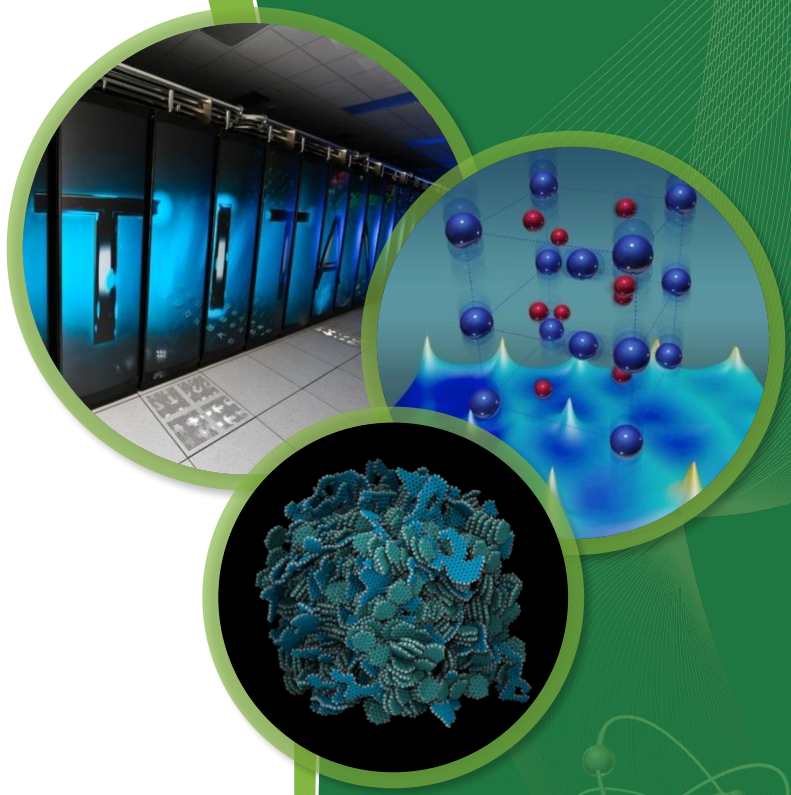
Improved I/O Using Native Spectrum Scale (GPFS) Clients on a Cray XC System



Jesse Hanley, Chris Muzyn, Matt Ezell

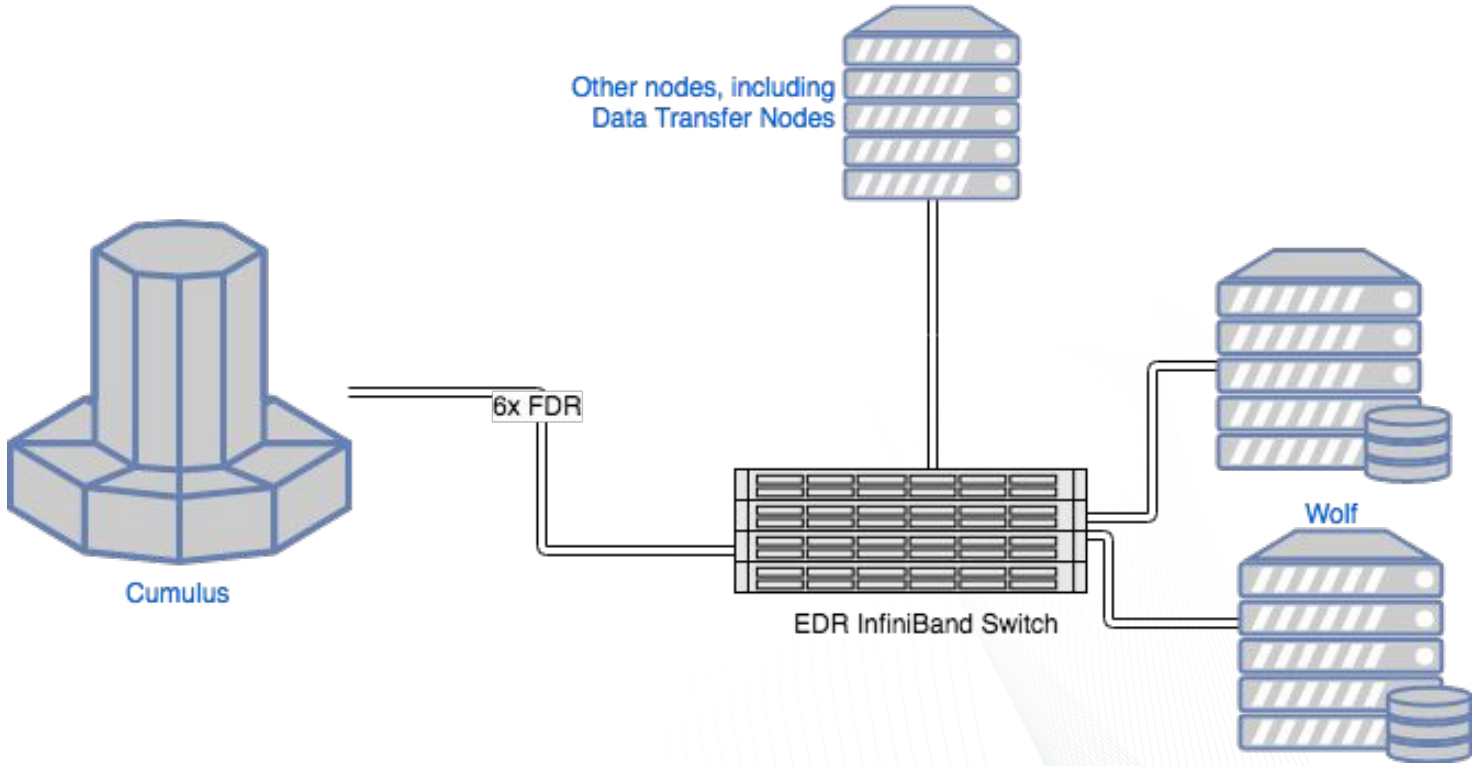
Oak Ridge National Lab

ORNL is managed by UT-Battelle
for the US Department of Energy

OAK RIDGE National Laboratory

Cumulus: An Overview

- Cray XC40
- Single Cabinet
- 112 Compute Nodes
  - 18 Core Broadwell
  - 256 GB DDR4 RAM
- 6 Service nodes with FDR InfiniBand
- 8 PB GPFS filesystem “Wolf” (running RHEL 7)



