Hewlett Packard
Enterprise

BOF: CSM updates, iSCSI boot content projection, and other CSM topics

Harold Longley, HPE
Ashalatha A. M., Ravikanth Nalla, HPE
Ravi Bissa, Dennis Walker, Jason Coverston, Siri Vias Khalsa, HPE

May 5, 2025

CSM BOF Agenda

- CSM-based software stack overview 2025
  - Recent changes across entire software stack
- iSCSI-based boot content projection
  - Scalable Boot Content Projection Service (SBPS)
- Rolling Reboots with Zero Downtime in HPC Systems
  - A Novel Approach to System Upgrades Without Service Disruption
- Open discussion on CSM topics

CSM-based Software Stack Overview 2025

Harold Longley, HPE

May 5, 2025

HPC CSM Software Recipes

Software Stack	COS	CPE	CSM	CSM Diags	HFP	SAT	SDU RDA	Slingshot	SHS	SMA	UAN	USS	SLE 15	x86_64	aarch64
24.03	3.0.0	23.12.3	1.5.0	1.5.25	23.12.0	2.6.14	2.3.1	2.1.1	2.1.2	1.9.11	2.7.1	1.0.0	SP5	Yes	Yes
24.08	3.1.0	24.07	1.5.2	1.5.46	24.8.1	2.6.16	3.2.0	2.2.0	11.0.1	1.9.18	N/A	1.1.0	SP5	Yes	Yes
25.01	3.2.0	24.11	1.6.0	1.6.7	24.11.0	N/A	3.3.7	2.3.0	11.1.0	1.10.6	N/A	1.2.0	SP6	Yes	No
25.03	3.3.0	25.03	1.6.1	1.6.14	25.2.1	N/A	3.5.1	2.3.0	12.0.0	1.10.15	N/A	1.3.0	SP6	Yes	Yes

- 24.03 was an Extended Support Release which had its own stream of minor updates
  - CSM 1.5.4 is highest version on this stream
- 24.08 and 25.01 were Continuous Releases
  - 24.08
    - UAN shifted from CSM software to be USS software
    - Workload Managers (PBS Pro and Slurm) are included in the USS documentation
    - Slingshot and SHS are decoupled to become independently installable
  - 25.01
    - Workload Manager (PBS Pro and Slurm) must be downloaded from vendor
      - Slurm is built from source code
    - SAT became released with the CSM software
    - Aarch64 architecture (NVIDIA CPUs) not supported with SLE 15SP6
- 25.03 is an Extended Support Release which will have its own stream of minor updates
  - Aarch64 architecture (NVIDIA CPUs) supported with SLE 15SP6

