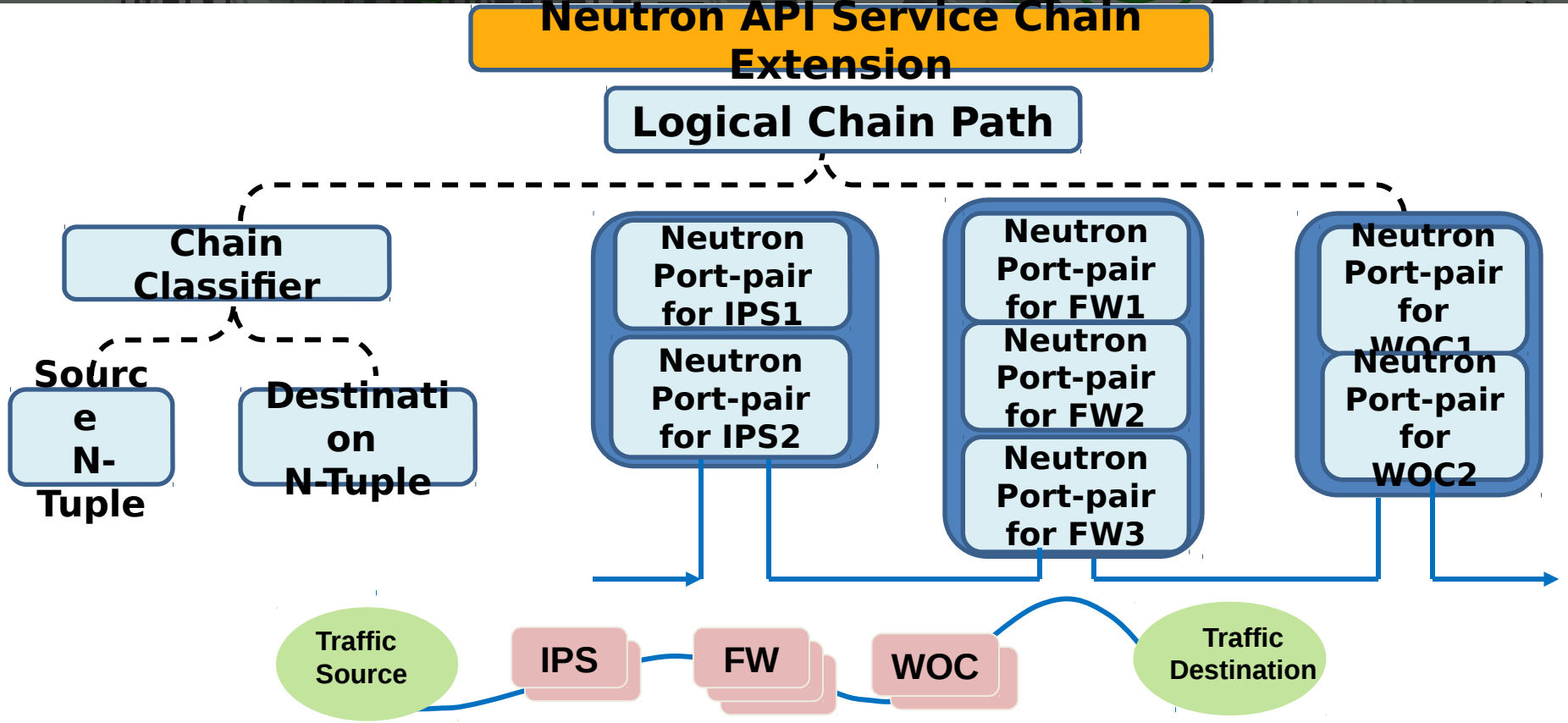
Flavio Fernandes
Louis Fourie
John McDowall
Farhad Sunavala

