EXT PORT = 2if received_on_port != EXT PORT: # Heartbeat

return ([],[])

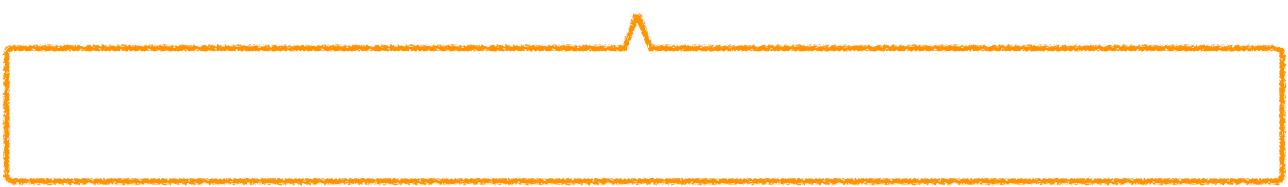
```
34
                                                                                                                                else:
    EXP_TIME = 10 * 1000
                                                                                                                       35
                                                                                                                                     alloc flow and process packet = True
    BACKEND EXP TIME = 3600000000 * 1000
                                                                                                                       36
                                                                                                                                if alloc_flow_and_process_packet:
    EXT PORT = 2
                                                                                                                       37
                                                                                                                                     if backend_ip_emap.exists_with_cht(cht, _LoadBalancedFlow_hash(packet_flow)):
       a_packet_received:
                                                                                                                       38
                                                                                                                                         bknd = backend_ip_emap.choose_with_cht(cht, _LoadBalancedFlow_hash(packet_flow))
        flow emap.expire all(now - EXP TIME)
                                                                                                                                         if not flow emap.full():
                                                                                                                       39
        backend_ip_emap.expire_all(now - BACKEND_EXP_TIME)
                                                                                                                       40
                                                                                                                                              idx = the_index_allocated
                                                                                                                       41
                                                                                                                                              flow_emap.add(packet_flow, idx, now)
        pop header(tcpudp, on mismatch=([],[]))
                                                                                                                       42
                                                                                                                                              flow_id_to_backend_id.set(idx, bknd)
        pop_header(ipv4, on_mismatch=([],[]))
                                                                                                                       43
                                                                                                                                         backend = backends.get(bknd)
       = pop_header(ether, on_mismatch=([],[]))
13
                                                                                                                                         return ([backend.nic],
                                                                                                                       44
14 assert a_packet_received
                                                                                                                       45
                                                                                                                                                  [ether(h1, saddr=..., daddr=backend.mac),
15 assert h1.type == 8 # 0x0800 == IPv4 in big endian
                                                                                                                                                   ipv4(h2, cksum=..., daddr=backend.ip),
                                                                                                                       46
16 assert h2.npid == 6 or h2.npid == 17 # 6/17 -> TCP/UDP
                                                                                                                       47
                                                                                                                                                   tcpudp(h3)])
17
                                                                                                                       48
                                                                                                                                     else:
       received on port == EXT PORT: # Packet from the external network - client
18
                                                                                                                                         return ([],[])
                                                                                                                       49
        packet_flow = LoadBalancedFlowc(h2.saddr, h2.daddr, h3.src_port, h3.dst_port, h2.npid)
19
        alloc_flow_and_process_packet = False;
                                                                                                                            else: # A heartbeat from a backend
20
        if flow_emap.has(packet_flow):
21
                                                                                                                       51
                                                                                                                                 bknd_addr = ip_addrc(h2.saddr)
22
            flow_id = flow_emap.get(packet_flow)
                                                                                                                                 if backend_ip_emap.has(bknd_addr):
                                                                                                                       52
            backend_id = flow_id_to_backend_id.get(flow_id)
23
                                                                                                                       53
                                                                                                                                     backend ip emap.refresh idx(backend ip emap.get(bknd addr), now)
            if backend_ip_emap.has_idx(backend_id):
24
                                                                                                                       54
                                                                                                                                else:
               flow_emap.refresh_idx(flow_emap.get(packet_flow), now)
25
                                                                                                                       55
                                                                                                                                     if not backend_ip_emap.full():
               backend = backends.get(backend_id)
26
                                                                                                                       56
                                                                                                                                         idx = the_index_allocated
27
                return ([backend.nic],
28
                       [ether(h1, saddr=..., daddr=backend.mac),
                                                                                                                                         backend_ip_emap.add(bknd_addr, idx, now)
                                                                                                                       57
                        ipv4(h2, cksum=..., daddr=backend.ip),
29
                                                                                                                       58
                                                                                                                                         backends.set(idx, LoadBalancedBackendc(received_on_port, h1.saddr, h2.saddr))
30
                        tcpudp(h3)])
                                                                                                                       59
                                                                                                                                return ([],[])
31
            else:
               flow_emap.erase(packet_flow)
                alloc_flow_and_process_packet = True
33
```

from state import flow emap, flow id to backend id, backends, backend ip emap, cht