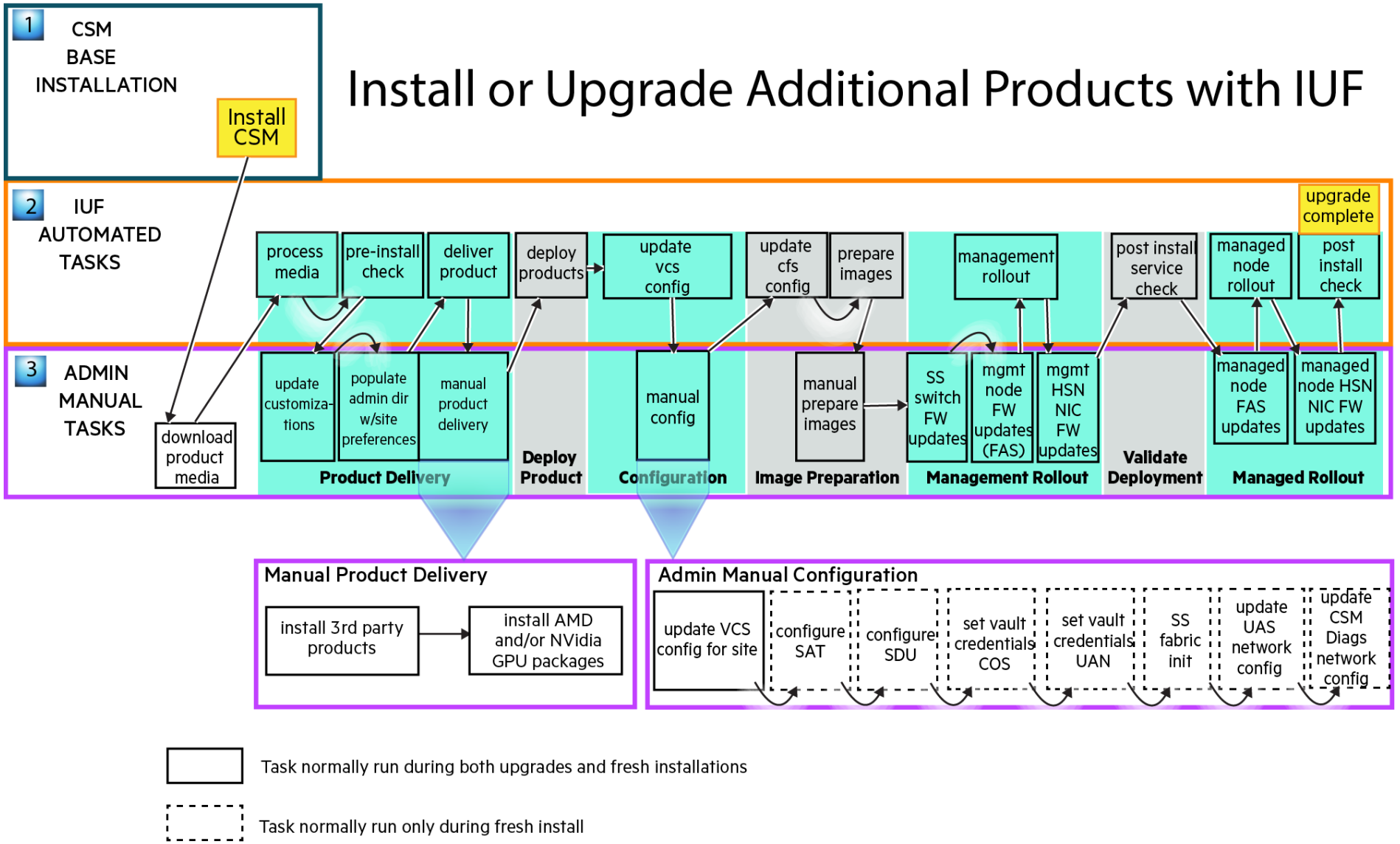
CSM 1.6.1 Improvements 1

- iSCSI-based boot content projection (SBPS) for operating system rootfs and PE images
- Multi-Tenancy improvements
- Bonded HSN interfaces supporting Slingshot resiliency
- VictoriaMetrics used in cray-sysmgmt-health
- Allow customization of ipxe debug options
- Boot Orchestration Service (BOS)
  - Make BOS migration pod more polite
  - Add context managers around BOS requests/sessions; enable paging of BOS components
- Configuration Framework Service (CFS)
  - Update CFS API spec to reject invalid component creation/update requests
  - Bypass needless work in some CFS queries
  - Make CFS Options class thread-safe and more performant
  - Add ability to create CFS source and specify secret name instead of username/password
  - Update CFS API spec with actual status code for successful source restore
  - Improve CFS config delete performance on scale systems

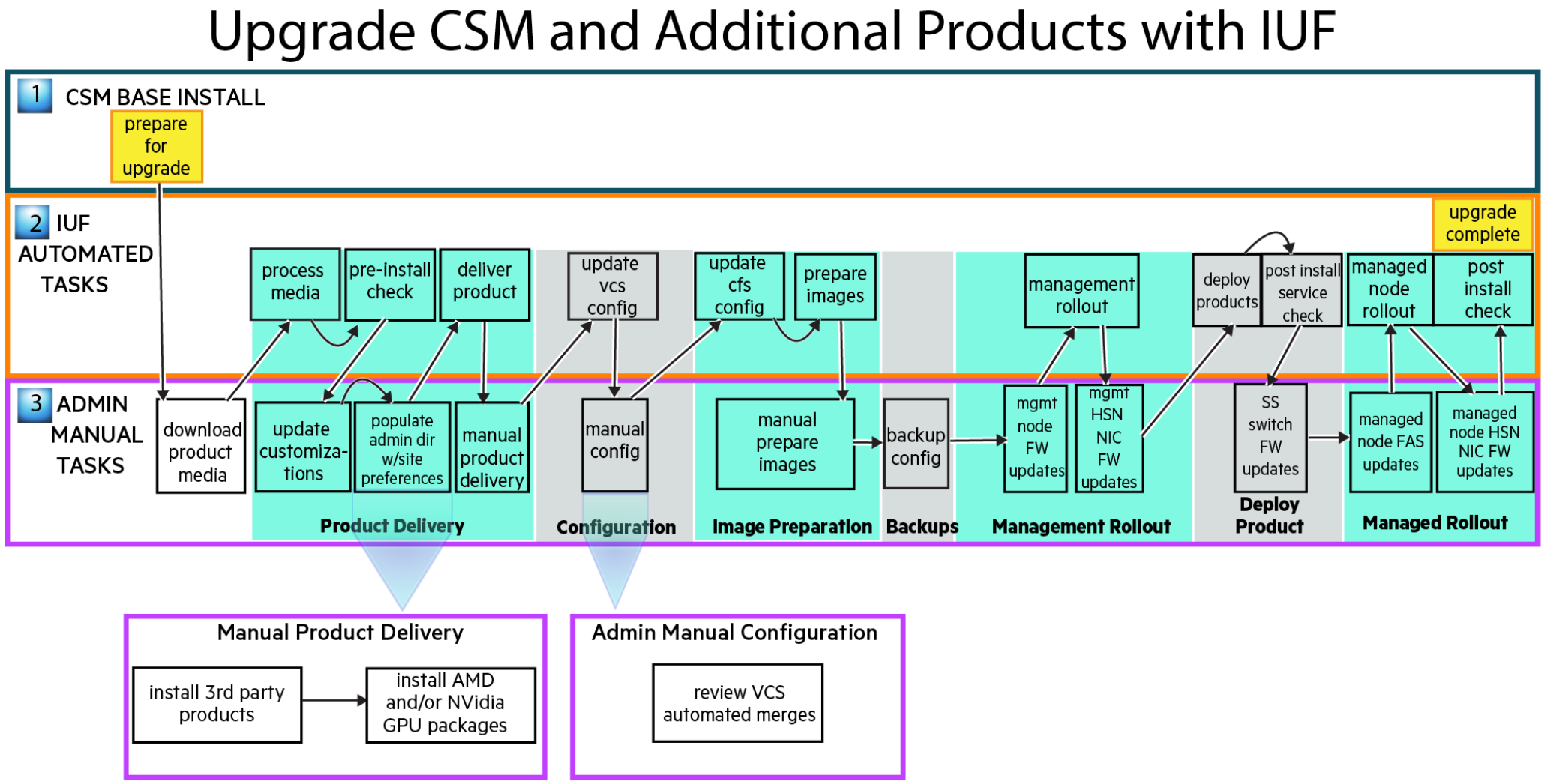
CSM 1.6.1 Improvements 2

- Mitigate resource leaks / heavy usage
  - Power Control Service (PCS), Hardware State Manager (HSM cray-smd), hmcollector, Firmware Action Service (FAS)
- System Admin Toolkit (SAT)
  - Update “sat bootprep” to support CFS v2 or v3
  - Update “sat bootsys” to support CFS v2 or v3
  - Add ability to sort reports by multiple fields
- Upgrade
  - CSM can now be upgraded with IUF
  - Cleanup previous/old SquashFS images during upgrade
  - BOS API now enforces limits that previously had only been recommended
    - When updating to CSM 1.6.x, BOS data is migrated to be in compliance with the API specification
  - Update customization.yaml for System Monitoring Application (SMA) Victoria metrics PVC size
- UAIs no longer run on worker nodes
  - See USS documentation for running UAI containers on UANs

