Network Integration of Perlmutter at NERSC

NERSC

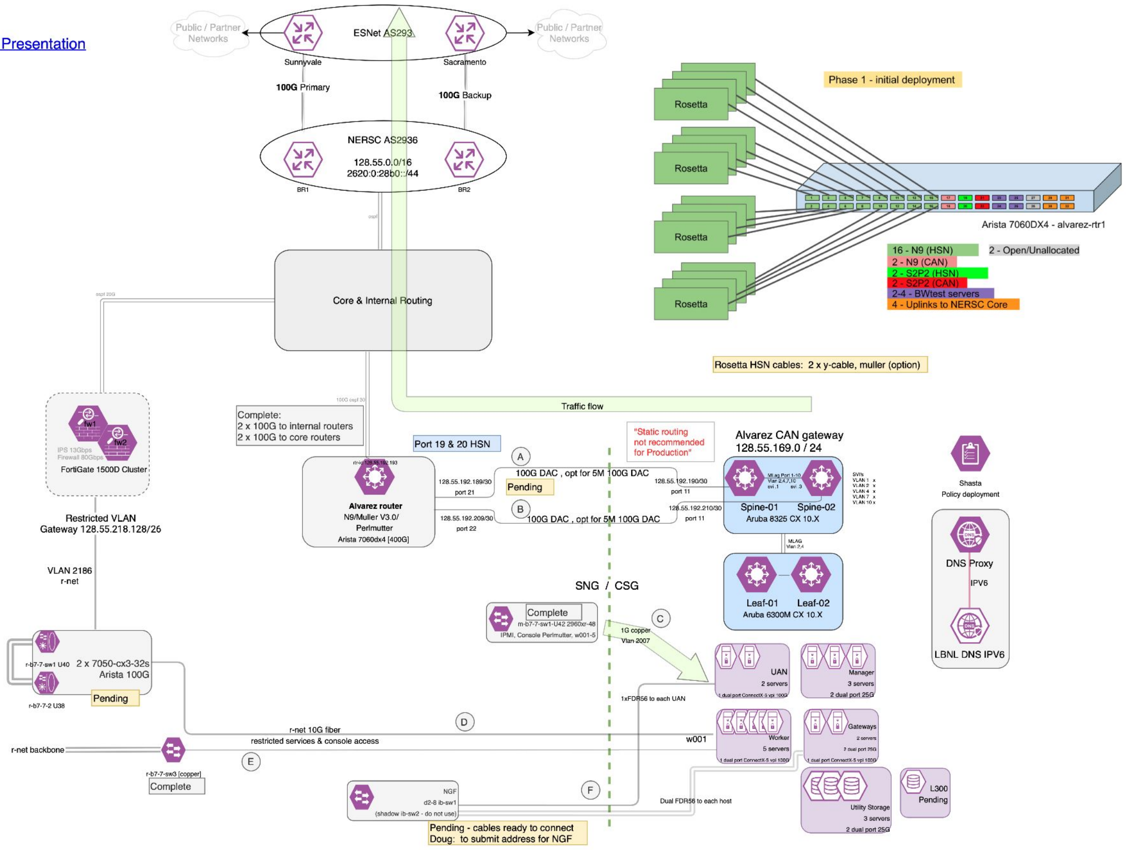
Ershaad A. Basheer, Eric Roman, Tavia Stone Gibbins, Christopher Samuel, Lisa Gerhardt, Ashwin Selvarajan, Damian Hazen, Douglas M. Jacobsen, Ronal Kumar
Lawrence Berkeley National Laboratory
May 5, 2022

Introduction

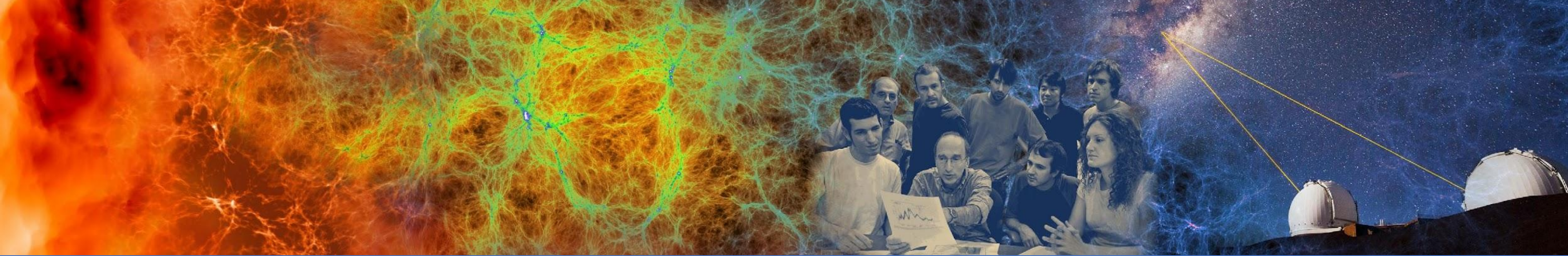
- Perlmutter
  - NERSC's next generation supercomputer
  - Named in honor of Lab’s Nobel Prize-winning astrophysicist Saul Perlmutter
  - HPE Cray EX "Shasta" platform
- Shasta system management network: SMNet
  - Logically split: Hardware Management, Node Management and Customer Access Network
- Slingshot high speed network: HSN
  - Message passing interconnect
  - Access to attached and site-wide filesystems
  - Login access