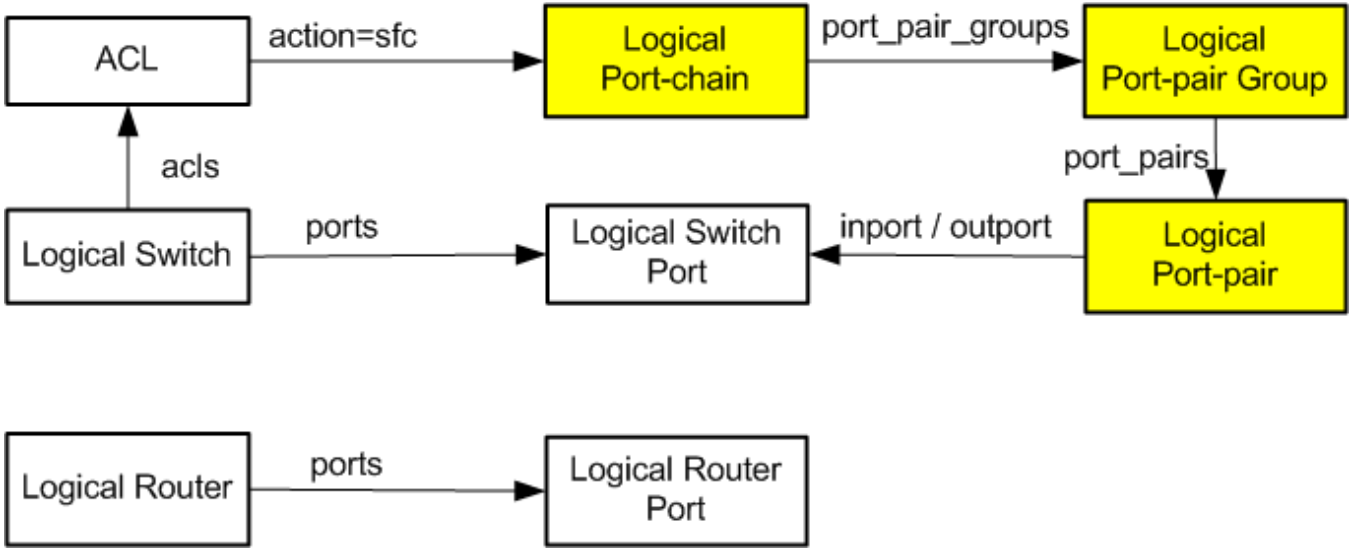
Service Function Chaining for OVN

OpenStack Neutron SFC Model



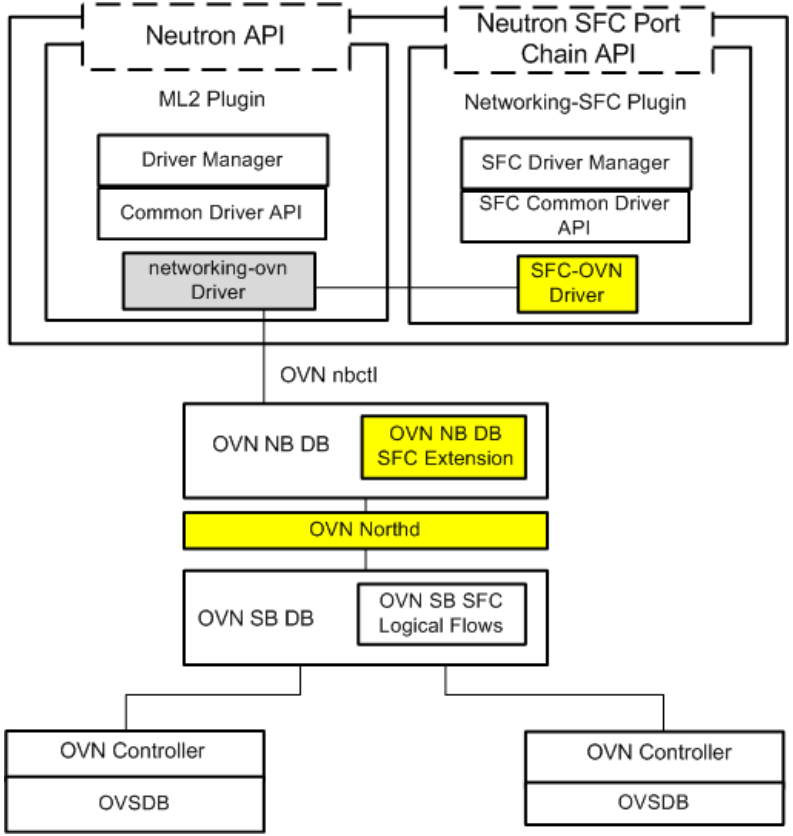
Networking-sfc / OVN Integration

- **Extend ovn-nb schema to add Port Chain, Port-Pair-Group, Port-Pair Tables**
- **Use Logical Port-pair (pairs of Logical Switch Ports) to define each SF**
- **Use ovn-nb ACLs as flow-classifiers**



Neutron – OVN Architecture

- Neutron drivers invoke ovn-nbctl
- SFC Extensions to OVN NB DB
- Translation to OVN SB SFC Logical Flows
- No changes to OVS



OVN-NB Schema Extension (ovn-nb.ovsschema)

```
"Logical_Port_Chain": {
        "columns": {
            "port_pair_groups": {"type": {"key": {"type": "uuid",
                                  "refTable": "Logical_Port_Pair_Group",
                                  "refType": "strong"},
                                  "min": 0, "max": "unlimited"}} },
        "isRoot": true},

"Logical_Port_Pair_Group": {
        "columns": {
            "port_pairs": {"type": {"key": {"type": "uuid",
                              "refTable": "Logical_Port_Pair", "refType": "strong"},
                              "min": 0, "max": "unlimited"}} },
        "isRoot": false},

"Logical_Port_Pair": {
        "columns": {
            "outport": {"type": {"key": {"type": "uuid",
                                 "refTable": "Logical_Switch_Port", "refType": "strong"},
                                 "min": 0, "max": 1}},
            "inport": {"type": {"key": {"type": "uuid",
                                "refTable": "Logical_Switch_Port", "refType": "strong"},
                                "min": 0, "max": 1}},
        "isRoot": false},
```