IUF Installation Workflow



IUF Upgrade Workflow



SMA 1.10 Improvements

- Postgres telemetry database (sma-postgres) replaced by VictoriaMetrics
  - VictoriaMetrics increases performance and improves disk efficiency
  - PVC space previously used for Postgres will be reused by VictoriaMetrics, storing much more telemetry data
- Flow replaces PMDB persister and LDMS persister
- Fluentbit and logstash replace rsyslog as log collector and log aggregator
  - Fluentbit better handles logs from Kubernetes and dealing with log file rotation
  - Logstash adds two log pipelines: cray_logs_syslog, cray_logs_containers
- LDMS
  - Includes incompatible config file format: `dcgm` replaced with `dcgm_sampler`
  - Requires administrator to run a workaround script to update ldms config files after the iuf deliver-product stage
- AIops admin updates and new dashboards
  - AIops generates dynamic thresholds to reduce false alarms in monitoring components (e.g. CDU thresholds)
  - Manage AIOPS features with cm aiops cli command
- Alerting
  - View alerts from the command line
    – cm health alertman
  - View alerts with OpenSearch dashboard
    `https://sma-dashboards.cmn.SYSTEM_DOMAIN_NAME`

SMA 1.10 Dashboards and Deprecations

- Grafana dashboards which formerly used Postgres now use VictoriaMetrics
- New Dashboards:
  - Anomaly forecast dashboard
  - Slingshot physical context congestion
  - Slingshot physical context congestion details
  - Slingshot physical context Temperature details
- Removed dashboards
  - Alerta Dashboard
  - Cluster Health Check
  - Prometheus Alerts Overview
- Deprecated features
  - elastalert
  - Telemetry API
  - Alerta
  - cm health alert cli
  - Monasca alerts deprecated and will be removed in CSM 1.7

User Services Software (USS) 1.3 New Features – Admin

- Support for a new iSCSI-based content projection system Scalable Boot Projection Service (SBPS) has been added for use on compute nodes
- Federal Information Processing Standards (FIPS) support
- Dynamic Kernel Module Support (DKMS) for all kernel modules
- Low Noise Mode (LNM) enhancements
  - `detect-detour` captures kernel traces for further analysis when OS noise is detected
- USS rpms moved to equivalent CSM rpms or removed
  - spire-agent, cray-auth-utils, cray-heartbeat, cray-node-identity, cray-orca
- DVS
  - Administrator can shut down a server without disturbing a running workload
  - ActiveMQ Artemis MQTT broker replacing ORCA, cray-heartbeat, HMS HeartBeat Monitor (HBTD), and HMS Node State Change Notifier (HMNFD)
  - DVS has the ability to control how often it rebuilds the node map on each node
    - For DVS clients, the defaults have changed so the node map is only rebuilt on demand
- Version updates
  - SLE 15SP6 x86_64 and aarch64
  - AMD ROCm 6.3 and AMD Driver (amdgpu) 6.3 are the new default versions
  - NVIDIA SDK 24.11 and NVIDIA driver 565.57.01 are the new default versions
  - Cray ClusterStor Lustre Client 2.15.5.x, like Lustre community 2.15.5 LTS release
- To be removed in USS 1.4 from CSM 1.7 worker nodes
  - Content Projection Service (CPS) replaced by Scalable Boot Projection Service (SBPS) in CSM 1.6 provides iSCSI projection of root file system and Cray Programming Environment (CPE) content
  - Data Virtualization Service (DVS)
  - Lustre client software no longer required for file system access in User Access Instances (UAIs)
    - UAIs instead run on User Access Nodes (UANs) where access to Lustre file systems is still available

User Services Software (USS) 1.3 New Features - User

- Apptainer support to create and run applications in containers
- `gpu-nexus-tool`
  - Allows any supported version to be uploaded to default Nexus locations for use in GPU images
  - Supports checking arbitrarily versioned GPU content
- UAIs on UANs
- WLM
  - Slurm Operator support for Slingshot Operator VNI range reservation
  - PALS support for 252 or more processes per Slingshot NIC
  - PALS MPIR debugger launch support for all applications
  - Added an ATOM NUMA node memory imbalance test to check that memory usage is spread evenly among NUMA nodes within the same compute node
  - The PALS default configuration now automatically uses the Spindle package and enables PMIx launch support
  - Added PALS PMIx_Publish, PMIx_Unpublish, and PMIx_Lookup support
  - Added local launch option to PALS to speed up single-node launches
  - Added ATOM GPU reset task to reset GPUs between jobs
  - Added ATOM task to clean up leftover processes
  - Added ATOM task to check CPU clock speed