s2p2 aka Alvarez (Transit Routing, CAN, HSN, r-net, m-net) - SNG V3.0 01/05/2021

NERSC 9 Presentation



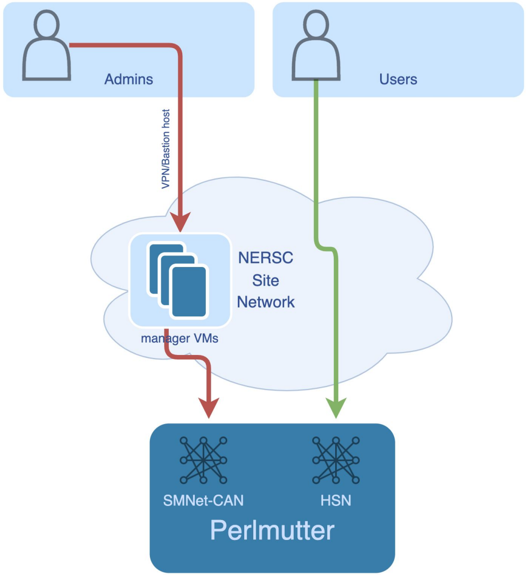
BERKELEY LAB
Bringing Science Solutions to the World
U.S. DEPARTMENT OF ENERGY
Office of Science



Privatized Customer Access Network

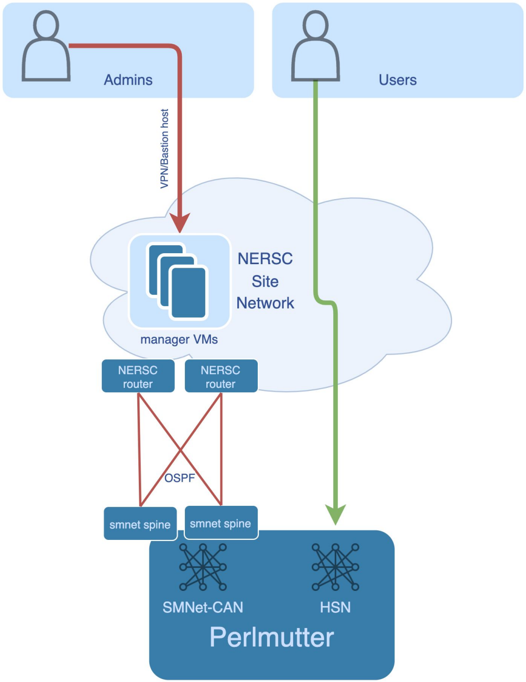
Customer Access Network

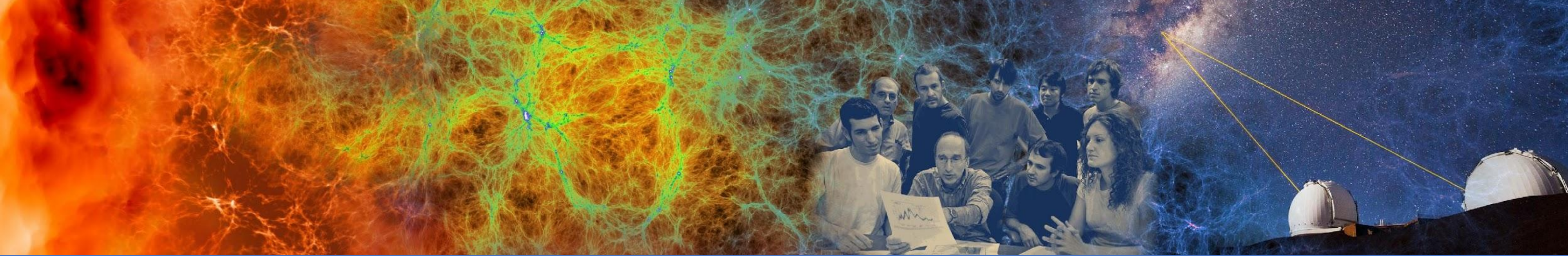
- Originally, Shasta CAN network allows users access from external networks
  - User Logins, Access to REST APIs, containerized logins etc.
  - Separate VLAN from HMN and NMN
- NERSC's security policies require administrative access be restricted to MFA-authenticated networks
  - We restrict users from accessing the CAN entirely
  - User traffic is routed through the edge routers to HSN
- Access to API gateway limited to dedicated management VM outside system
  - All sysadmin tasks and management workflows run in VM as normal user
  - Including CLI tools that interact with APIs like `cray` and `kubectl` commands
  - SSH to Compute Nodes utilize `m001` as jump host