The Problem

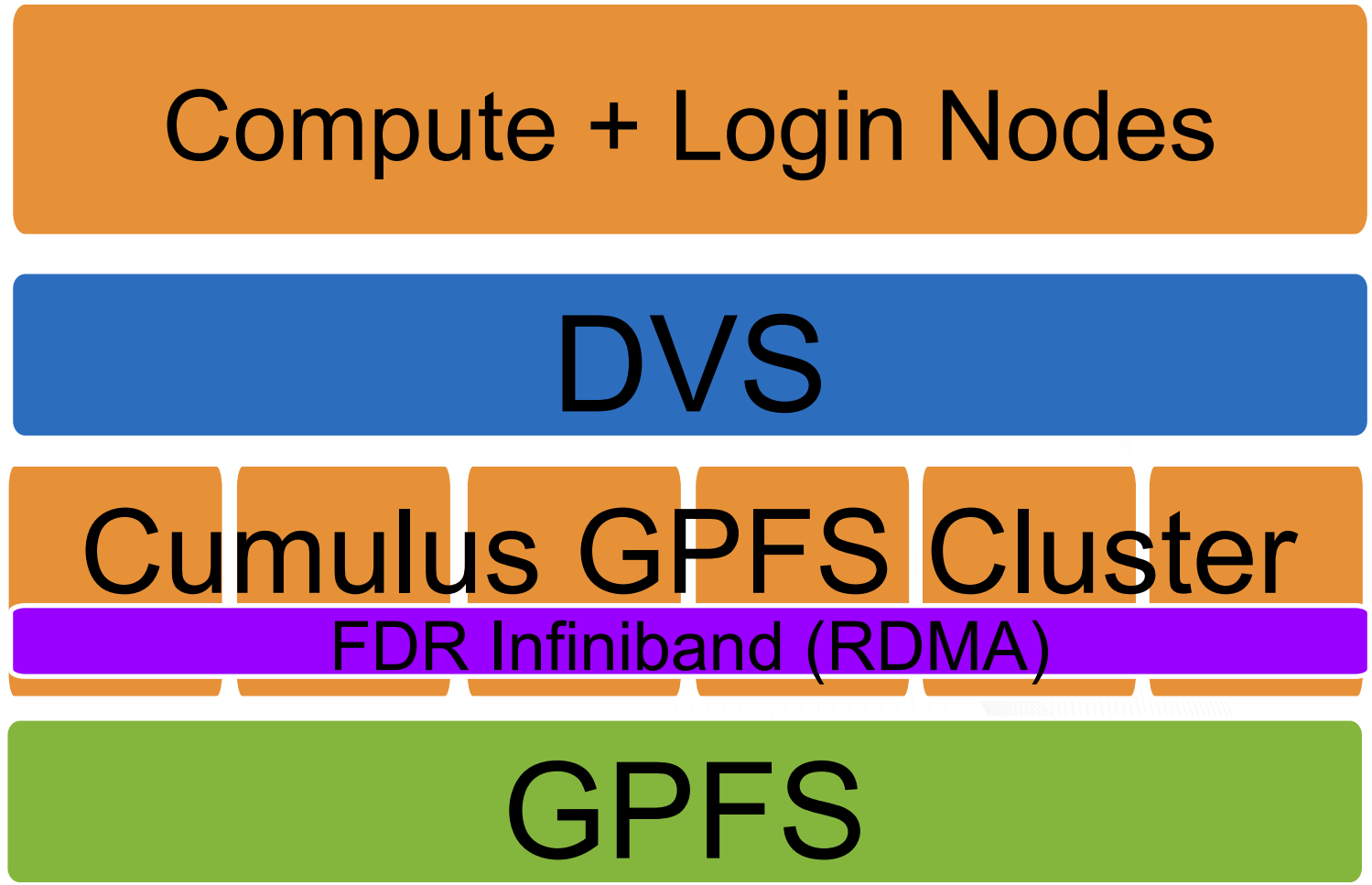
- How can we get the non-routed filesystem mounted on Compute Nodes?
- Compute nodes do not have external IO
- Remote GPFS Cluster positioning

DVS Features

- Caching
- Resilience
- Load Balancing
- Striping
- Atomicity

DVS Solution Implementation

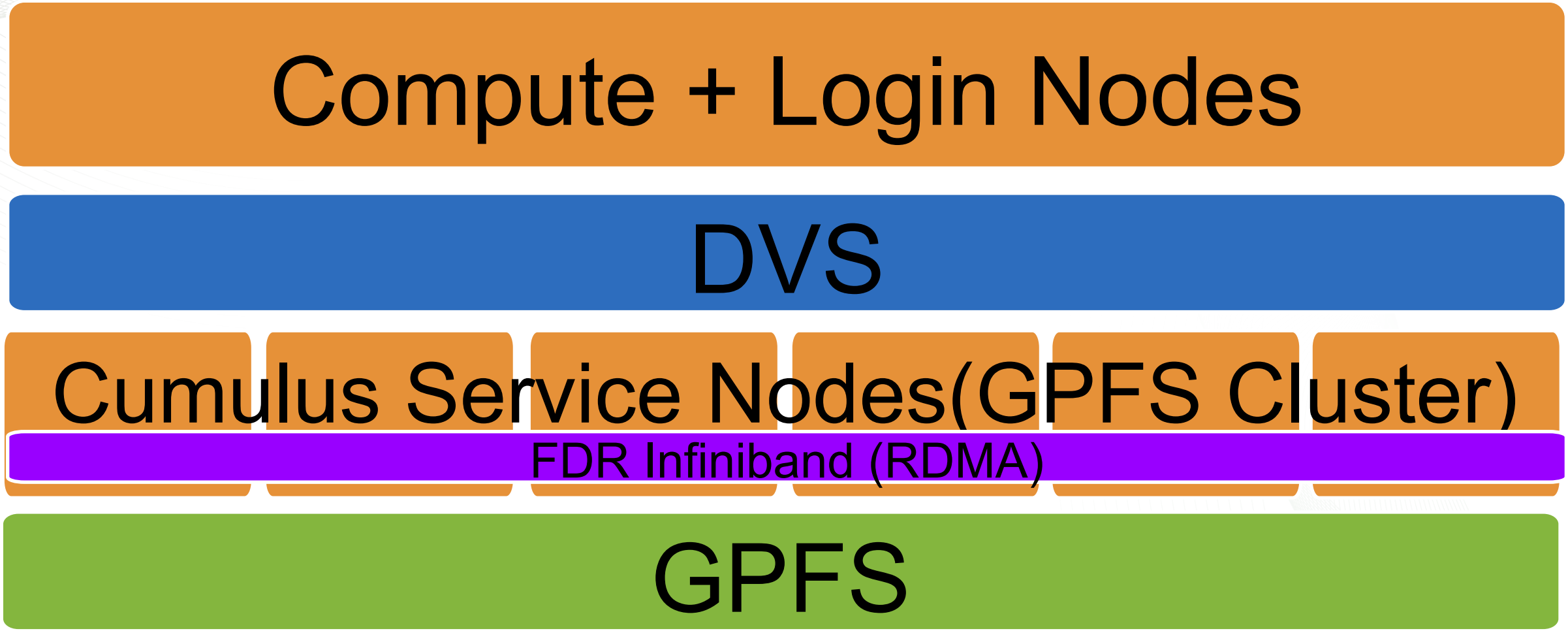
- Create remote GPFS Cluster on our 6 Service/IO nodes
- Use DVS to project GPFS to the Compute and Login Nodes