Slingshot 2.3 New Features

- HPE Slingshot Network Operator
  - Designed to support multi-tenancy in CSM 1.6
- Import HSN IP addresses from an existing fabric template to a new one
- PML Recovery Improvements
- Support for configuring maximum member ports on per LAG ID level
- Slingshot Switch certificate expiration notification
- Network Transceiver + Passive Optical Cable Support
- Enhancements to scale Fabric Agent
- Metrics and alerts can stream to multiple collectors
- Support for Passive Flow Control (PFC) on non-Cassini Edge ports
- Enable programmable switch/port conditions
- Add diff-fabric-template-files command
- Slingshot fabric manager RBAC support
- Fat Tree topology support – Phase 1
  - 2-stage Fat Tree with leaf and spine switches
- Orchestrated Maintenance - Phase
  - Policy-drive firmware management to manage the switch firmware via switch policy
  - Allowlist/blocklist of firmware revisions
  - Set desired firmware revision
  - Tools for identifying non-compliant switches
  - Mechanisms for safely allowing desired firmware to be forced into compliance with the switch policy
  - Switch policies have new flags that enable more convenient control of online/offline status for multiple fabric and edge ports
- Distributed/Autonomous routing – Phase 1 (alpha release so off by default)
  - Limitations: fat tree topologies not supported, orchestrated maintenance not supported, adding edge ports not supported
  - Reduces latency from tens of seconds to under a second by placing the detection of the event and response to it at the switch
  - Workloads will run with less disruption caused by link state changes

SHS 12.0 New features

- CXI Driver has been updated to allow AMD GPUs to use dmabuf
  - The CXI provider can be utilized when AMD GPU memory registration is enabled with DMA-BUF in Libfabric
- KDREG2 is installed by default
  - Set the default memory monitor to KDREG2 by configuring the Libfabric environment variable
- DKMS updates done in parallel
- Improve MR close `fi_mr_close()` performance when FI_CXI_MR_MATCH_EVENTS=1
- On platforms with 64 KiB page sizes, the Ethernet driver was not efficiently using memory to sync incoming packets
  - This may present itself as performance degradation with running Ethernet applications or TCP/ UDP benchmarks like iperf
  - The Ethernet driver has been updated to efficiently use pages when the page size is greater than 4 KiB
- libfabric CXI provider has been updated to allow the aws-ofi-nccl and awsofi-rccl plugins to use the 1.18 version of the libfabric interface

iSCSI-based Boot Content Projection

Ashalatha A. M., Ravikanth Nalla, HPE

May 5, 2025

iSCSI-based Boot Content Projection Agenda

- Introduction
- Challenges with DVS
- iSCSI SBPS features
- iSCSI SBPS solution
  - Architecture overview
  - Solution workflow
- iSCSI SBPS Metrics
- References

Introduction

- What is Boot Content Projection & Why?
- What do we project and for which nodes ?
  - **Rootfs** and **PE** images for diskless nodes like **Compute** or **UAN** nodes
- Current Solution: **DVS** (Cray Data Virtualization Service)
- **NERSC & LANL** asked for alternatives to DVS due to the challenges faced
  - In Dec 2022 NERSC approached HPE (COS, CSM) as they lost availability and faced operational complexity
  - Asked for alternative solution which is easier & faster to **deploy**, easier to **manage**, more **reliable, available** & **secure**
- **Alternatives to DVS**
  - **NFS**: Tried by both NERSC & LANL
    - Not a robust network load balancing model, requires intermediate storage location, concern with muti-tenancy
  - **Ceph RBD:** Ceph Remote Block Device tried by LANL.
    - Ceph cluster nodes do not have HSN adapter, require replication of s3 content onto RBD devices.
  - **NVMe-oF:** Reviewed but not Pursued
    - iSCSI was developed in 1998, extensively developed and standardized as a network block device projection protocol
    - NVMe-oF developed in 2011, seen some standardization in late 2022 & 2023, borrowing heavily from iSCSI

Challenges with DVS

- Reliability
- Availability
- Security
- Ease & speed of deployment
- Ease of management

Challenges with DVS RAS

- **Reliability and Availability**
  - **Integrated with CSM and uses Lnet** (Lustre Networking) which supports low level network drivers (LNDs), requires to install/host multiple kernel modules
  - Largely **implemented** via **Linux kernel modules**, **susceptible** to any operational issues
  - **Kernel integration and delicate network fabric** had **kernel panics**, causing loss of availability
  - Outages had challenges in recovery
  - Troubleshooting was very hard
- **Security**
  - Not designed, as deployed in CSM
  - Multiple **vulnerabilities** since CSM inception
  - Cross-product-stream dependencies lead to lagging kernel versions and security vulnerabilities

