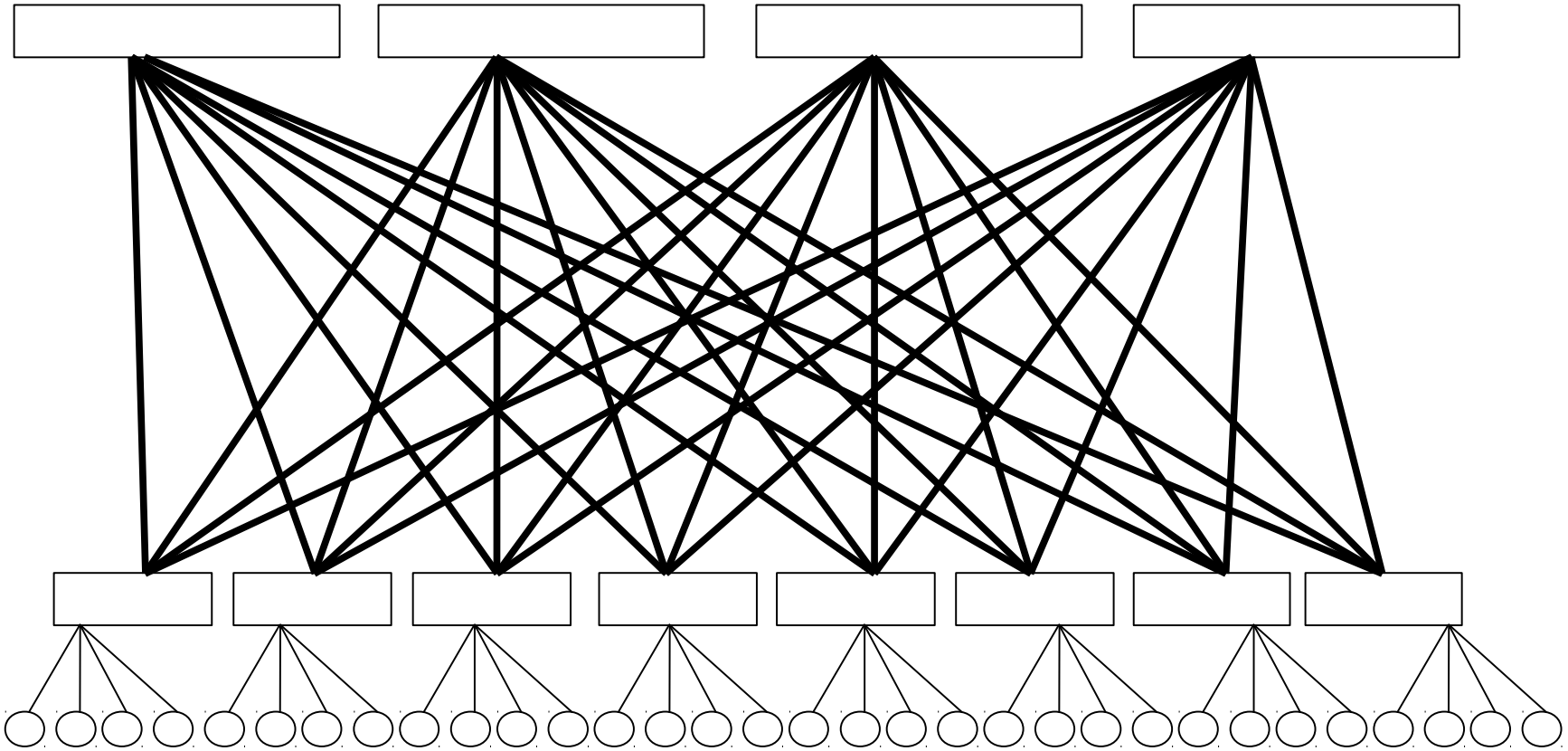
Prescriptive Topology Daemon

Dinesh G Dutt, Pradosh Mohapatra
Cumulus Networks

Data Center Network Design Transition

- Scale-up → Scale-out
- Layer 2 → Layer 3
- Legacy software → Linux ecosystem

Network topologies that cater to the transition



Topology properties

- High cross-sectional bandwidth
- No single point of failure
- Predictable performance

- Recent proposals for irregular topologies, e.g. Jellyfish from UIUC
  - Randomness has good properties