Challenges of Existing Solution

- Lack of Cache Coherency
- Lack of Posix Compliance
  - flock() system call
  - byte level locking
- Sub optimal performance on non-large I/O

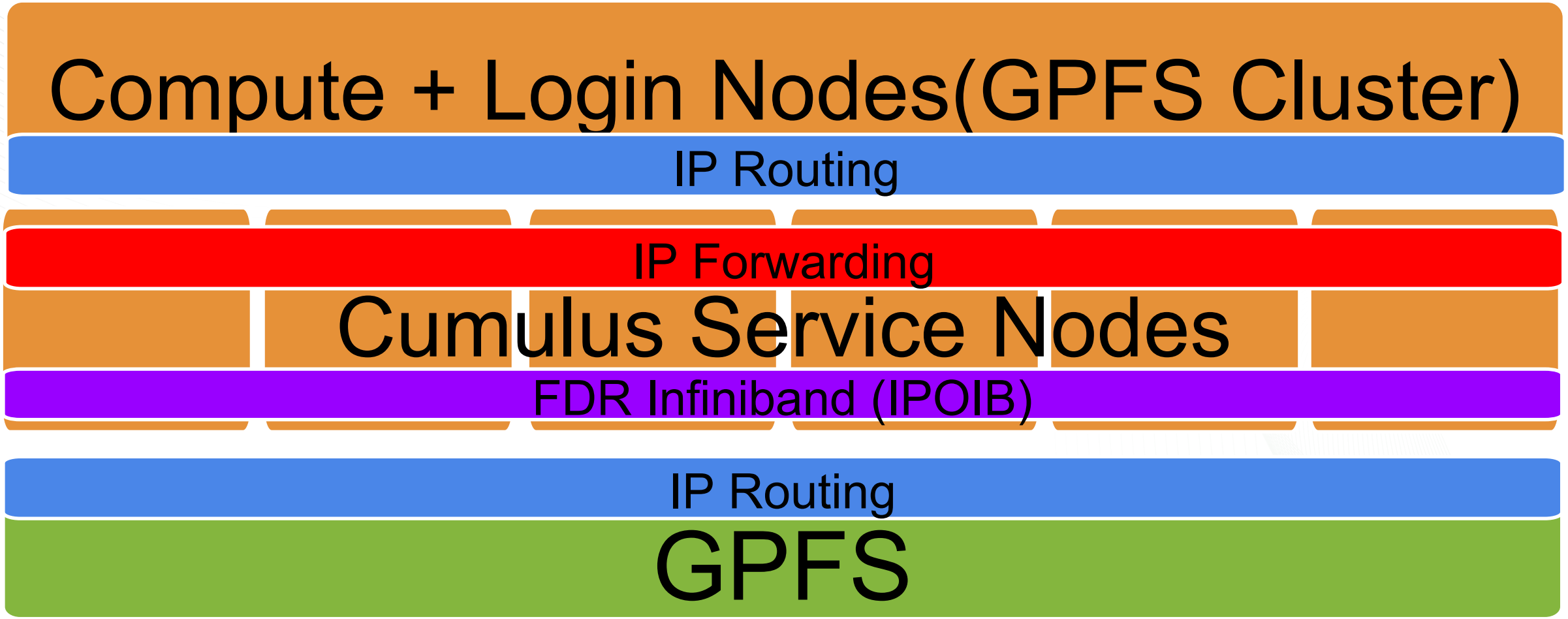
DVS Mounted GPFS



Native GPFS Solution

- Create a GPFS Cluster across all compute and login nodes
- Use IP forwarding across the six service nodes to communicate to the storage network
- Use IP routing tables on:
  - Cumulus compute and login nodes
  - Wolf GPFS Cluster