Customer Access Network

- Employ point-to-point routes between the two SMNet spines and our data center routers
- Exchange routes via OSPF
- Use a separate CAN VRF for the data center routes to avoid advertising RFC 1918 routes to data center
    - Required adjusting BGP configuration
- This setup allows flexibility in managing data center and SMNet routers

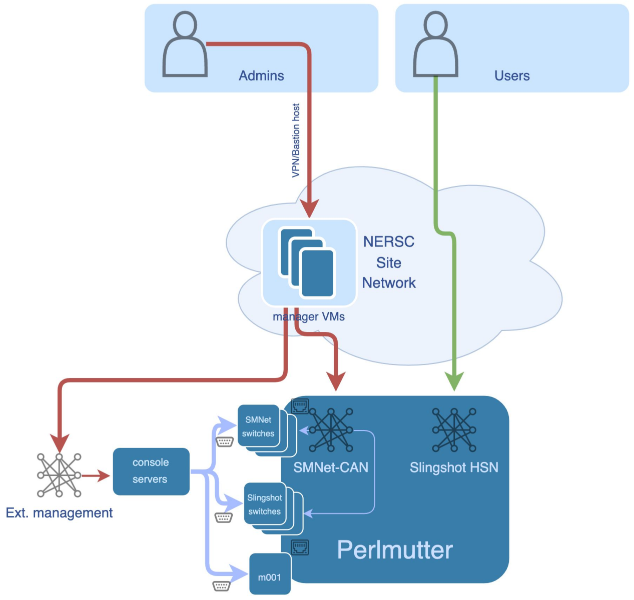


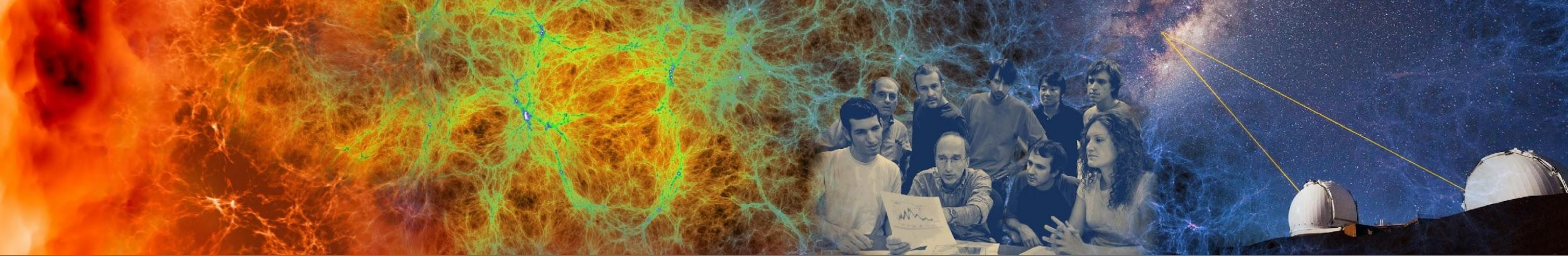


External Management Network

External Management Network

- Site access restricted
- SMNet switch configuration is in-band
- We set up a secondary management network for when the CAN is not available
- Especially valuable during bootstrap and system install
    - Out of band access to
        - SMNet switch management ports. Access to serial ports of SMNet and River Slingshot switches via console servers
        - BMC port of first/bootstrap management server (`m001`)
- Isolated network only accessible from management VM
- Allows us to recover from SMNet misconfigurations
- Also, allows
    - Access to switch logs
    - Switch firmware updates
    - Collection of switch debug data





External DNS service

Background

- LBL handles `nersc.gov` domain
- Shasta Kubernetes cluster can host services that are exposed as DNS entries in the `perlmutter.nersc.gov` domain
  - Currently handled by CoreDNS which reads entries from etcd and accessed over CAN
    - CAN is IPv4 only at this time
    - LBL requires DNSSEC. Not supported in default configuration
    - no AXFR or IXFR

Initial Deployment

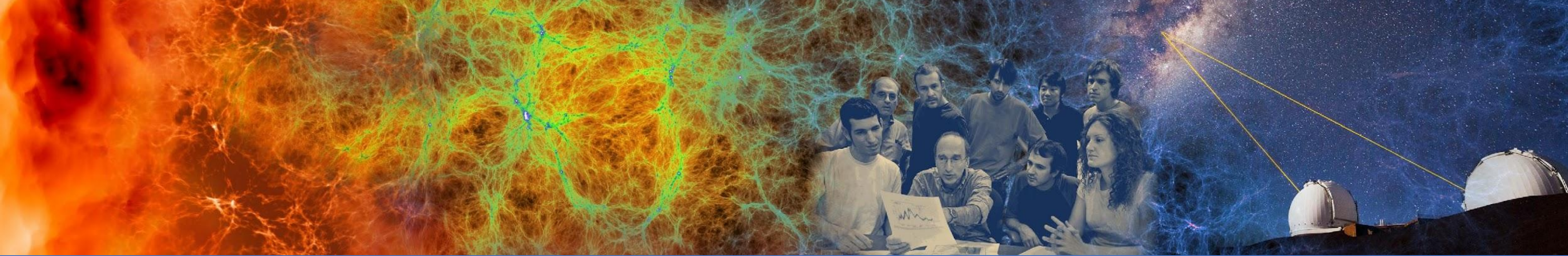
- Agreed that queries to delegated domain are limited to NERSC internal network
- Login node entries added directly to LBL-controlled `nersc.gov` domain
- Worked with HPE to modify CoreDNS deployment to accept DNSSEC keys from Vault
- Still not sufficient to expose DNS externally
    - Two options being considered