Scale-out and cabling complexity

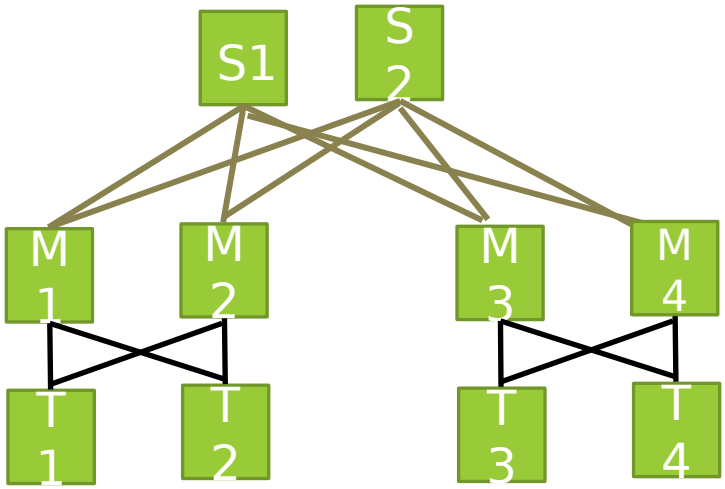
- With network growth, #cables grows rapidly
- An “m x n” 2-level fat tree cluster requires O(mn) cables
  - Goes higher as #levels increase
  - Tens of thousands of cables in a data center
- Topology design → Network blueprint → Cable install
- Steady state → Failures → Recabling
- How do we ensure cabling correctness?

Cabling Errors

- “To err is human” – Alexander Pope
- Issues caused by improper cabling:
  - Reachability issues
  - Unpredictable (and low) performance

Prescriptive Topology Manager

- Verify connectivity is as per operator specified cabling plan
- Take defined actions on topology check dynamically
  - For example, routing adjacency is brought up only if physical connectivity check passes
- Example:
  - `T1, port1 is connected to M1, port1`
  - `T1, port2 is connected to M2, port1`
  - …
  - `M1, port 3 is connected to S1, port1`
  - `M1, port 4 is connected to S2, port1`