ACL Table Extension for SFC

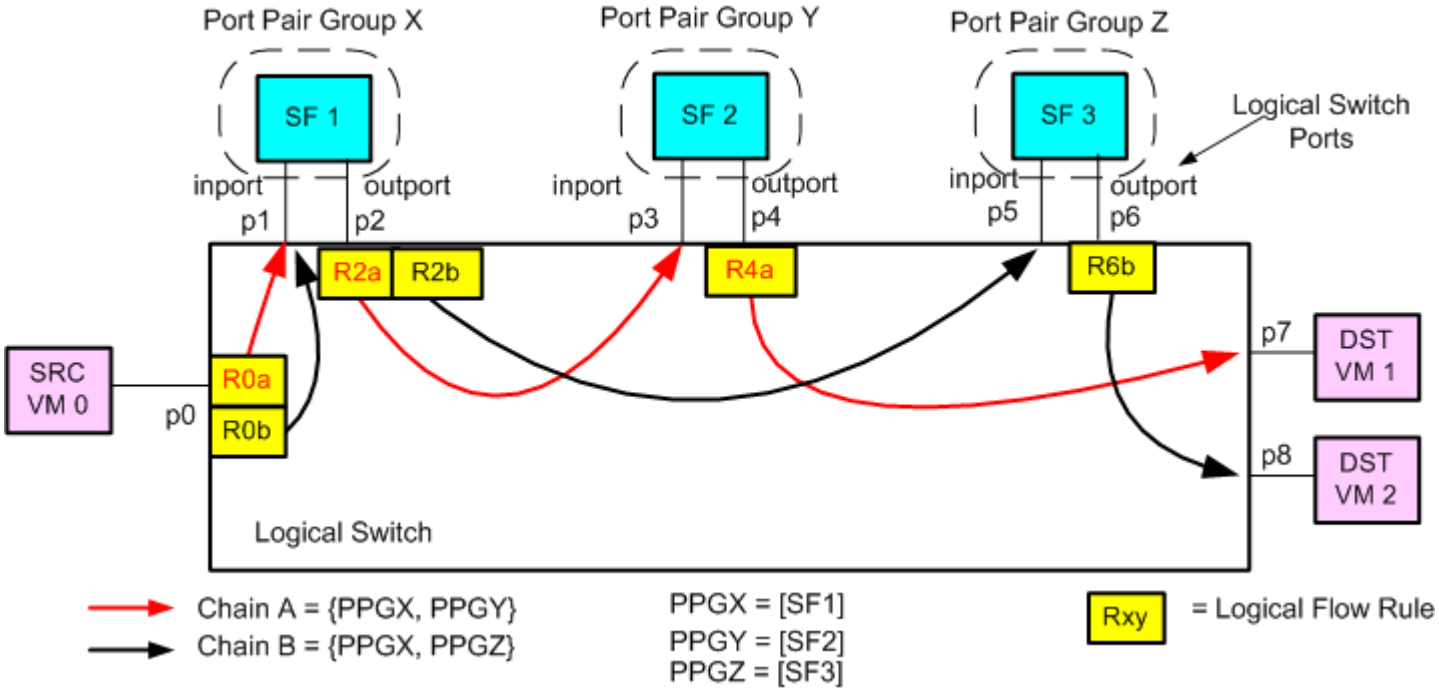
```
"ACL": {
       "columns": {
         "priority": {"type": {"key": {"type": "integer",
                                       "minInteger": 0,
                                       "maxInteger": 32767}}},
         "direction": {"type": {"key": {"type": "string",
                                        "enum": ["set", ["from-lport", "to-port"]]}}},
         "match": {"type": "string"},
         "action": {"type": {"key": {"type": "string",
                                     "enum": ["set", ["allow", "allow-related",
                                                      "drop", "reject", "sfc"]]}}},
         "log": {"type": "boolean"},
         "options": {"type": {"key": "string", "value": "string",
                     "min": 0, "max": "unlimited"}},
         "external_ids": {"type": {"key": "string", "value": "string",
                          "min": 0, "max": "unlimited"}}},
```