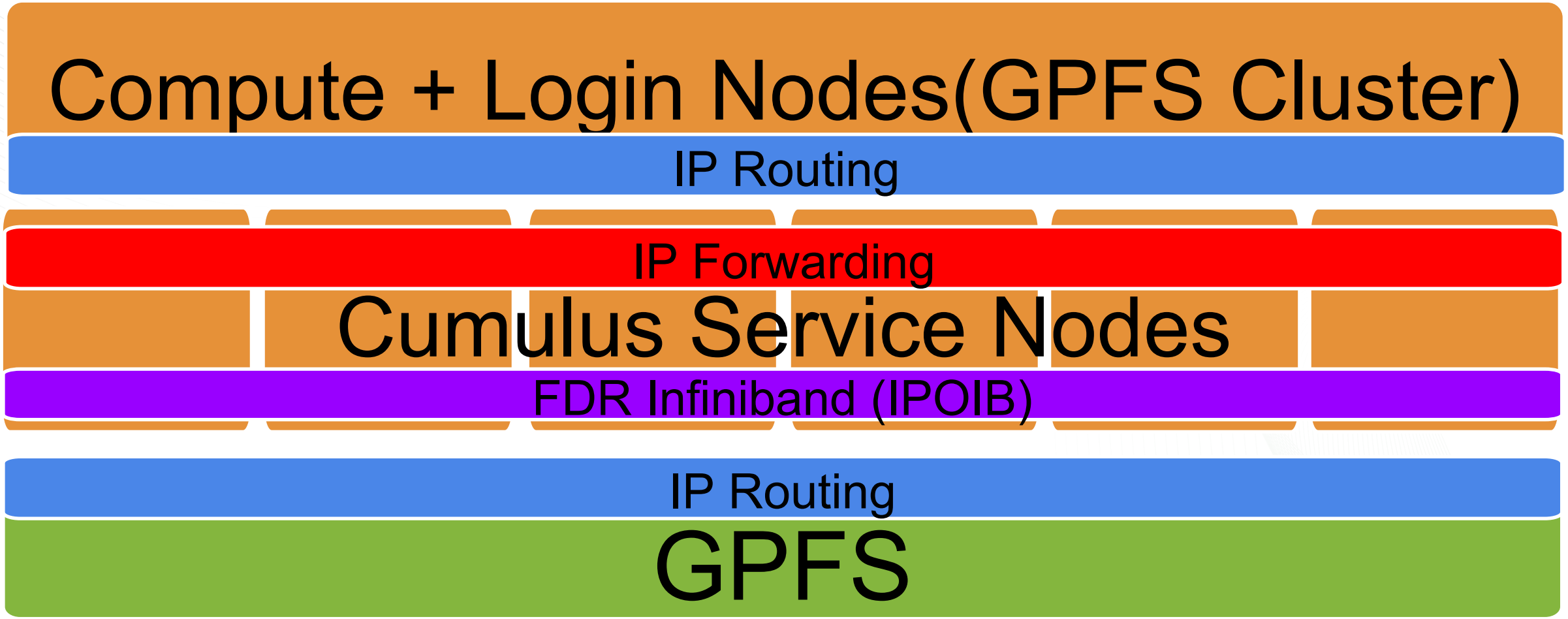
Native GPFS



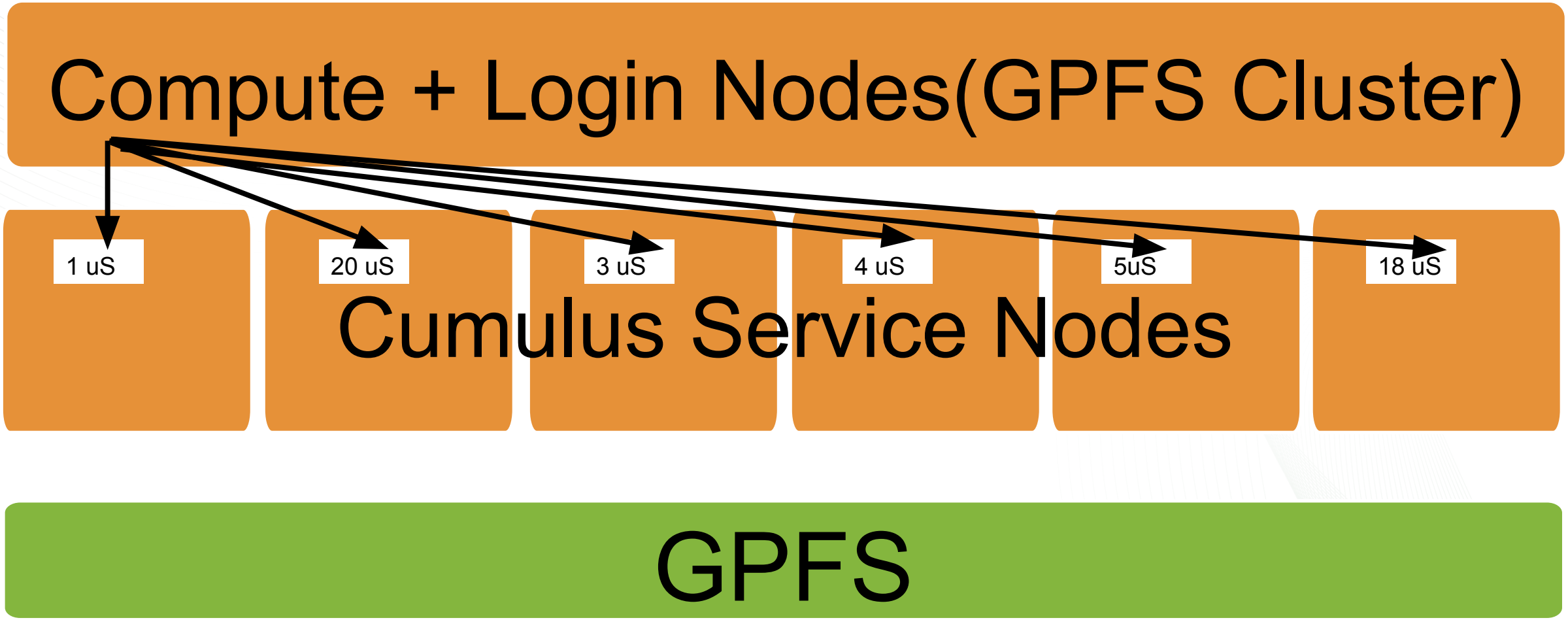
Linux Kernel Routing

- “Trie” based routing has been in the Linux kernel since version 2.6.39
  - Has been reimplemented a few times since then
- Allows number of entries in routing table to scale
  - Searches are O(L) time where L is the length of the IP Address.
- User defined routing tables
  - require kernel option to be enabled