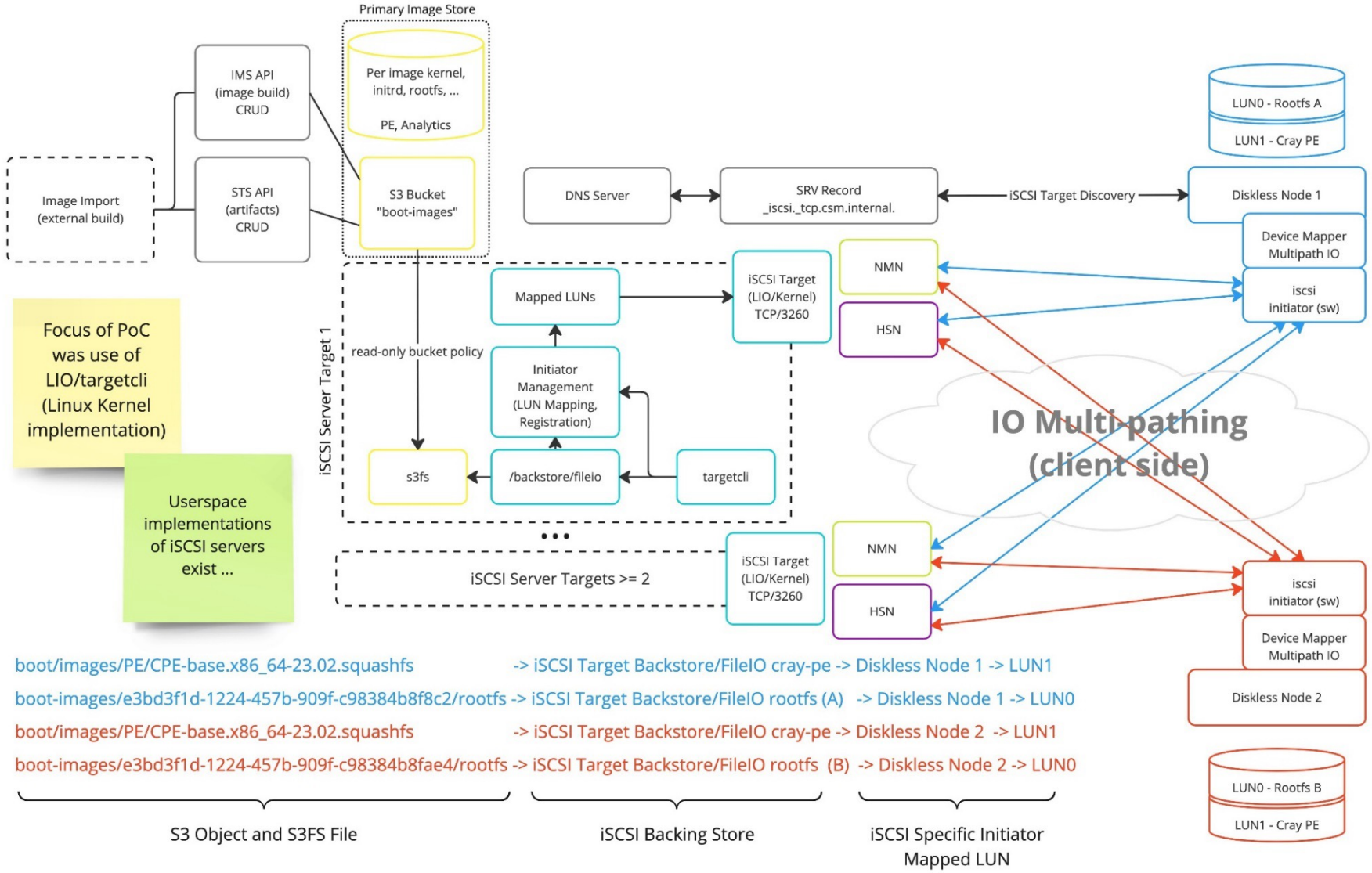
Challenges with DVS Ease

- **Ease & speed of deployment**
  - COS/USS is deployed as product stream on top of CSM
    - Using node personalization COS product stream installer installs DVS and Lustre kernel modules
  - Requires a stable Slingshot fabric which may not available during early boot
  - Hard to resolve package dependencies
  - Introduces host networking tuning parameters, often incompatible with CSM
  - Debug and triage issues leading to delayed installs and loss of availability for customers
  - Multiple teams (CSM, COS/USS, Slingshot) have to ensure there is kernel level compatibility across all OS versions
- **Ease of management:**
  - Usability is a challenge due to tight coupling across CSM, COS/USS, Slingshot
  - Speed of update delivery is a concern
  - Adding or removing DVS servers and DVS clients is a concern

iSCSI SBPS features

- **SBPS:** Scalable Boot Content Projection Service
  - Open-source friendly solution for read-only Squashfs content projection (rootfs and PE)
  - Horizontally scalable content projection services (CPS)
  - Delivers active/ active IO operation and seamless failover/ failback for clients
  - Supports projection over both HSN and NMN networks
  - Does not require additional hardware infrastructure
  - Coexists with DVS until it is deprecated (one projection at a time active) in CSM 1.7.0
  - Enables future work related to image access control, multi-tenancy etc.,
  - Does not require duplication of Squashfs content from S3
  - Supports monitoring of CPS services for performance and reliability engineering
  - Aligns with future plans for similar functionality in next generation systems management solutions

