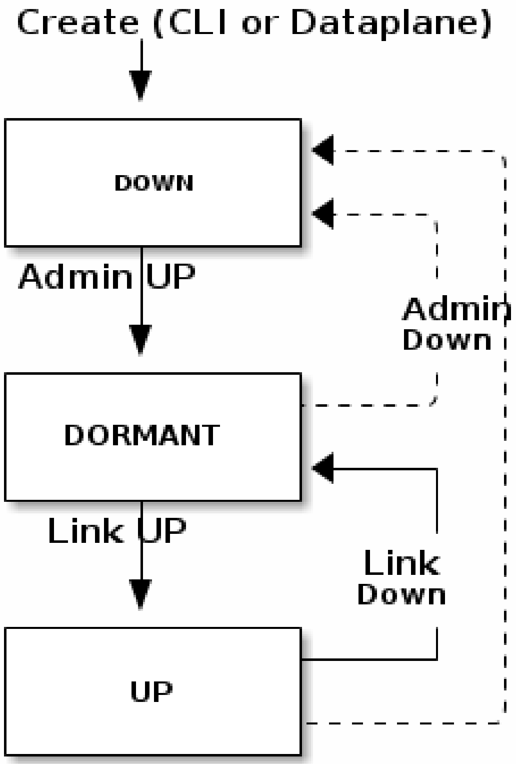
```
Dataplane CPU activity

Core Interface     RX Rate    TX Rate       Idle
------------------------------------------------
1    p1p1            14.9M                     0
2    p1p1                0                   250
3    p33p1               0                   250
4    p33p1               1                   250
5    p1p1                           0        250
6    p33p1                      11.9M          1
```

Idle sleep

- 100% Poll → 100% CPU
  - CPU power limits
  - No Turbo boost
  - PCI bus overhead
- Small sleep's
  - 0 - 250us
  - Based on activity

Link state



- TAP device created by dataplane
- LINK UP/DOWN
  - When change
  - 5sec interval
  - Updates statistics
  - Acts as keepalive

Slowpath

- Packets placed in DPDK rte_ring
  - Wakeup via eventfd
- Shadow thread
  - Poll's for event or kernel packets
- Packet's received
  - Sent to kernel via TAP device
- Local packets
  - injected into Tx Thread

Perf – active thread

```
Samples: 16K of event 'cycles', Event count (approx.): 11763536471
 14.93%  dataplane  [.] ip_input
 10.04%  dataplane  [.] ixgbe_xmit_pkts
  7.69%  dataplane  [.] ixgbe_recv_pkts
  7.05%  dataplane  [.] T.240
  6.82%  dataplane  [.] fw_action_in
  6.61%  dataplane  [.] fifo_enqueue
  6.44%  dataplane  [.] flow_action_fw
  6.35%  dataplane  [.] fw_action_out
  3.92%  dataplane  [.] ip_hash
  3.69%  dataplane  [.] cds_lfht_lookup
  2.45%  dataplane  [.] send_packet
  2.45%  dataplane  [.] bit_reverse_ulong
```

Performance rules

- No syscall’s
- No mutex’s
- Avoid using spinlock
- Real-time SCH_FIFO

TSC counter

CPU stall

```
while(1)
  cur_tsc = rte_rdtsc();
  diff_tsc = cur_tsc – prev_tsc;

  if (unlikely(diff_tsc > drain_tsc)) {
     for (portid = 0; portid < RTE_MAX_ETHPORTS;
portid++) {

       send_burst(qconf,
            qconf->tx_mbufs[portid].len,
            portid);
```

Heisenburg: observing performance slows it down fw_action_in

```
       │   struct ip_fw_args fw_args = {
                         .m = m,
       │                 .client = client,
       │                 .oif = NULL };
  1.54 │1d:    movzbl %sil,%esi
  0.34 │       mov    %rsp,%rdi
  0.04 │       mov    $0x13,%ecx
  0.16 │       xor    %eax,%eax
 57.66 │       rep    stos %rax,%es:(%rdi)
  4.68 │       mov    %esi,0x90(%rsp)
 20.45 │       mov    %r9,(%rsp)
```

Why is QoS slow?