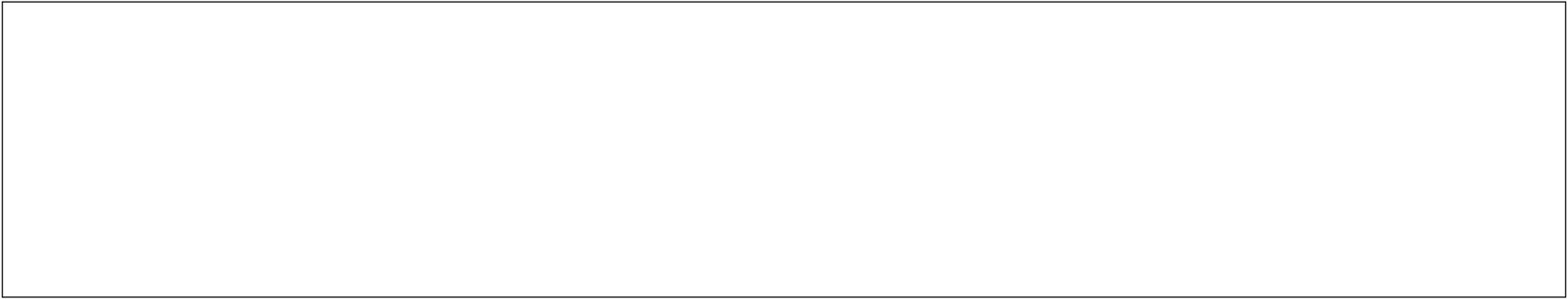
Full-stack



Push-button



Support for partial specifications





Support for partial specifications

$EXT_PORT = 2$

if received_on_port != EXT_PORT: # Heartbeat return ([],[])

```
from state import flow_emap, flow_id_to_backend_id, backends, backend_ip_emap, cht
   EXP\_TIME = 10 * 1000
    BACKEND_EXP_TIME = 3600000000 * 1000
    EXT_PORT = 2
 6 if a_packet_received:
        flow_emap.expire_all(now - EXP_TIME)
        backend_ip_emap.expire_all(now - BACKEND_EXP_TIME)
    h3 = pop_header(tcpudp, on_mismatch=([],[]))
    h2 = pop_header(ipv4, on_mismatch=([],[]))
    h1 = pop_header(ether, on_mismatch=([],[]))
14 assert a_packet_received
15 assert h1.type == 8 # 0x0800 == IPv4 in big endian
    assert h2.npid == 6 or h2.npid == 17 # 6/17 -> TCP/UDP
18 if received_on_port == EXT_PORT: # Packet from the external network - client
        packet_flow = LoadBalancedFlowc(h2.saddr, h2.daddr, h3.src_port, h3.dst_port, h2.npid)
        alloc_flow_and_process_packet = False;
        if flow_emap.has(packet_flow):
            flow_id = flow_emap.get(packet_flow)
23
            backend_id = flow_id_to_backend_id.get(flow_id)
24
            if backend_ip_emap.has_idx(backend_id):
                flow_emap.refresh_idx(flow_emap.get(packet_flow), now)
                backend = backends.get(backend_id)
                return ([backend.nic],
                         [ether(h1, saddr=..., daddr=backend.mac),
                         ipv4(h2, cksum=..., daddr=backend.ip),
30
                         tcpudp(h3)])
31
            else:
32
                flow_emap.erase(packet_flow)
33
                alloc_flow_and_process_packet = True
```

```
else:
            alloc_flow_and_process_packet = True
        if alloc_flow_and_process_packet:
            if backend_ip_emap.exists_with_cht(cht, _LoadBalancedFlow_hash(packet_flow));
                bknd = backend_ip_emap.choose_with_cht(cht, _LoadBalancedFlow_hash(packet_flow))
                if not flow_emap.full():
                    idx = the_index_allocated
                    flow_emap.add(packet_flow, idx, now)
                    flow_id_to_backend_id.set(idx, bknd)
                backend = backends.get(bknd)
                return ([backend.nic],
                         [ether(h1, saddr=..., daddr=backend.mac),
                         ipv4(h2, cksum=..., daddr=backend.ip),
                         tcpudp(h3)])
            else:
                return ([],[])
    else: # A heartbeat from a backend
        bknd_addr = ip_addrc(h2.saddr)
         if backend_ip_emap.has(bknd_addr):
            backend_ip_emap.refresh_idx(backend_ip_emap.get(bknd_addr), now)
53
54
55
            if not backend_ip_emap.full():
56
                 idx = the_index_allocated
57
                backend_ip_emap.add(bknd_addr, idx, now)
58
                backends.set(idx, LoadBalancedBackendc(received_on_port, h1.saddr, h2.saddr))
        return ([],[])
59
```

Whole SW stack: framework, OS, drivers



NF: 3 KLOC

Framework, driver: 85 KLOC