iSCSI SBPS Architectural Overview



Figure#1: iSCSI SBPS implementation using per-initiator LUN Mapping Architecture

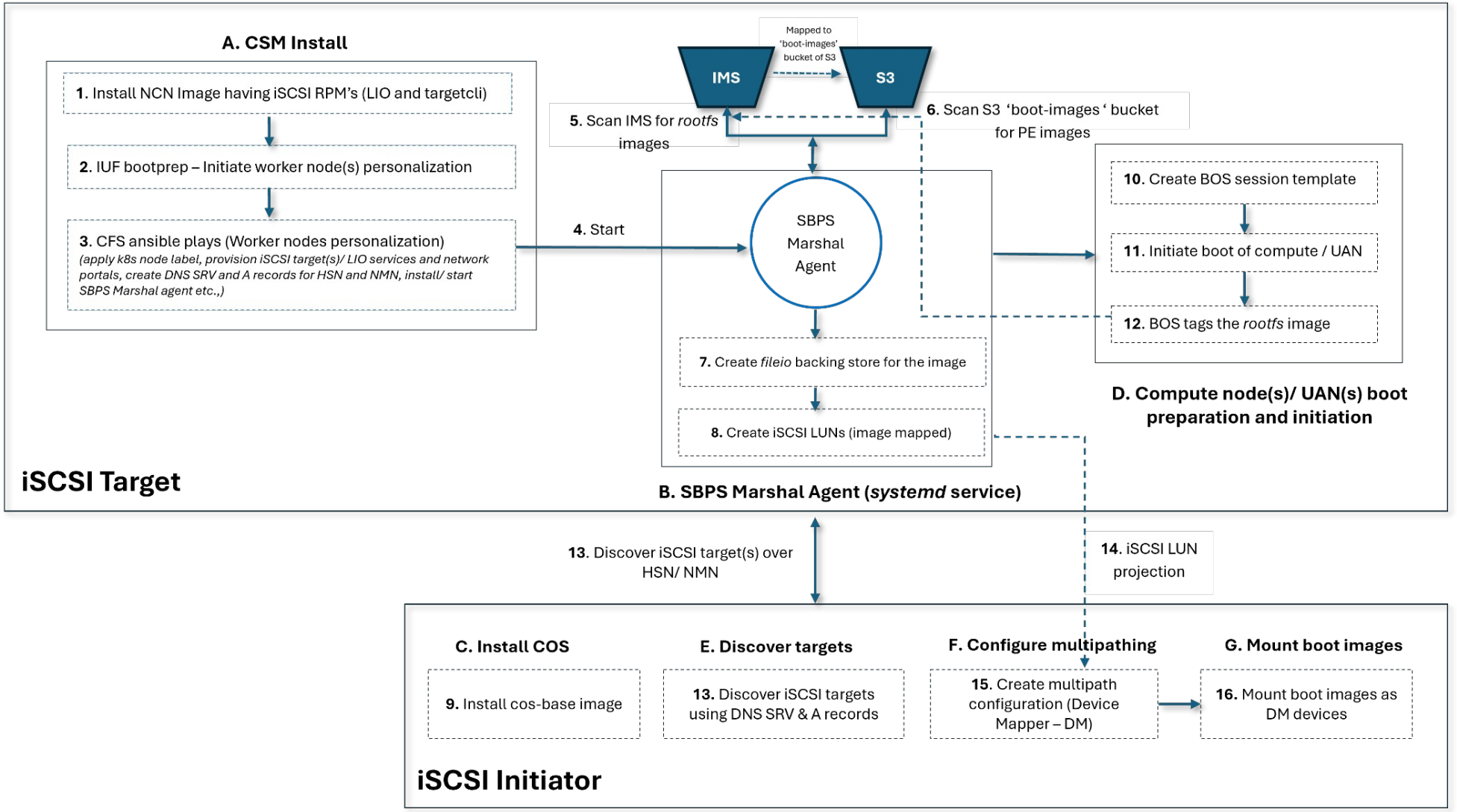
iSCSI SBPS Solution Worker Node

- iSCSI Target (Server) Implementation: (Worker Node)
  - **iSCSI Target Software:**
    - Standard Linux kernel + user space iSCSI target services (LIO: software-based iSCSI target implementation) + targetcli.
  - **SBPS Marshal Agent**
    - Main named as SBPS Marshal Agent to manage lifecycle of projected content (rootfs and PE)
      - Scan S3 anserviced IMS for rootfs and PE images (Uses Spire and OPA for authentication)
      - Uses *targetcli* to create fileio backing store and iSCSI LUNs
      - Runs as a *systemd* service and scans every 180 seconds
  - **SBPS Provisioning**
    - Fileio backstore
      - basis for LUN projection to iSCSI initiators
    - CFS (Configuration Framework Service) Ansible plays used for worker node personalization
      - Apply Kubernetes node label
      - Provision iSCSI target(s)/ LIO services and network portals
      - Create DNS “SRV” and “A” records (for HSN and NMN) for iSCSI target discovery from iSCSI initiator
      - Install and configure SBPS Marshal Agent