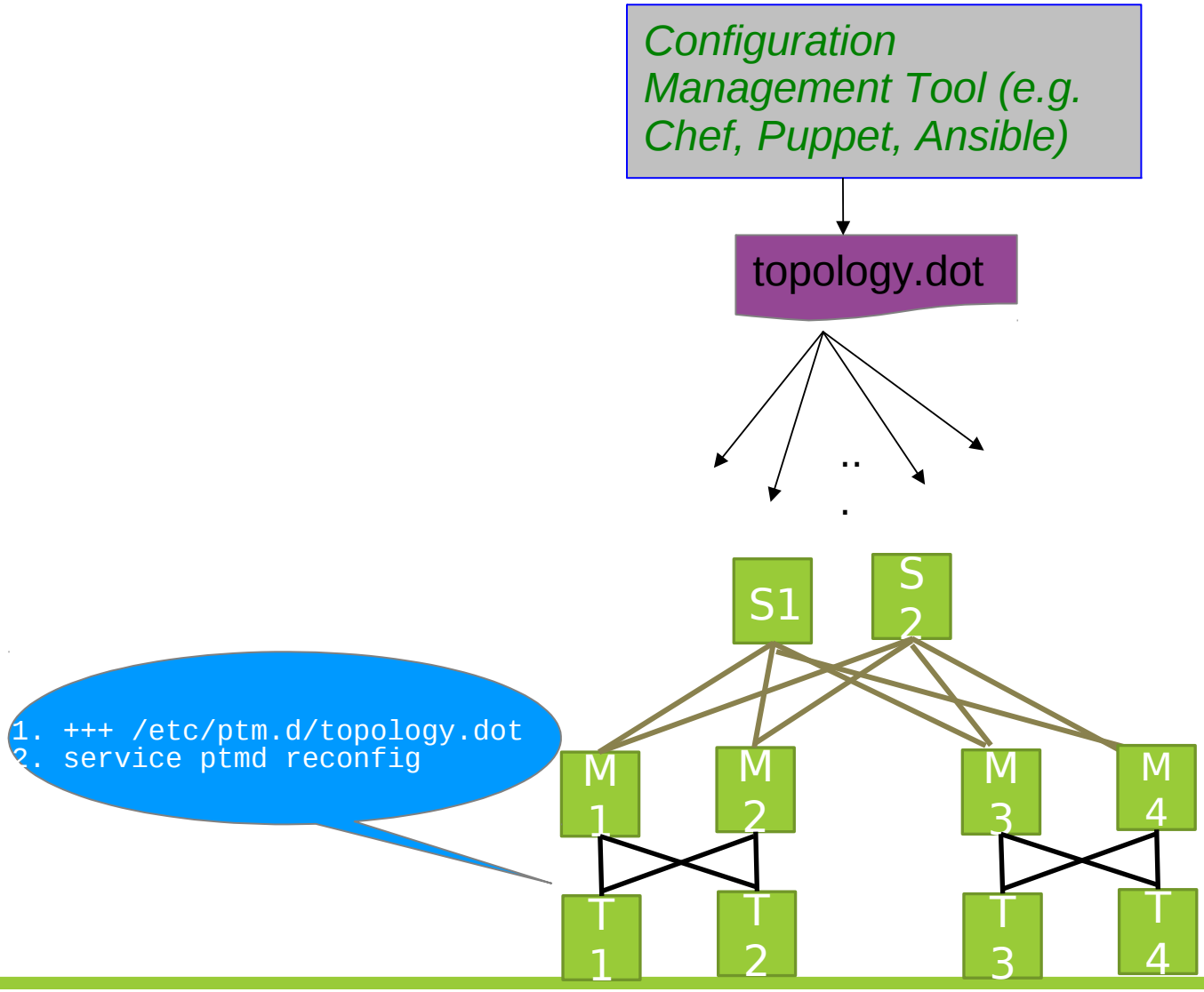


Building Blocks

- Graphviz: Network topology specified via DOT language
  - Well understood graph modeling language
  - Wide range of supported tools
  - Open source
- Central management tool: Network topology is pushed out to all nodes
  Each node determines its relevant information
- LLDP: Use the discovery protocol to verify connectivity

```
digraph G {
        graph [hostidtype="hostname", version="1:0", date="06/26/2013"];
        S1:swp1 -> M1:swp3;
        S1:swp2 -> M2:swp3;
        S1:swp3 -> M3:swp3;
        S1:swp4 -> M4:swp3;
        S2:swp1 -> M1:swp4;
        M1:swp1 -> T1:swp1;
        M1:swp2 -> T2:swp1;
        M2:swp1 -> T1:swp2;
        M2:swp2 -> T2:swp2;
}
```

Picture


Configuration Management Tool (e.g. Chef, Puppet, Ansible)
topology.dot
..
.
S1
S2
1. +++ /etc/ptm.d/topology.dot
2. service ptmd reconfig
M1
M2
M3
M4
T1
T2
T3
T4