ovn-nbctl SFC Commands

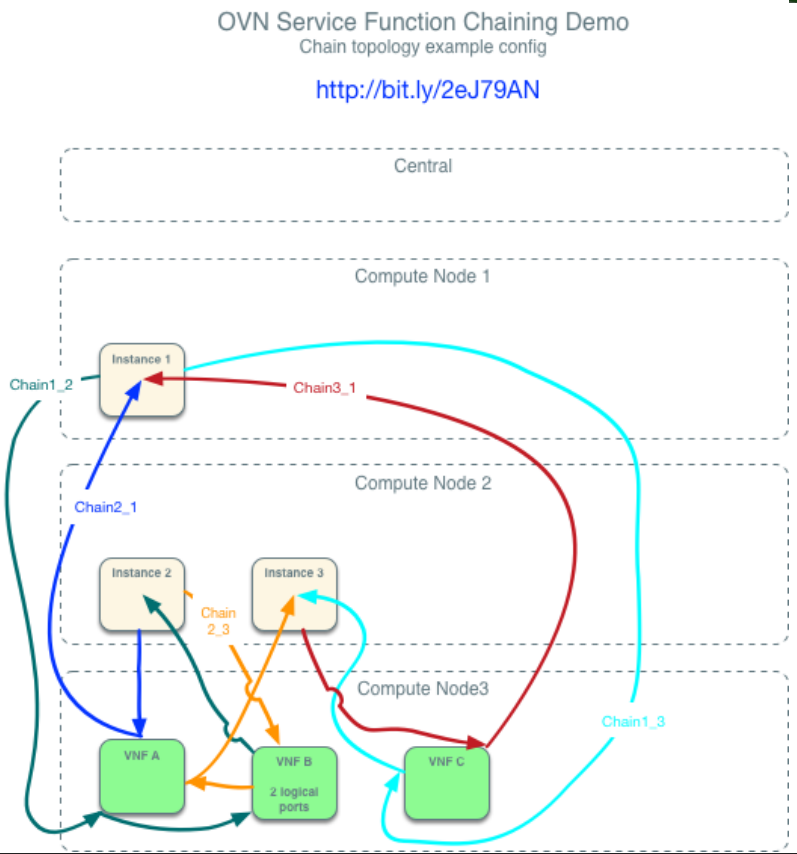
```
lchain-add lchain [lsp-pair-group] ...
lchain-del lchain
lchain-set-port-pair-group lchain [lsp-pair-group] ...
lchain-list lchain
lchain-set-options lchain key=value [key=value]...

lsp-pair-group-add port-pair-group [lsp-pair]...
lsp-pair-group-del lsp-pair-group
lsp-pair-group-set-port-pair lsp-pair-group [lsp-pair]
lsp-pair-group-list
lsp-pair-group-set-options lsp-pair-group key=value [key=value]...

lsp-pair-add lsp-pair [inport, outport]
lsp-pair-del lsp-pair
lsp-pair-list
lsp-pair-set-options lsp-pair key-value [key=value] ...

acl-add ls direction priority match sfc [sfc-port-chain=<lchain>]
```

OVN SB Rules for SFCs on one Logical Switch

- **Service Functions shared by multiple SFCs**
- **Logical rules at each SF outport**