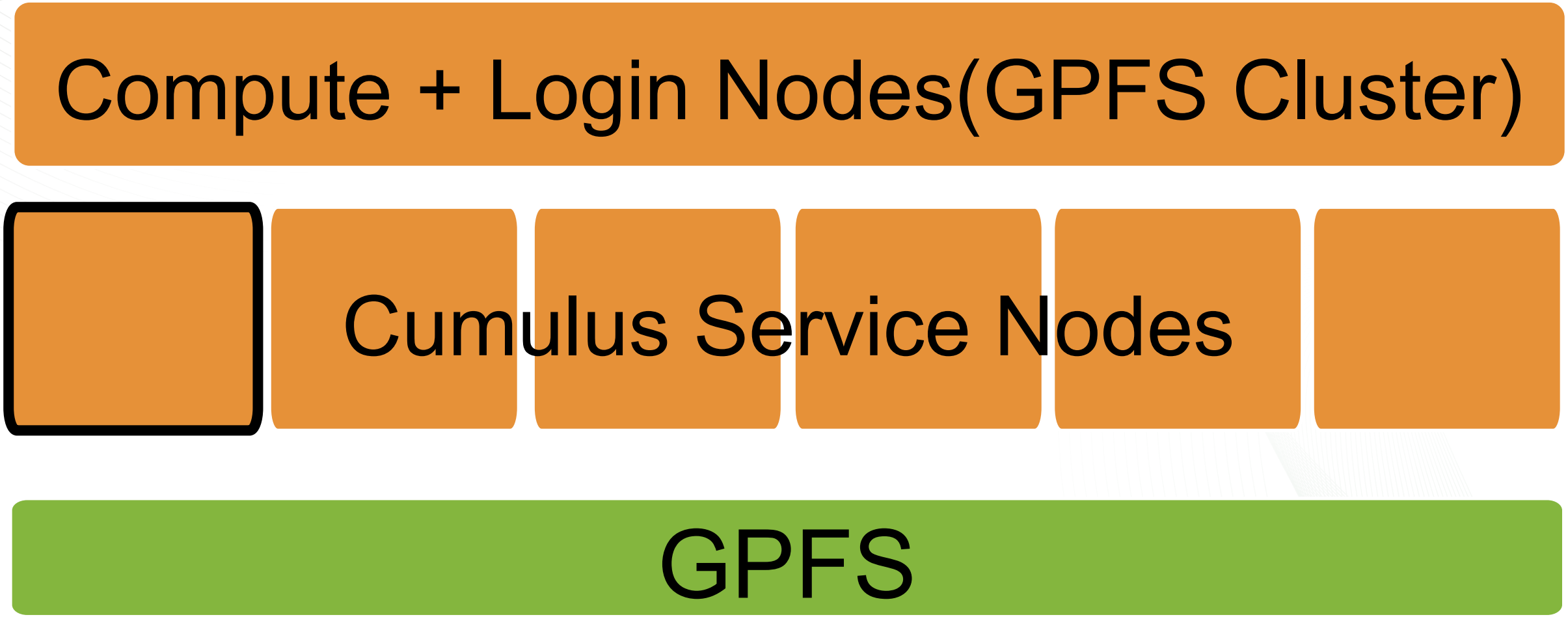
TCP Routing



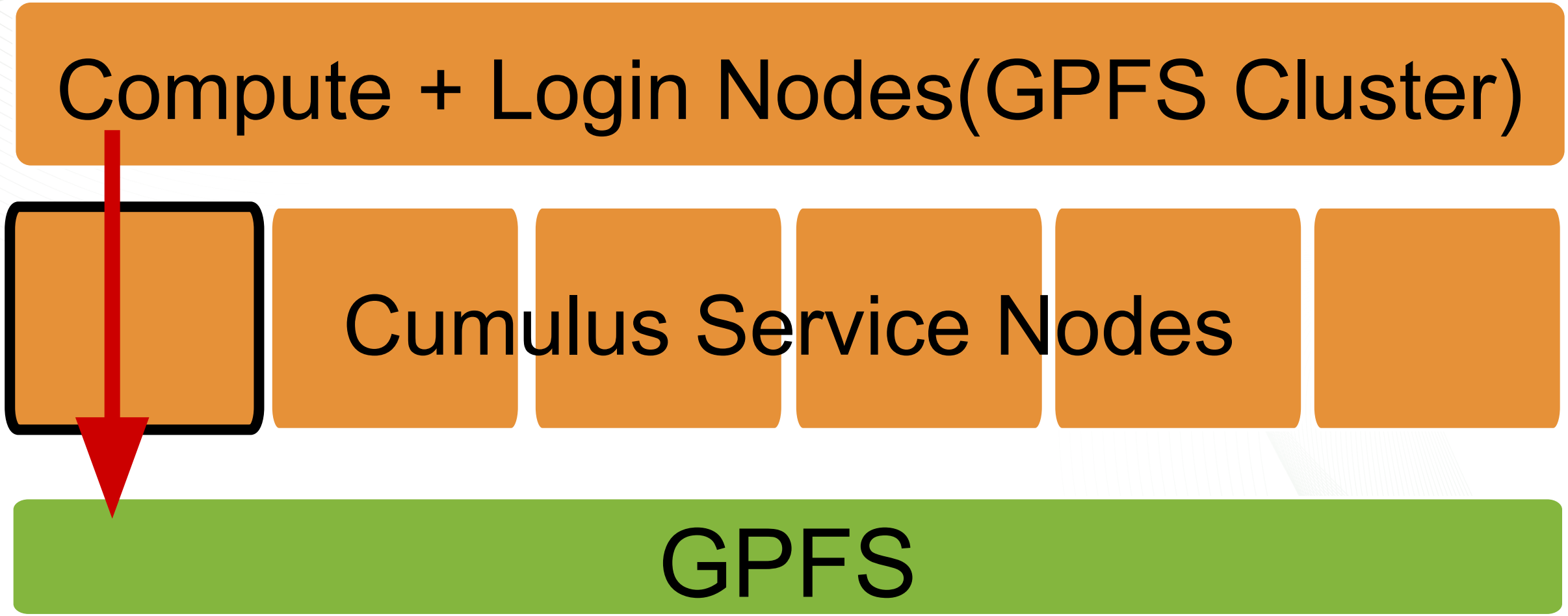
TCP Routing(Determine Latency)



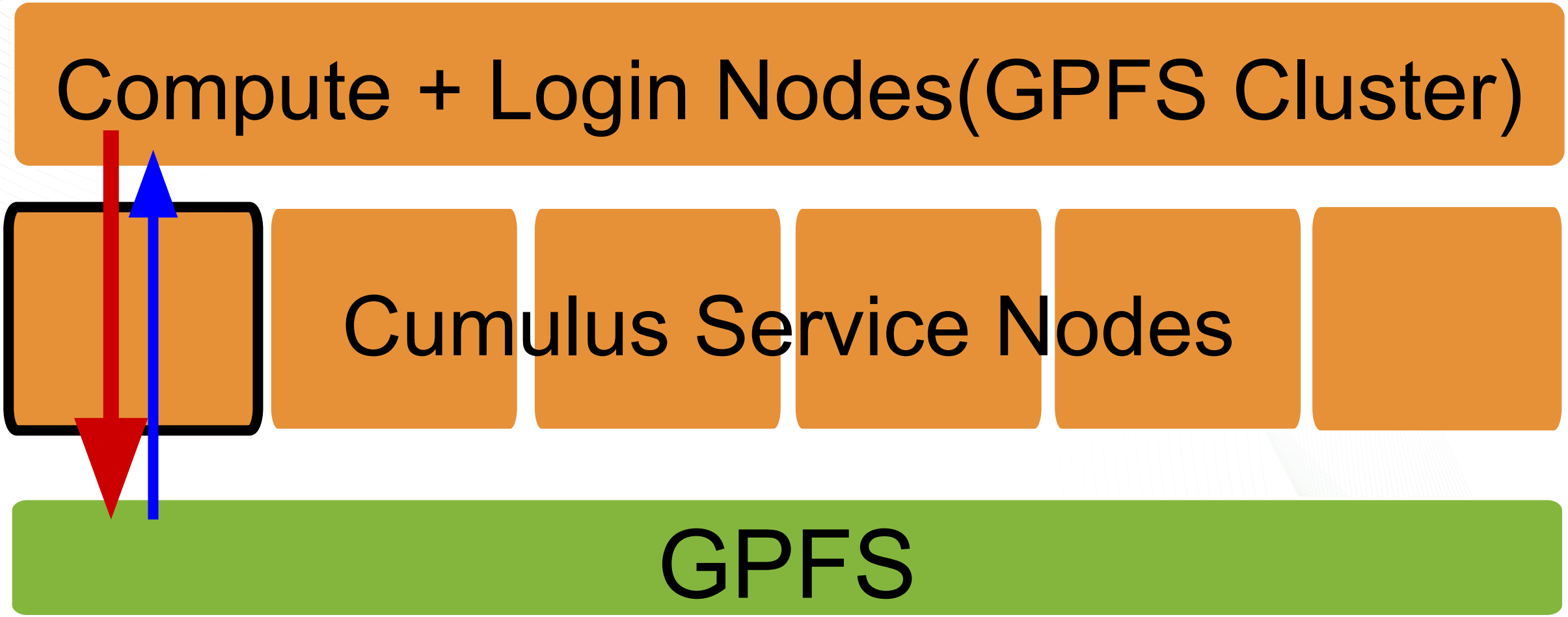
TCP Routing(Determine Latency)



TCP Routing(Outgoing Stream)



TCP Routing(Incoming Stream)



Implementation: Kernel Features

- Storage Kernel Options

```
CONFIG_IP_ROUTE_MULTIPATH=y
CONFIG_IP_MULTIPLE_TABLES=y
```

- Compute Kernel Options

```
CONFIG_IP_MULTICAST=y
CONFIG_IP_ADVANCED_ROUTER=y
CONFIG_IP_MULTIPLE_TABLES=y
CONFIG_IP_ROUTE_MULTIPATH=y
CONFIG_IP_ROUTE_VERBOSE=y
```