External DNS service

- Dynamically update an external DNS service
- LBL handles external DNS
- Service that will use `nsupdate` to synchronize entries with LBL
- Will support DNSSEC, IPv6, Secondary servers
- No CSM changes
- Disadvantages
    - Would require polling to detect changes
    - Stale entries would need to be cleaned up by periodically scanning entire external zone

External DNS service

- Use zone transfer instead of `nsupdate`
- CoreDNS etcd plugin does not support zone transfers
- Zone transfer keeps all entries in sync
- Secondary servers are also kept in sync in the process
- IPv6 already supported
- Disadvantages
  - CSM support required
  - Need to handle DNSSEC duties
  - CSM would need to handle key rotation and separate zone and key signing key support



Storage Gateway and Login Load-balancing

NERSC BERKELEY LAB U.S. DEPARTMENT OF ENERGY Office of Science

Background

- Storage systems that Perlmutter has access to
    - System-attached Lustre filesystem
    - NERSC's shared GPFS filesystems
        - Home directories
        - Community File System
    - HPSS Archive
- GPFS servers and clients interconnected by Infiniband fabric
- Ethernet-only clients connect to GPFS over TCP/IP
    - Need Ethernet-IPoIB gateways to route traffic between fabrics
    - No RDMA
- NERSC uses DVS to project GPFS filesystems to computes on XC systems
- Wished to evaluate native GPFS mounts on computes

Requirements

- Existing Infrastructure
  - GPFS servers and clients interconnected by Infiniband FDR fabric
  - Direct mount GPFS on all Slingshot connected compute nodes in Cray EX
    - Requires Slingshot-Infiniband gateways
  - Perlmutter has 24 service/gateway servers
    - 2x Slingshot NICs
    - 2x Infiniband HCAs
- Requirements
  - Has to be resilient to multiple gateways failures, including link failures
  - Traffic needs to be load-balanced across the gateways