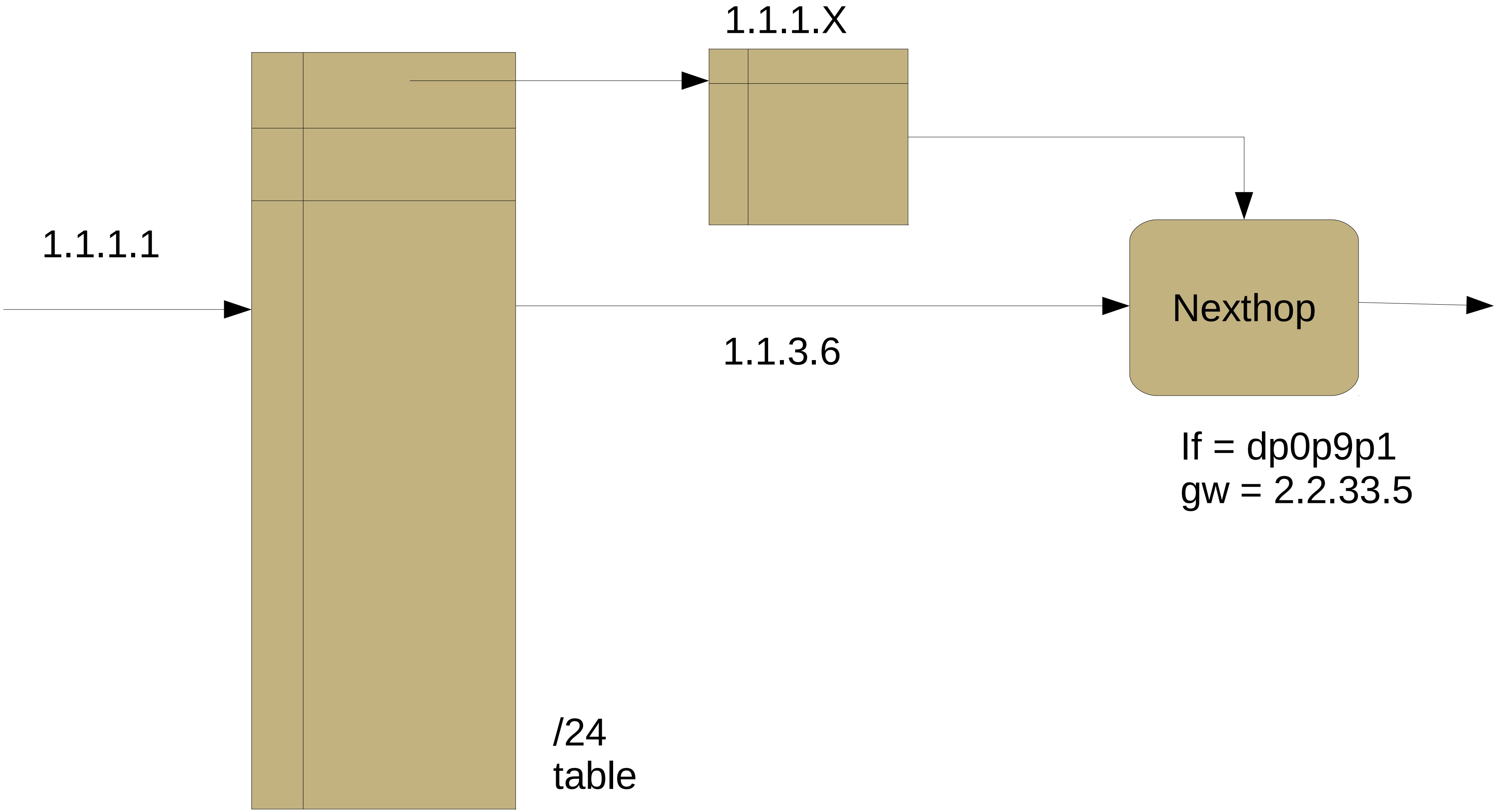
```
static inline void
rte_sched_port_time_resync(struct rte_sched_port *port)
{
     uint64_t cycles = rte_get_tsc_cycles();
     uint64_t cycles_diff = cycles - port->time_cpu_cycles;
     double bytes_diff = ((double) cycles_diff) /
                         port->cycles_per_byte;

     /* Advance port time */
     port->time_cpu_cycles = cycles;
     port->time_cpu_bytes += (uint64_t) bytes_diff;
```

Mutual Exclusion

- Locking
  - Reader/Writer lock is expensive
  - Read lock has more overhead than spin lock
  - Posix locks even more expensive
- Userspace RCU
  - Don't modify, create and destroy
  - Impacts thread model

Longest Prefix Match



LPM issues

- Prefix → 8 bit next hop
- Missing barriers
- Rule update
- Fixed size /8 table

Q & A

Thank you

Stephen Hemminger
stephen@networkplumber.org
@networkplumber