Implementation:
Storage Servers

- RHEL Package:
  - NetworkManager-dispatcher-routing-rules
- Each router node gets its own table
- Next-hop routing

```
# /etc/sysconfig/network-scripts/rule-interface
  from 10.10.0.11 table 1
  from 10.10.0.12 table 2
  from 10.10.0.13 table 3
  from 10.10.0.14 table 4
  from 10.10.0.15 table 5
  from 10.10.0.16 table 6
```

```
# /etc/sysconfig/network-scripts/route-interface

  10.100.0.0/16 dev ib1 scope link table 1
  default via 10.10.1.91 table 1
  10.100.0.0/16 dev ib1 scope link table 2
  default via 10.10.1.92 table 2
  10.100.0.0/16 dev ib1 scope link table 3
  default via 10.10.1.93 table 3
  10.100.0.0/16 dev ib1 scope link table 4
  default via 10.10.1.94 table 4
  10.100.0.0/16 dev ib1 scope link table 5
  default via 10.10.1.95 table 5
  10.100.0.0/16 dev ib1 scope link table 6
  default via 10.10.1.96 table 6
  10.100.0.0/16 scope global \
    nexthop dev ib1 via 10.10.0.11 \
    nexthop dev ib1 via 10.10.0.12 \
    nexthop dev ib1 via 10.10.0.13 \
    nexthop dev ib1 via 10.10.0.14 \
    nexthop dev ib1 via 10.10.0.15 \
    nexthop dev ib1 via 10.10.0.16
```

```
Cumulus Aries Network
IB Interfaces(Cumulus)
IB Interfaces(Wolf)
```

Implementation: Compute Nodes

- Similar to storage nodes
- Settings are not saved as files; routing tables added by Ansible plays

```
# ip route show table all
default via 10.100.0.91 dev ipogif0  table 1
10.10.1.0/24 dev ipogif0  table 1  scope link
default via 10.100.0.92 dev ipogif0  table 2
10.10.1.0/24 dev ipogif0  table 2  scope link
default via 10.100.0.93 dev ipogif0  table 3
10.10.1.0/24 dev ipogif0  table 3  scope link
default via 10.100.0.94 dev ipogif0  table 4
10.10.1.0/24 dev ipogif0  table 4  scope link
default via 10.100.0.95 dev ipogif0  table 5
10.10.1.0/24 dev ipogif0  table 5  scope link
default via 10.100.0.96 dev ipogif0  table 6
10.10.1.0/24 dev ipogif0  table 6  scope link
10.10.1.0/24
  nexthop via 10.100.0.91  dev ipogif0 weight 1
  nexthop via 10.100.0.92  dev ipogif0 weight 1
  nexthop via 10.100.0.93  dev ipogif0 weight 1
  nexthop via 10.100.0.94  dev ipogif0 weight 1
  nexthop via 10.100.0.95  dev ipogif0 weight 1
  nexthop via 10.100.0.96  dev ipogif0 weight 1
...
```

```
Cumulus Aries Network
IB Network()
IB Network()
```

Implementation: Compute Nodes (Incoming)

```
#cat create_routes_incoming.sh
ip rule add from 10.100.0.91 table 1
ip rule add from 10.100.0.92 table 1
ip rule add from 10.100.0.93 table 1
ip rule add from 10.100.0.94 table 1
ip rule add from 10.100.0.95 table 1
ip rule add from 10.100.0.96 table 1

ip route add 10.10.1.0/24 dev ipogif0 scope link table 1;
ip route add default via 10.100.0.91 table 1;
ip route add 10.10.1.0/24 dev ipogif0 scope link table 2;
ip route add default via 10.100.0.92 table 2;
ip route add 10.10.1.0/24 dev ipogif0 scope link table 3;
ip route add default via 10.100.0.93 table 3;
ip route add 10.10.1.0/24 dev ipogif0 scope link table 4;
ip route add default via 10.100.0.94 table 4;
ip route add 10.10.1.0/24 dev ipogif0 scope link table 5;
ip route add default via 10.100.0.95 table 5;
ip route add 10.10.1.0/24 dev ipogif0 scope link table 6;
ip route add default via 10.100.0.96 table 6;
```

Note: The iproute2 package must be installed on the compute nodes for ip commands.

```
#ip route show table all
default via 10.100.0.91 dev ipogif0  table 1
10.10.1.0/24 dev ipogif0  table 1  scope link
default via 10.100.0.92 dev ipogif0  table 2
10.10.1.0/24 dev ipogif0  table 2  scope link
default via 10.100.0.93 dev ipogif0  table 3
10.10.1.0/24 dev ipogif0  table 3  scope link
default via 10.100.0.94 dev ipogif0  table 4
10.10.1.0/24 dev ipogif0  table 4  scope link
default via 10.100.0.95 dev ipogif0  table 5
10.10.1.0/24 dev ipogif0  table 5  scope link
default via 10.100.0.96 dev ipogif0  table 6
10.10.1.0/24 dev ipogif0  table 6  scope link
10.10.1.0/24
  nexthop via 10.100.0.91  dev ipogif0 weight 1
  nexthop via 10.100.0.92  dev ipogif0 weight 1
  nexthop via 10.100.0.93  dev ipogif0 weight 1
  nexthop via 10.100.0.94  dev ipogif0 weight 1
  nexthop via 10.100.0.95  dev ipogif0 weight 1
  nexthop via 10.100.0.96  dev ipogif0 weight 1
...
```

Implementation: Compute Nodes (Outgoing)

```
#cat create_routes_outgoing.sh
ip route add 10.40.1.0/24 scope global \
  nexthop dev ipogif0 via 10.132.0.7 \
  nexthop dev ipogif0 via 10.132.0.67 \
  nexthop dev ipogif0 via 10.132.0.31 \
  nexthop dev ipogif0 via 10.132.0.83 \
  nexthop dev ipogif0 via 10.132.0.94 \
  nexthop dev ipogif0 via 10.132.0.95
```

Note: The iproute2 package must be installed on the compute nodes for ip commands.

```
#ip route show table all
default via 10.100.0.91 dev ipogif0  table 1
10.10.1.0/24 dev ipogif0  table 1  scope link
default via 10.100.0.92 dev ipogif0  table 2
10.10.1.0/24 dev ipogif0  table 2  scope link
default via 10.100.0.93 dev ipogif0  table 3
10.10.1.0/24 dev ipogif0  table 3  scope link
default via 10.100.0.94 dev ipogif0  table 4
10.10.1.0/24 dev ipogif0  table 4  scope link
default via 10.100.0.95 dev ipogif0  table 5
10.10.1.0/24 dev ipogif0  table 5  scope link
default via 10.100.0.96 dev ipogif0  table 6
10.10.1.0/24 dev ipogif0  table 6  scope link
10.10.1.0/24
  nexthop via 10.100.0.91  dev ipogif0 weight 1
  nexthop via 10.100.0.92  dev ipogif0 weight 1
  nexthop via 10.100.0.93  dev ipogif0 weight 1
  nexthop via 10.100.0.94  dev ipogif0 weight 1
  nexthop via 10.100.0.95  dev ipogif0 weight 1
  nexthop via 10.100.0.96  dev ipogif0 weight 1
...
```