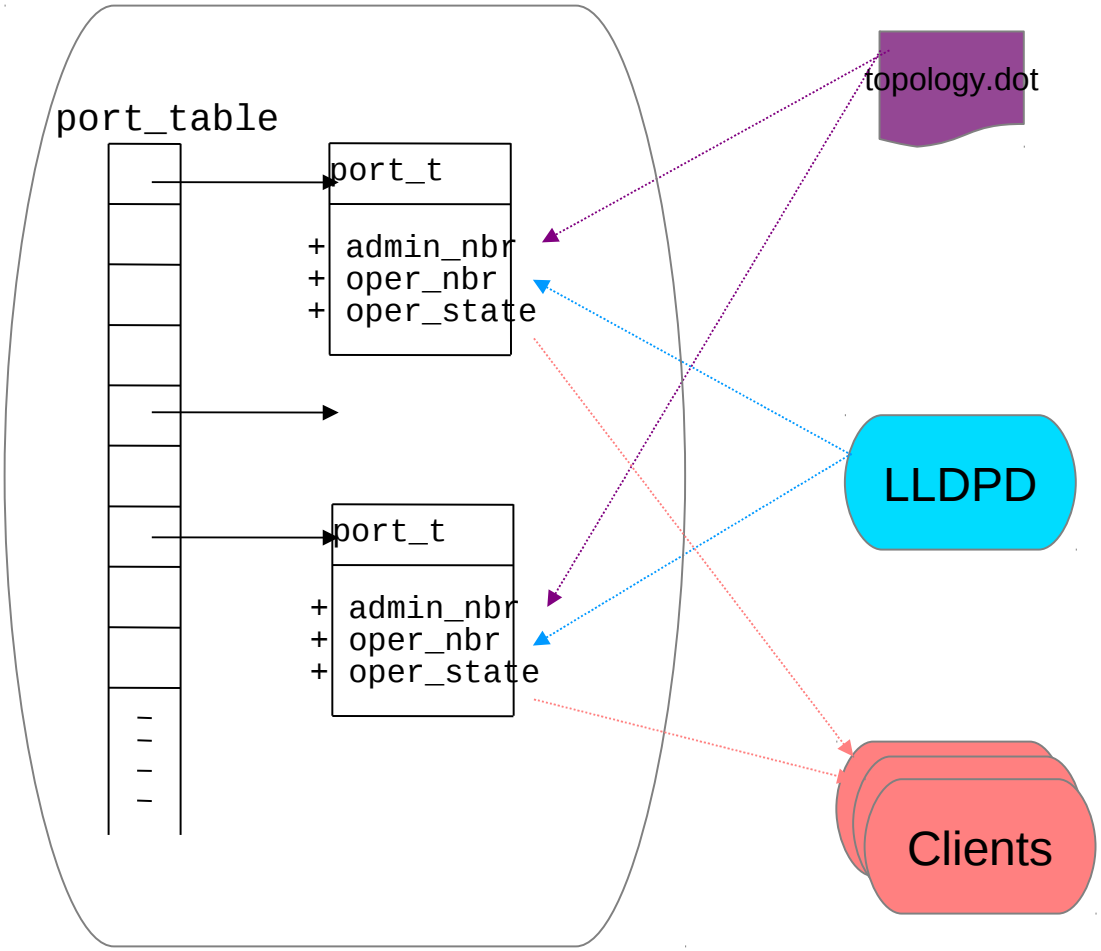
Implementation

- Developed and tested on Linux (wheezy release of Debian)
- Written in C and Python
- Communicates with LLDPD (based on *https://github.com/vincentbernat/lldpd*)
- PTMD executes scripts on topology pass and topology fail