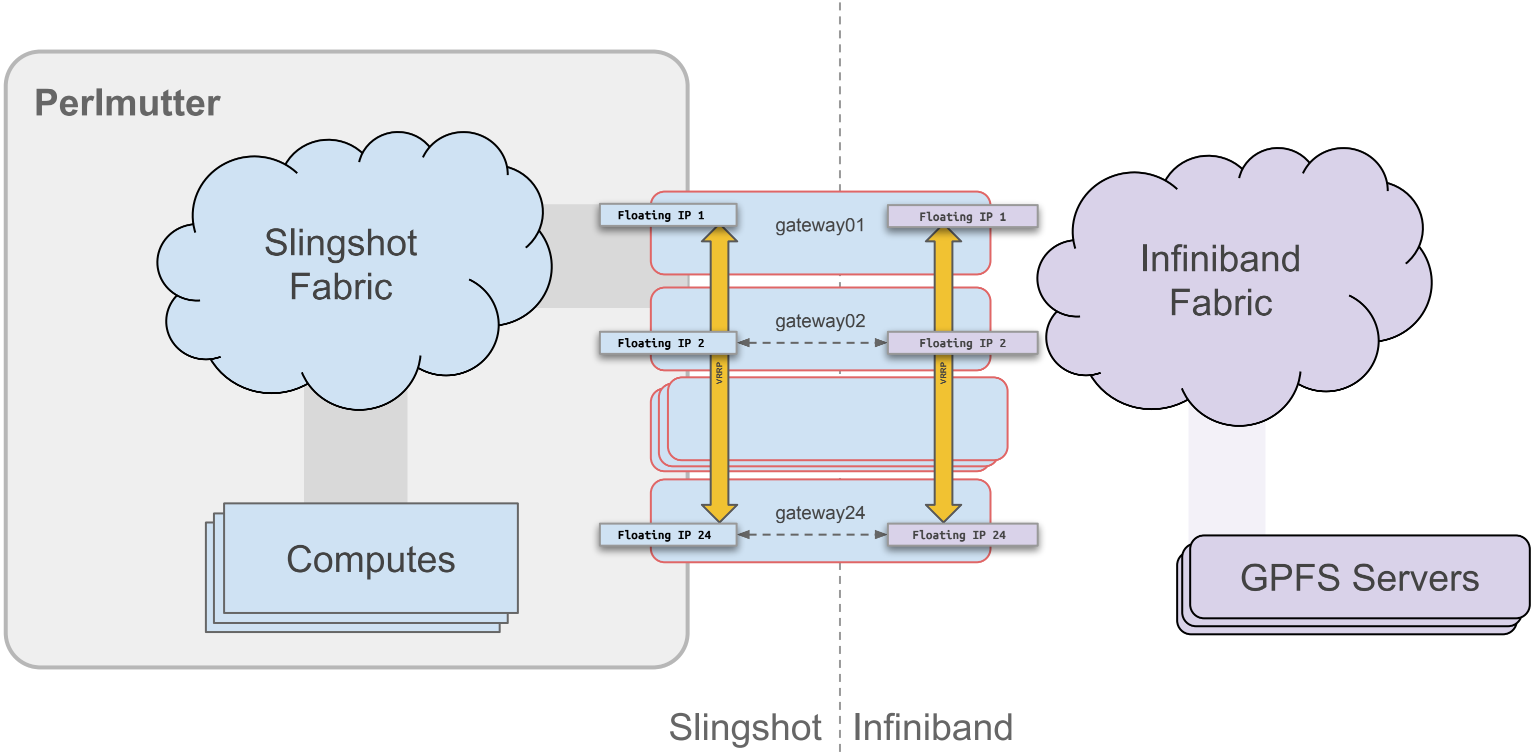
Design

- Gateway resiliency
  - Compute nodes use gateway IPs as the nexthop to reach GPFS servers
  - We need to be able to handle gateway nodes going offline
    - Without complicated dynamic route distribution to computes
  - Instead we keep gateway IPs reachable by failing them over when a gateway is offline

Design

- VRRP
  - Keepalived is a software implementation of the VRRP protocol
  - Each of the 24 gateways is assigned a Floating/Virtual IP address
  - Each VIP is attached to a VRRP redundancy group spanning all gateways
    - A VIP can failover to any of the other 23 gateways
    - Assign priorities randomly within each group, so that groups of IPs don't gather on the same gateway

Design


Perlmutter
Slingshot Fabric
Computes
Floating IP 1
gateway01
Floating IP 1
gateway02
Floating IP 2
Floating IP 2
VRRP
VRRP
gateway24
Floating IP 24
Floating IP 24
Infiniband Fabric
GPFS Servers
Slingshot
Infiniband

Design

- Compute node routing tables
  - Possible options
    - Static assignments to a particular gateway
    - Dynamic routing daemon
  - ECMP: Equal Cost Multipath
    - Allow multiple nexthops per routing table entry
    - Every compute node has identical table that includes all 24 gateway VIPs
    - Nexthop is chosen based on L3+L4 hash
      - source/dest IP, protocol, source/dest port
      - reduces packet reordering within TCP streams
    - Balances load by GPFS server
      - One socket per client-server pair

```
128.55.148.0/22 src 10.249.11.112 mtu 2044
        nexthop via 10.249.252.0 dev hsn0 weight 1
        nexthop via 10.249.252.1 dev hsn0 weight 1
        nexthop via 10.249.252.2 dev hsn0 weight 1
        nexthop via 10.249.252.3 dev hsn0 weight 1
        nexthop via 10.249.252.4 dev hsn0 weight 1
        nexthop via 10.249.252.5 dev hsn0 weight 1
        nexthop via 10.249.252.6 dev hsn0 weight 1
        nexthop via 10.249.252.7 dev hsn0 weight 1
        nexthop via 10.249.252.8 dev hsn0 weight 1
        nexthop via 10.249.252.9 dev hsn0 weight 1
        nexthop via 10.249.252.10 dev hsn0 weight 1
        nexthop via 10.249.252.11 dev hsn0 weight 1
        nexthop via 10.249.252.12 dev hsn0 weight 1
        nexthop via 10.249.252.13 dev hsn0 weight 1
        nexthop via 10.249.252.14 dev hsn0 weight 1
        nexthop via 10.249.252.15 dev hsn0 weight 1
        nexthop via 10.249.252.16 dev hsn0 weight 1
        nexthop via 10.249.252.17 dev hsn0 weight 1
        nexthop via 10.249.252.18 dev hsn0 weight 1
        nexthop via 10.249.252.19 dev hsn0 weight 1
        nexthop via 10.249.252.20 dev hsn0 weight 1
        nexthop via 10.249.252.21 dev hsn0 weight 1
        nexthop via 10.249.252.22 dev hsn0 weight 1
        nexthop via 10.249.252.23 dev hsn0 weight 1
```

Login Node Load-balancing

- Perlmutter has 40 login nodes that users access with SSH
- Requirements
  - Balance SSH login sessions across all the login nodes
  - Spread ingress SSH traffic across all nodes.
    - In contrast with Cori which has one dedicated load balancer that handles all ingress traffic.
  - Minimize disruption of established SSH sessions if nodes fail