iSCSI SBPS Solution Client Node

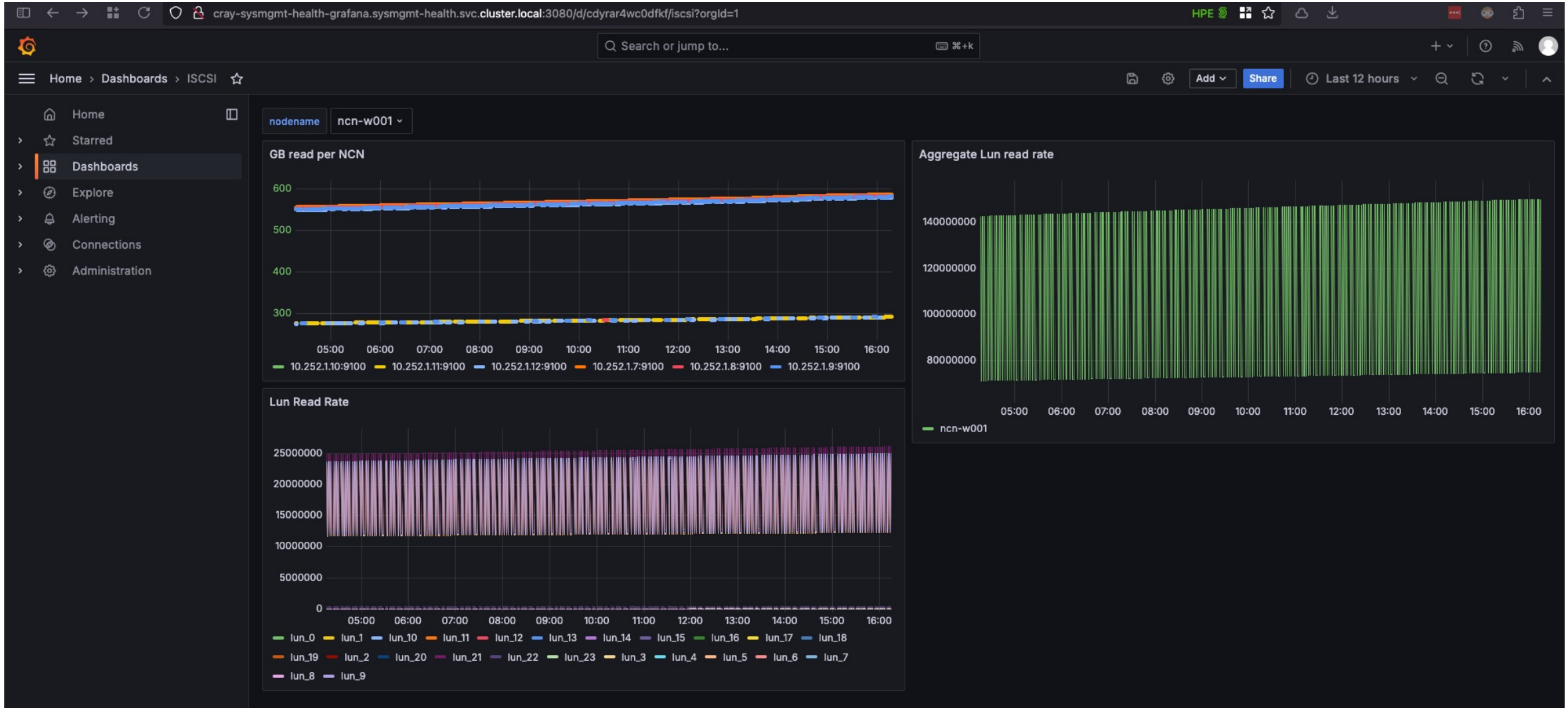
- iSCSI Initiator (Client) Implementation: (Compute/UAN Node)
  - **iSCSI Initiator Software**
    - Standard Linux kernel and userspace iSCSI initiator services part of the SLES distribution and multi-pathing software
  - **iSCSI target discovery**
    - Discovery from iSCSI initiator is done using DNS “SRV” and “A” records (which is a boot parameter to BOS session template)
  - **Use Linux Device Mapper (DM) Multipath IO:**
    - For seamless failover/failback for high availability, and for active/active IO load balancing
  - **BOS (Boot Orchestration Service) Enablement**
    - Create a boot session template with new SBPS boot parameters to boot compute nodes
    - New *rootfs_provider*, sbps and new *rootfs_provider_passthrough* schema added for SBPS:
      - `sbps:<schema version>:<IQN Domain>:<DNS SRV record reference>:<client discovery timeout in seconds>`
      - **Example for NMN:**
        - `sbps:v1:iqn.2023-06.csm.iscsi:_sbps-nmn._tcp.local:300`
      - **Example for HSN:**
        - `sbps:v1:iqn.2023-06.csm.iscsi:_sbps-hsn._tcp.local:300`

iSCSI SBPS Solution Workflow



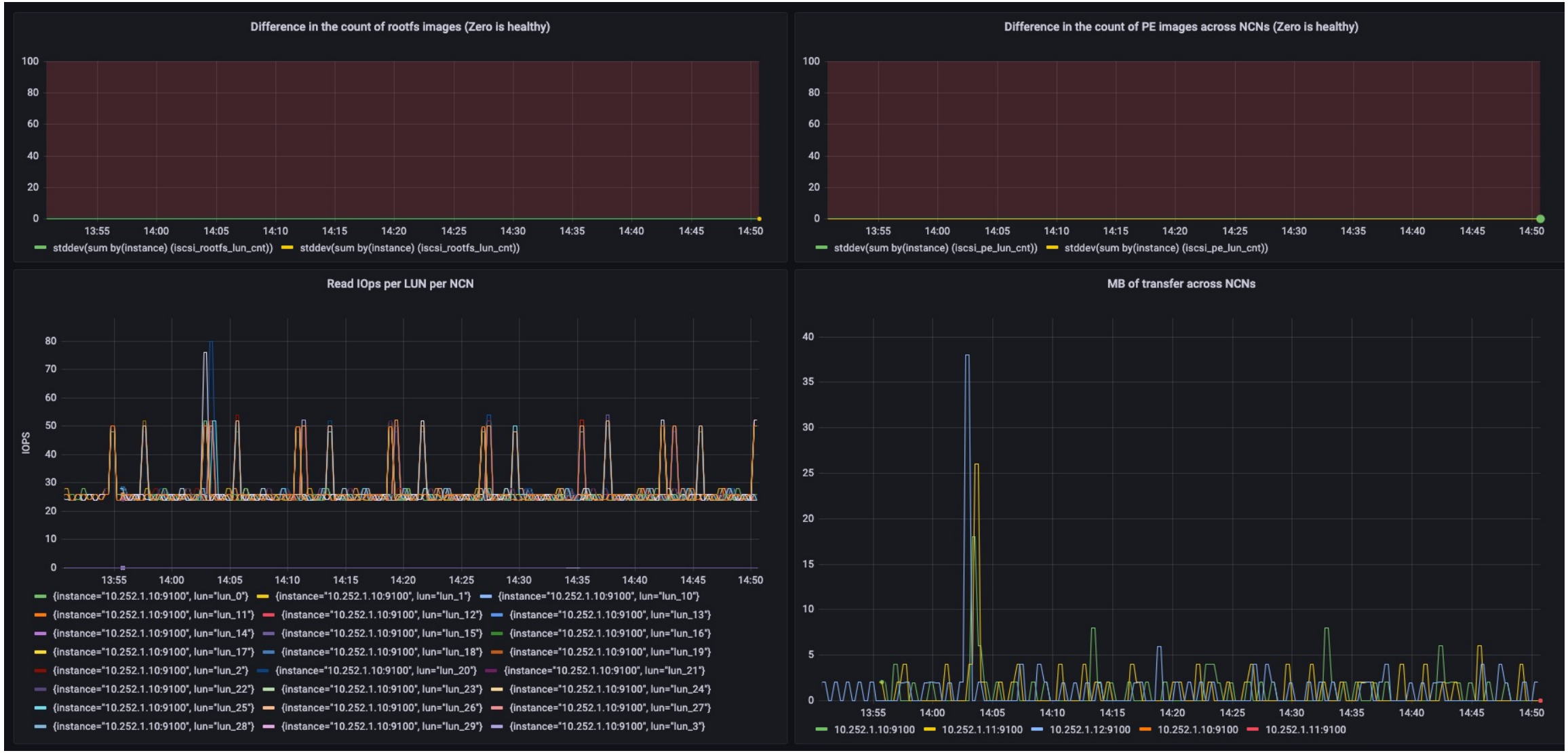
Figure#2: iSCSI SBPS Solution flow diagram