`/etc/ptm.d/if-topo-pass, /etc/ptm.d/if-topo-fail`

`Example: add/del routing protocol interface configuration`

Core implementation details



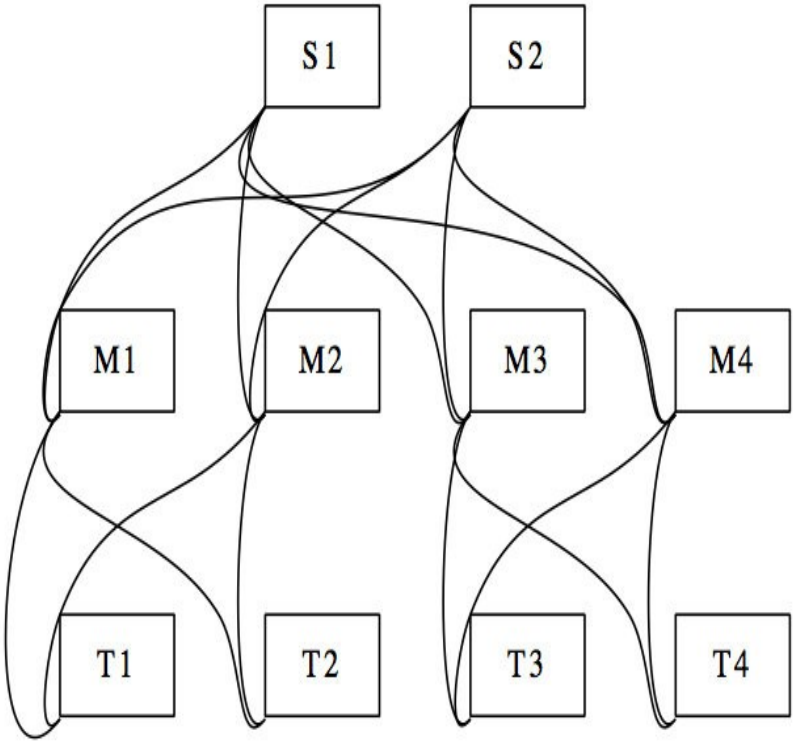
ptmctl

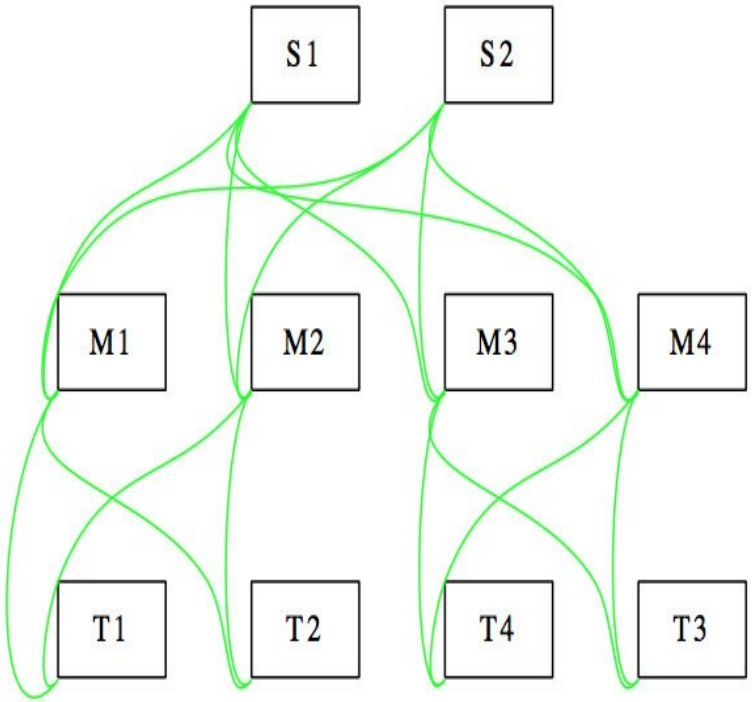
```
cumulus@S1:~# ptmctl
--------------------------------------------------------------------
Port   Status Expected Nbr       Observed Nbr       Last Updated
--------------------------------------------------------------------
swp1   pass   M1:swp3            M1:swp3            17h:39m:21s
swp2   pass   M2:swp3            M2:swp3            17h:39m:21s
swp3   pass   M3:swp3            M3:swp3            17h:39m:21s
swp4   pass   M4:swp3            M4:swp3            17h:39m:21s

cumulus@S1:~#
```