Login Node Load-balancing

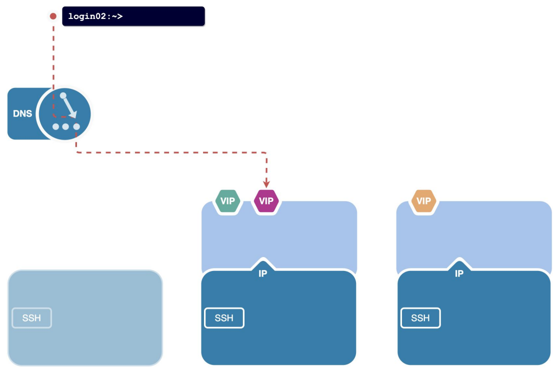
- Used the same technique of using VIPs to build the load-balancing solution
- Load-balancer setup made up of 3 components
  - DNS
  - VRRP/Keepalived
  - IPVS

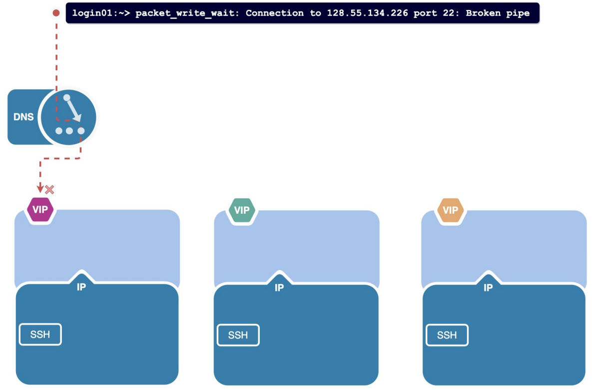
Login Node Load-balancing

- DNS
  - `login.perlmutter.nersc.gov` resolves to 40 IP address
    - Returned in random order
    - Client decides which IP is used for connecting
  - No quick way to remove a login node from the load-balancer
    - DNS caching
  - Progress can be very slow if many login nodes are down
    - Client tries each IP in turn
    - A hanging connection can make it difficult to connect to a working node

Login Node Load-balancing

- VRRP/Keepalived
  - Each login node has a Virtual IP assigned to it
  - VIP of an unhealthy or offline login nodes is taken over by another node
    - Node can be removed from the load-balancer by stopping Keepalived
    - or node fails a health check
  - As long as one node is online, every client will be able to connect
  - Flawed handling of failback



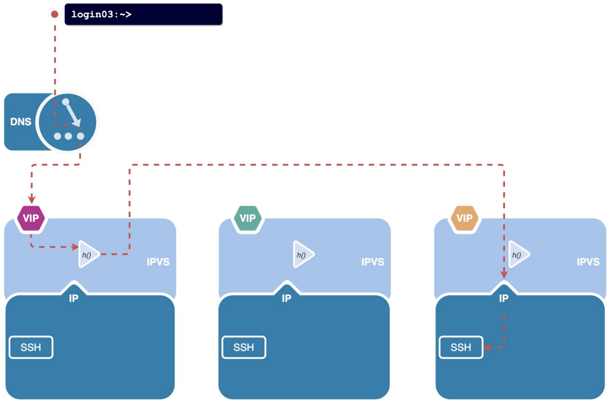


Login Node Load-balancing

- IPVS
  - IP Virtual Server implements transport-layer (TCP/UDP) load-balancing inside the Linux kernel
  - IPVS exposes a virtual service IP
    - Connections to virtual service are redirected to replicated backend called real servers
    - Backend servers chosen by a configurable scheduler
    - Various redirection methods available
      - Including direct routing/gatewaying. Packet forwarded with destination MAC of chosen backend server

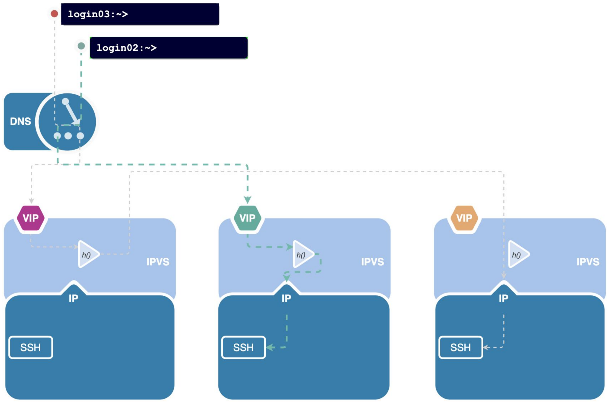
Login Node Load-balancing

- IPVS Source hashing
  - Scheduler chooses backend based on hash of source IP address and source port
- Every login node is initialized with an identical hash table