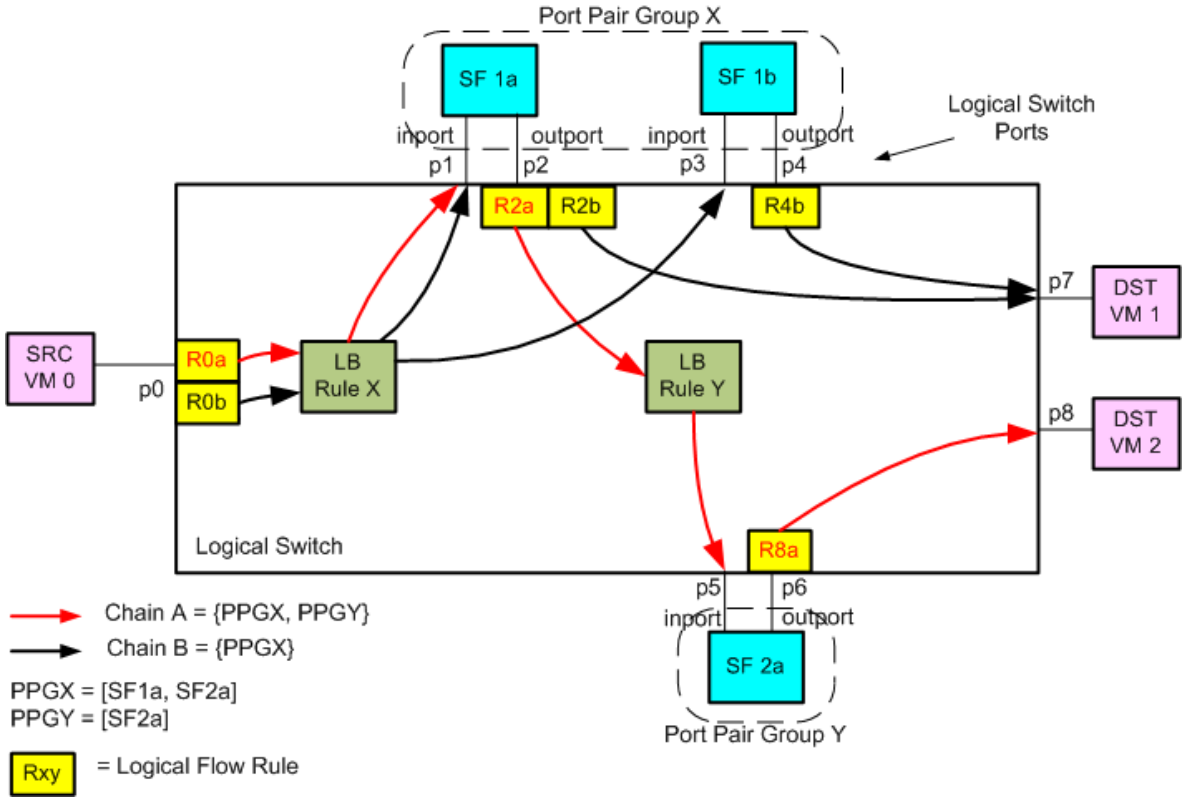


Demo


OVN Service Function Chaining Demo
Chain topology example config
http://bit.ly/2eJ79AN
Central
Compute Node 1
Instance 1
Chain1_2
Chain3_1
Compute Node 2
Chain2_1
Instance 2
Instance 3
Chain 2_3
Compute Node3
Chain1_3
VNF A
VNF B
2 logical ports
VNF C

SFC/OVN Demo

Load-balance to multiple SFs in a Port-pair Group



OVN Ingress Pipeline

L2 Port Security	Security of ports
IP Port Security	IP Port Security
ND Port Security	Neighbor discover port security constraints on ARP and IPv6
Pre-ACL	Handle connection tracking ACL Packets
Pre-LB	Deal with potentially fragmented packets for LB
Pre-Stateful ACL	Defragment connection tracking packets
	Apply ACL Rules
Service Chain	Service chain steering
LB	Apply LB rules (extension for LSP balancing)
Stateful	Continue stateful packets
ARP Response	Send ARP Response
DHCP Options	Set DHCP Options on packet
DHCP Response	Send DHCP Response