Tuning

- IP Tuning
  - Optimize TCP buffers
- Infiniband Tuning
  - Connected Mode
  - 65k MTU

```
net.ipv4.tcp_timestamps=0
net.ipv4.tcp_sack=1
net.ipv4.tcp_low_latency=1
net.ipv4.tcp_adv_win_scale=1
net.core.netdev_max_backlog=250000
net.core.rmem_max=536870912
net.core.wmem_max=536870912
net.core.optmem_max=536870912
net.ipv4.tcp_mem=4096 87380 268435456
net.ipv4.tcp_rmem=4096 87380 268435456
net.ipv4.tcp_wmem=4096 87380 268435456
net.ipv4.tcp_no_metrics_save=1
net.core.default_qdisc=fq
net.ipv4.tcp_congestion_control=htcp
net.ipv4.tcp_mtu_probing=1
net.ipv4.tcp_fin_timeout=30
net.ipv4.tcp_tw_recycle=1
net.ipv4.tcp_tw_reuse=1
```

IP Tuning- Wolf Configuration

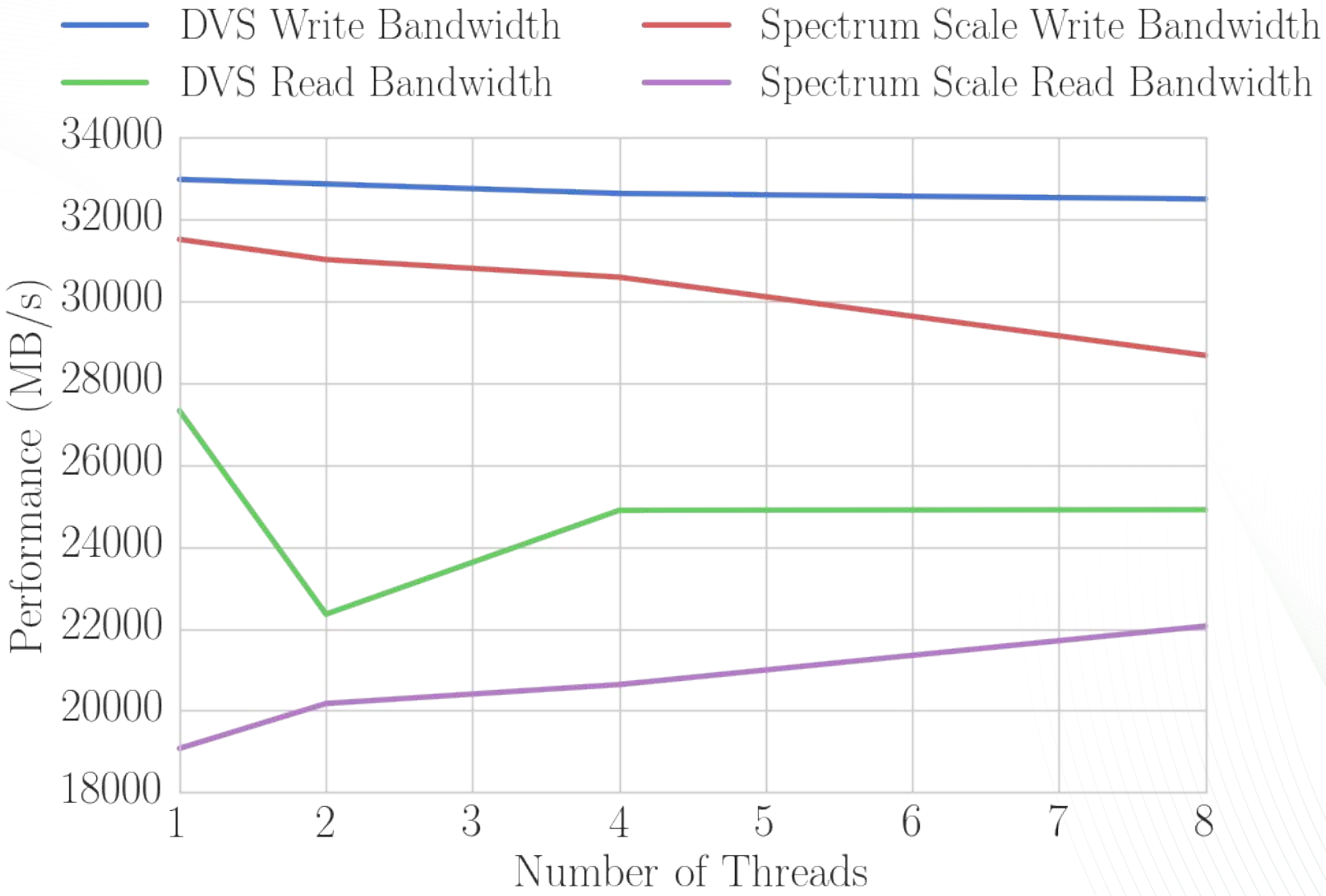
Performance Changes - Single Client

- Streaming read performance increased
  - ~2x single-process read performance, up to ~9.3 GB/s with 4 processes
- Multi-process write increase
  - ~5% performance hit with single process
  - ~15% performance increase with higher process count
- Metadata rates (file creation, stat, and removal) increase
  - Orders of magnitude faster with 0-length files
  - 3x+ increase displayed with 1k, 4k, and 32k length files