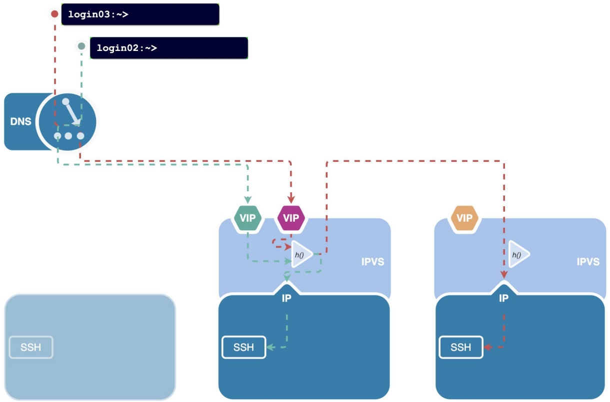
Login Node Load-balancing

- IPVS Source hashing
  - Scheduler chooses backend based on hash of source IP address and source port
- Every login node is initialized with an identical hash table
  - Connection endpoint based on client IP and port
  - Regardless of VIP/virtual service of connection



Login Node Load-balancing

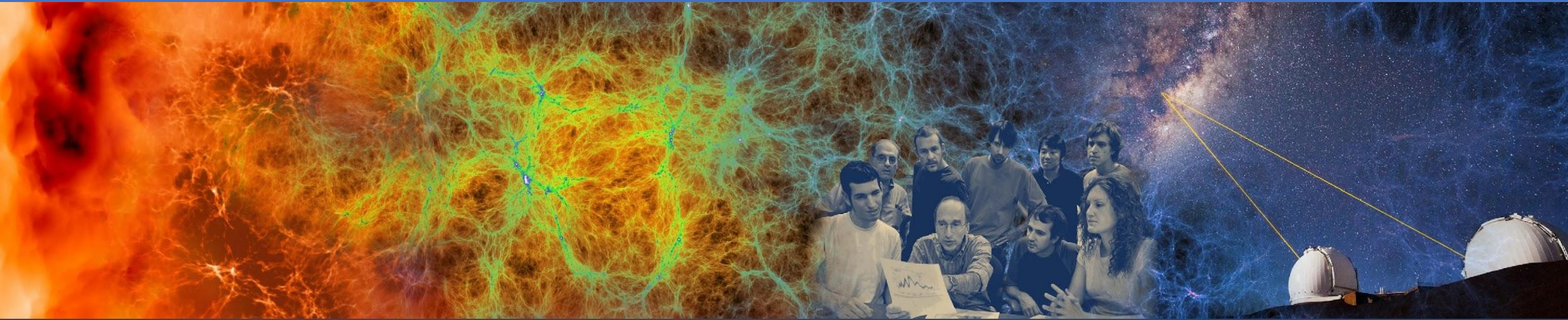
- IPVS Source hashing
  - Scheduler chooses backend based on hash of source IP address and source port
- Every login node is initialized with an identical hash table
  - Connection endpoint based on client IP and port
  - Regardless of which VIP/virtual service the client connected to



Login Node Load-balancing

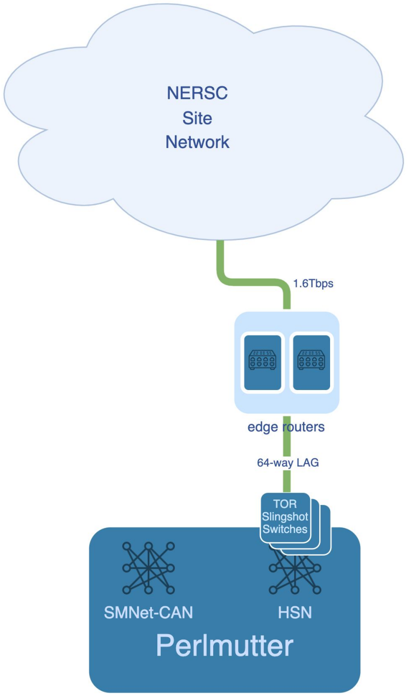
- Health Checks
  - IPVS can be manually configured with `ipvsadm` command
    - Does not have features to track health of real servers
  - Keepalived has integrated management of IPVS configuration
    - Can set up and dynamically maintain IPVS configuration
    - Keep IPVS in sync with current state of topology
      - Can remove/exclude a real server if it is offline or fails health check
      - Each node periodically checks health of all login nodes via HTTP
      - Our health-check scripts are easily extended
      - Check for presence of `/etc/noload` created by admin to exclude a node

Slingshot HSN



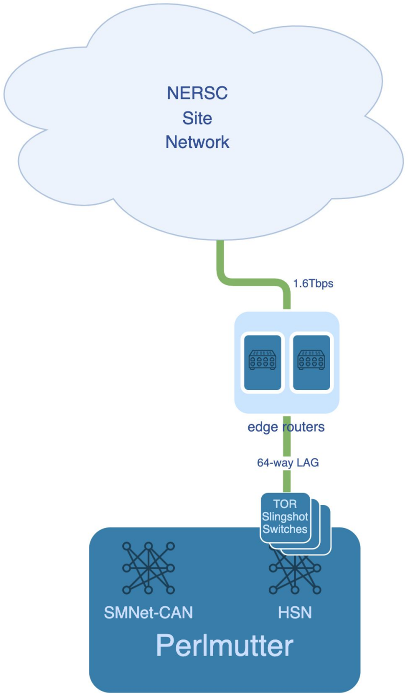
Slingshot HSN

- Ethernet compatible interconnect
  - no protocol translation at system edge
- Early testing of Rosetta compatibility with Arista routers
  - transceiver and cabling validation
  - ensure link comes UP, verify 100G/200G speed
  - initial LAG testing of 2 or 4 links
- 64-way LAG connects Slingshot to Edge-Router pair
  - L_1 and L_2 resiliency to the 16 Slingshot switches in our Service/Login group
  - edge-router configured for MLAG setup with VARP
-