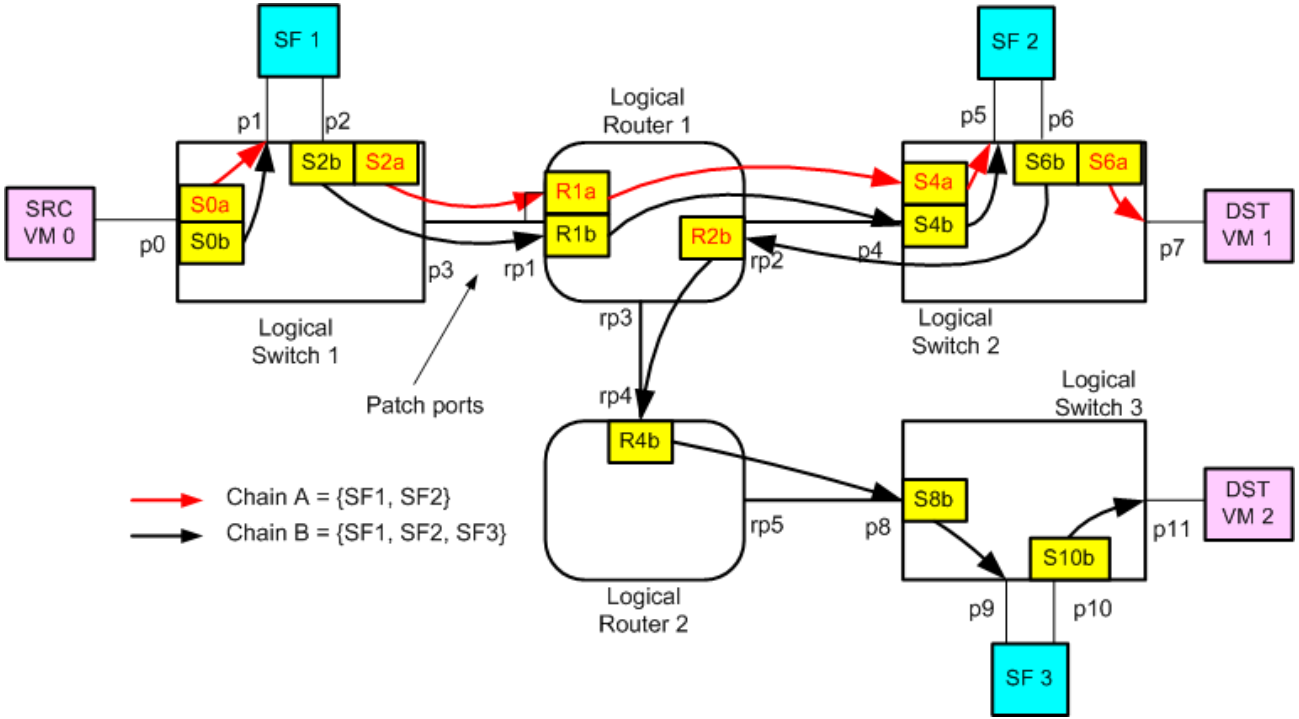
OVN Ingress Pipeline Tables

1. L2 Port Security Table (ls_in_port_sec_l2). Validates that the source MAC of a packet received from a LSP matches the actual MAC for that port. This must be bypassed since the SF egress ports will send packets with many different source MACs
2. Chain Table (ls_in_chain) rules (re-)classify traffic and steer it to a rule in the Port Load Balancer (PLB) Table that represents the next hop Port Pair Group
3. Port Load Balancer (PLB) Table load balances traffic over a set of LSPs in a Logical Switch. Each rule in this Table corresponds to a Port Pair Group and has a Port Load Balance (plb) action which selects an output LSP from a list of output LSPs. Each Port Load Balancer action is mapped to an OVS group where each bucket is a LSP.

```
table=x(ls_in_plb), priority=2002, match=(reg2 == LB_ID),
     action=(plb(weight=x, outportX, weight=y, outportY,
                 weight=z, outportZ, ...); next;)
```

SFCs that span multiple Logical Switches

- Inport and outport for each SF must be on same LS
- All Port-pairs in a Port-Pair group must be on same LS



Issues for discussion with OVN Team

- SFC that span multiple LS - SFC steering rules on Logical Router
- Enhance OVN LB for port-pair-groups
- NSH-aware SFs
- NSH over Geneve
  https://datatracker.ietf.org/doc/draft-fourie-nvo3-nsh-geneve-encap/
- GitHub: https://github.com/doonhammer/ovs/tree/sfc.v2

Backup Material

Chain Table Logical Flow Rules

Source Port Rule classifies traffic from the source LSP and steers it to the first Load Balancer entry for the Logical Chain in the PLB Table. Metadata register reg2 selects LB entry in the PLB Table.

```
table=x(ls_in_chain), priority=2002,
    match=(inport == “ingress-port” && flow-classifier),
    action=(reg2 = LB_ID1; next(ls_in_plb);)
```

SF Re-classifier Rules re-classifies traffic from the SF outport and steers it to the next Load Balancer in the PLB Table. For an NSH-aware SF the flow-classifier matches the NSP/NSI and for NSH-unaware SF the flow-classifier matches the N-tuple.

```
table=x(ls_in_chain), priority=2002,
    match=(inport == “SFn-outport” && flow-classifier),
    action=(reg2 = LB_IDx; next(ls_in_plb);)
```

Final Re-classifier Rule re-classifies traffic from the last SF outport of a Logical Chain and steers it to the destination port.

```
table=x(ls_in_chain), priority=2002,
    match=(inport == “SFfinal-outport” && flow-classifier),
    action=(outport="destport"; output;)
```