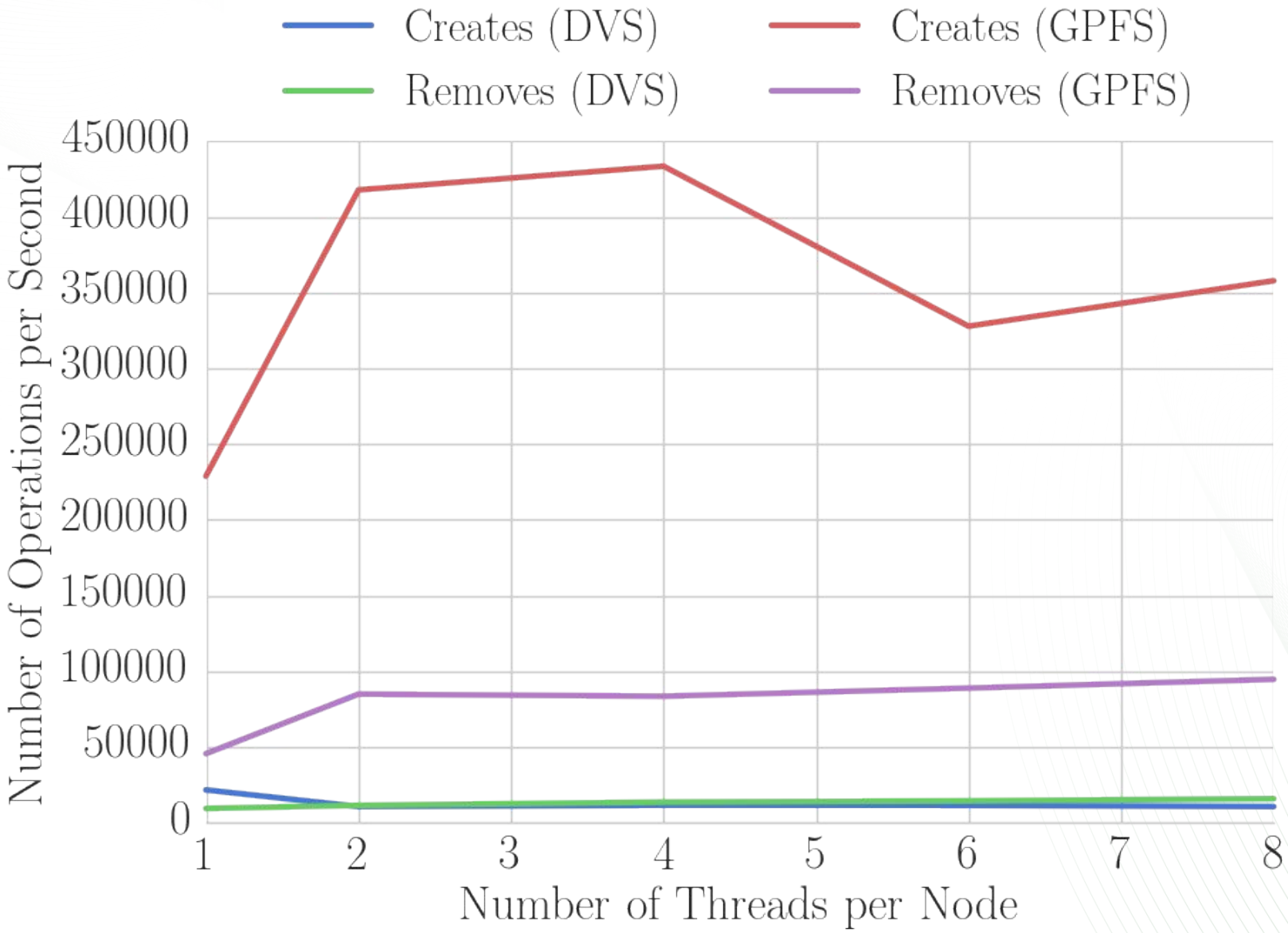
Performance Changes - Multi-client

- DVS showed higher peak performance
- Native shows a reduction in total system bandwidth
- Metadata performance using Native method out-scales DVS

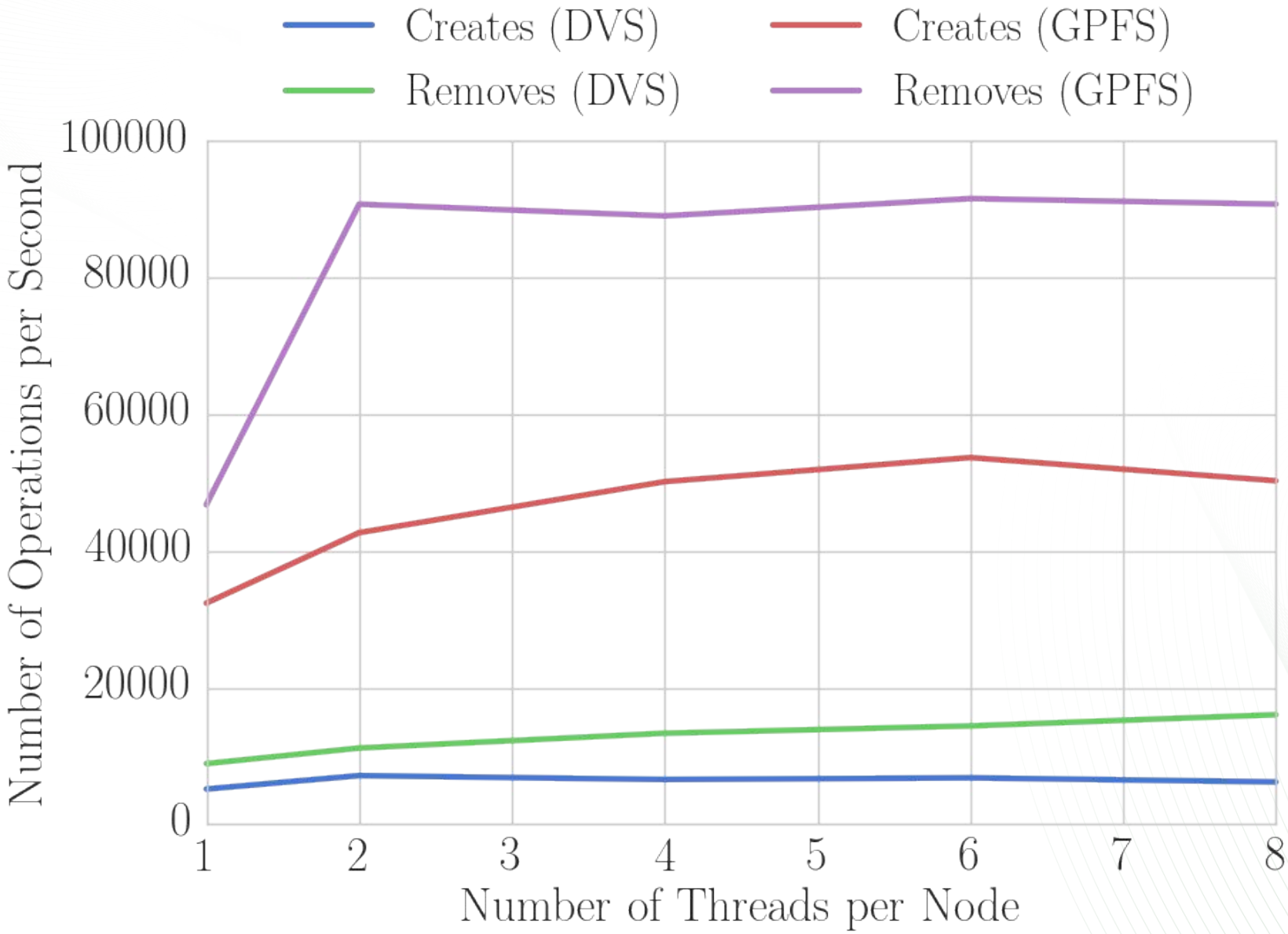
Streaming IO 64 Nodes



Metadata Performance (64 Nodes, 0 Byte Files)



Metadata Performance (64 Nodes, 32 KB Files)



Gains

- Fully posix compliant
- Much better metadata performance

Trade Offs

- Potential for jitter in I/O communication
- Reduced memory availability on compute nodes
- Lower Large Streaming I/O Performance

Future Work

- Further tuning
- Benchmarking of user codes
- Adaptive routing behavior
- Failover
- Scaling

Conclusion

- While there are some tradeoffs using the native method, it fits our users' workload.
- Cumulus uses the Native GPFS mounting method in production.

Questions?

- Jesse Hanley
  - hanleyja@ornl.gov
- Chris Muzyn
  - muzyncj@ornl.gov