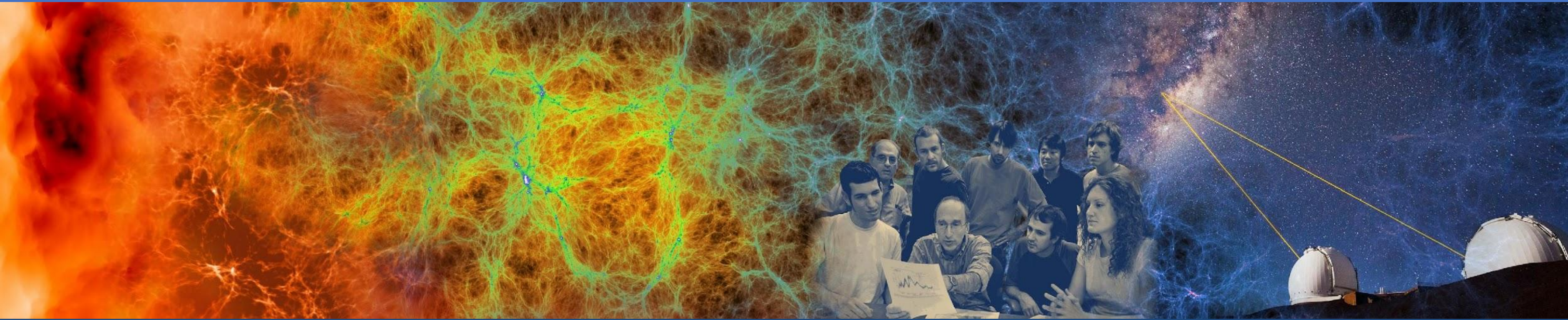


Slingshot HSN

- 4x400G ECMP links to NERSC data center networks
- Every Slingshot Edge host has a publicly routable IPv4/IPv6 address
- NIC and OS settings adjusted for performance ex: Interrupt affinity, read/write socket buffers
- Site DNS and perimeter security configured for SSH logins via the HSN



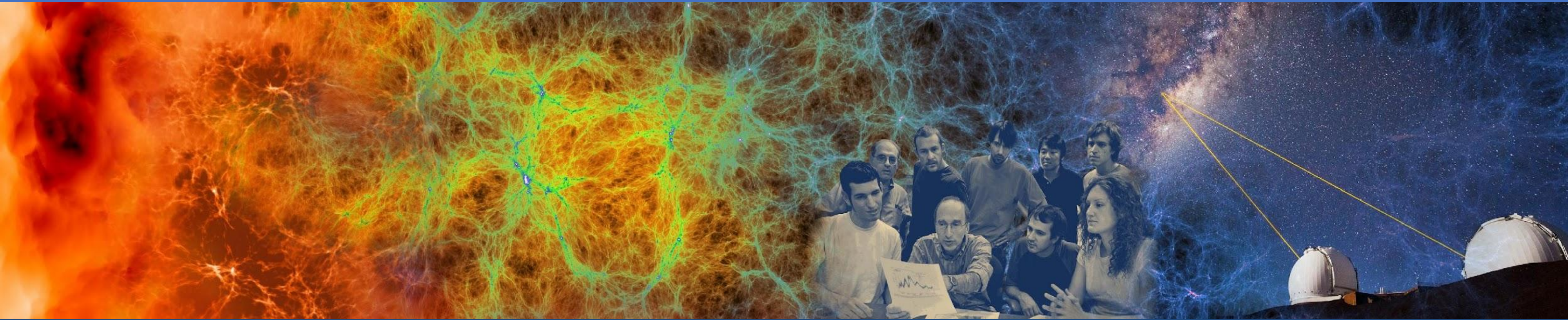
Future



Future Work

- Explore whether GPFS MCOT can help utilize links better
  - Multiple sockets per client server pair
- Research whether queue disciplines can improve latency behavior under load
- Load balance over multiple GPFS server NICs
  - GPFS servers have 4 NICs per server
  - Ethernet bonding
- Tune TCP or (soft)RoCE when storage fabric migrates to Ethernet
- Will have separate 8x400G ECMP link for future storage fabric
- Dynamically enable routes to allow experimental facilities to stream data directly into compute nodes

Thank you



GPFS performance