ptmviz – topology analysis

- Generate the DOT file corresponding to the observed physical network topology



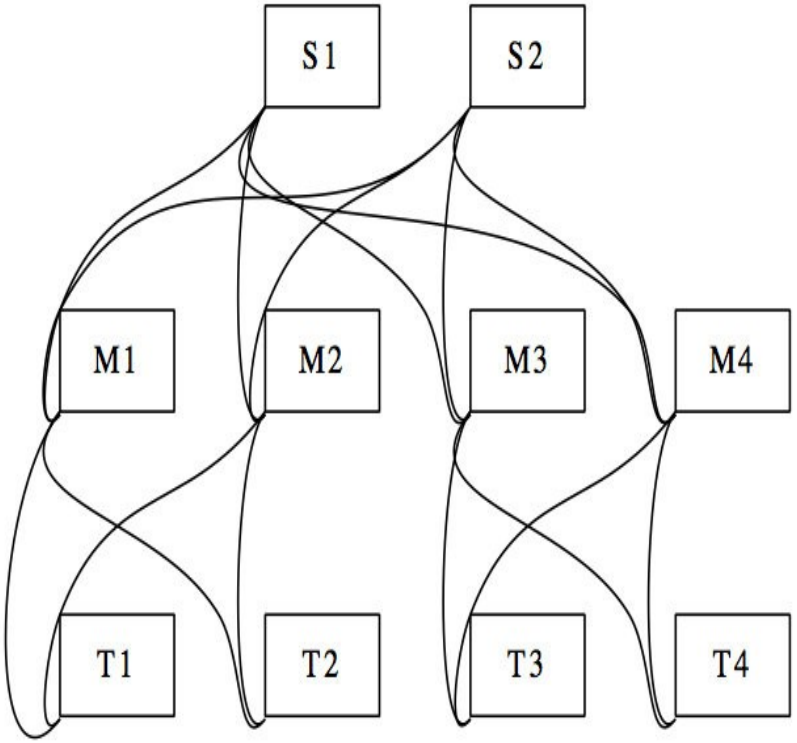


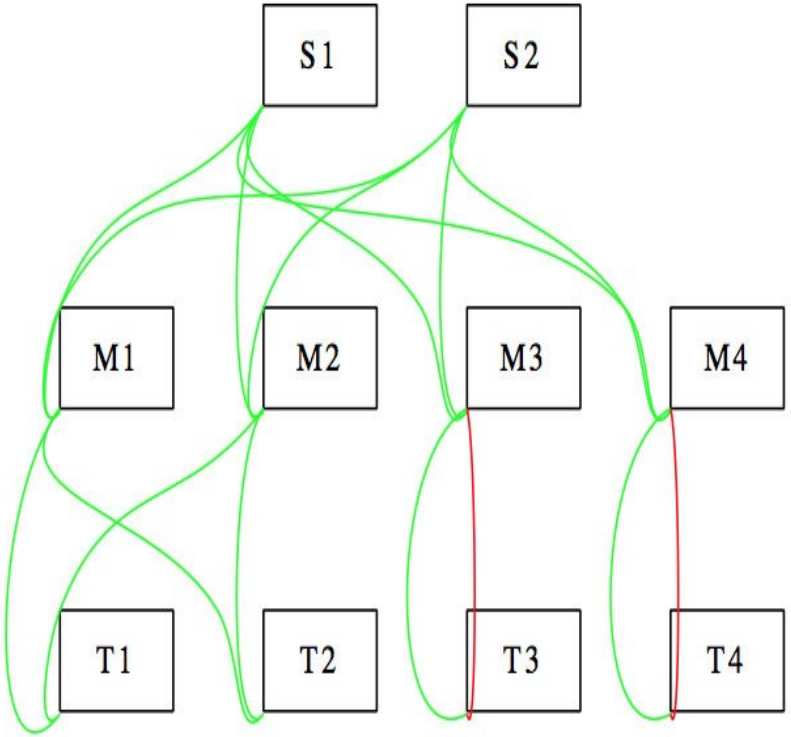

```
dot -Tps prescribed.dot -o prescribed.ps
```

```
dot -Tps physical.dot -o physical.ps
```

ptmviz – topology analysis

- Generate the DOT file corresponding to the observed physical network topology






```
dot -Tps prescribed.dot -o prescribed.ps
```

```
dot -Tps physical.dot -o physical.ps
```

Quagga integration

- New command to enable PTM oper-state based routing protocol bring-up
- Quagga acts as PTM client. Listens to oper-state notifications

```
cumulus@S1:~# sudo vtysh -c 'conf t' -c 'ptm-enable'
cumulus@S1:~# sudo vtysh -c 'show interface swp1'
Interface swp1 is up, line protocol is up
  PTM status: pass
  index 3 metrix 1 mtu 1500
  flags: <UP,BROADCAST,RUNNING,MULTICAST>
  HWaddr: 00:02:00:00:00:11
  inet 21.0.0.2/24 broadcast 21.0.0.255
  inet6 fe80::202:ff:fe00:11/64
cumulus@S1:~#
```

Interoperability

- Any device running an LLDP daemon
- Routing adjacencies can be brought by th device running PTM.

```
digraph G {
        graph [hostidtype="hostname", version="1:0", date="06/
        S1:swp1 -> S2:swp1;
        S1:swp2 -> S2:swp2;
        S1:swp3 -> "procurve1.lab":21;
        S1:swp4 -> "procurve1.lab":22;
        S1:swp5 -> "cisco1.lab":"GigabitEthernet0/1";
        S1:swp5 -> "jmx480":"xe-0/0/0";
        S1:swp7 -> webserver1:eth0;
        S1:swp8 -> webserver1:eth1;
}
```

```
cumulus@S1:~# ptmctl
--------------------------------------------------------------------------
Port   Status Expected Nbr       Observed Nbr       Last Updated
--------------------------------------------------------------------------
swp1   pass   S2:swp1            S2:swp1            17m: 2s
swp2   pass   S2:swp2            S2:swp2            17m: 2s
swp3   pass   procurve1.lab:21   procurve1.lab:21   17m: 10s
swp4   pass   procurve1.lab:22   procurve1.lab:22   17m: 10s
swp5   pass   cisco1.lab:GigabitEthernet0/1 cisco1lab:GigabitEthernet0/1  17m: 8s
swp6   pass   jmx480.lab:xe-0/0/0  jmx480.lab:xe-0/0/0  17m: 1s
swp7   pass   webserver1:eth0    webserver1:eth0    17m: 3s
swp8   pass   webserver1:eth1    webserver1:eth1    17m: 3s
```

Availability

- Open source, published under Eclip
(EPL)
- https://github.com/CumulusNetwork

Roadmap

- Provide abstractions for:
  - routing configuration
  - Network troubleshooting

Thank you

- Questions?

www.cumulusnetworks.com