iSCSI SBPS Metrics



Figure#3: Read IO metrics: snapshot-1

iSCSI SBPS Metrics Contd.,



Figure#4: Read IO metrics: snapshot-2

References

- https://github.com/Cray-HPE/docs-csm/blob/release/1.6/operations/iscsi_sbps/iscsi_sbps.md
- https://github.com/Cray-HPE/docs-csm/tree/release/1.6/operations/configuration_management
- What is Amazon S3? - Amazon Simple Storage Service
  - https://docs.aws.amazon.com/AmazonS3/latest/userguide/Welcome.html
- What is iSCSI and How Does it Work? (techtarget.com)
  - https://www.techtarget.com/searchstorage/definition/iSCSI#:~:text=ISCSI%2C%20which%20stands%20for%20Internet,(WANs)%20or%20the%20internet.

Rolling Reboots with Zero Downtime in HPC Systems

A Novel Approach to System Upgrades Without Service Disruption

Ravi Bissa, Dennis Walker, Jason Coverston, Siri Vias Khalsa, HPE

May 5, 2025

From Downtime to Zero Impact: Our Reboot Strategy

Challenge:

- Upgrading OS/firmware sacrifices Maximum Compute Time (MCT = 24hrs × 365days × N)
- Every offline node directly impacts revenue—even single-node downtime is a business problem.
- Traditional OS and firmware upgrades typically require scheduled maintenance windows, during which nodes are drained or taken offline. This results in:
  - Interrupted jobs
  - Decreased user productivity
  - Bottlenecks in job queueing
  - Revenue impact in costed systems

Industry Pain Points:

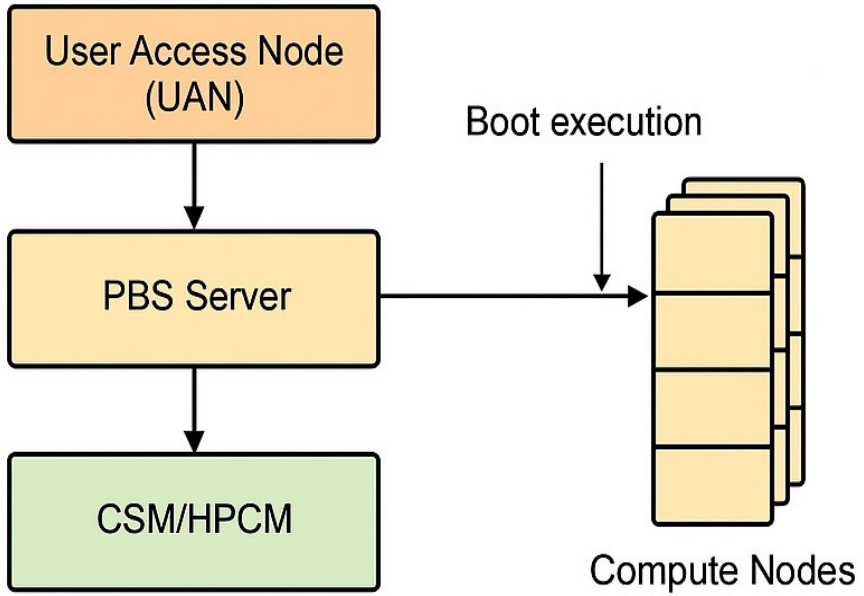
- Downtime stretches from hours to days
- Manual orchestration leads to errors
- Job preemption or kill signals disrupt scientific workflows

HPE's Solution:

- We introduce a **zero-downtime, rolling reboot model** that delivers OS/firmware updates with no impact to active jobs.
  - No job preemption or kill
  - Continuous availability
  - Node-by-node orchestration using PBS, BOS, and monitoring hooks

System Architecture

- This reboot orchestration framework builds upon existing HPC infrastructure by integrating and extending core components for automated, resilient operations:
  - **User Access Node (UAN):** Acts as the control entry point, hosting orchestration scripts and managing configuration workflows.
  - **PBS Server:** Manages job scheduling, queue-based reboot logic, and executes hooks to track node readiness and control concurrency.
  - **Compute Nodes:** Are rebooted independently in a non-disruptive manner, ensuring that active workloads on other nodes remain unaffected.
  - **CSM/HPCM Stack:** Provides the foundational provisioning and monitoring services — including node power control and health validation — through interfaces like Redfish or IPMI.



Rolling Reboot Workflow Breakdown

Node Selection Criteria:

- PBS continuously monitors node states.
- Reboot candidates are selected based on **real-time availability** (i.e., node must be idle and not running jobs).
- Nodes must also pass **pre-reboot checks** (e.g., health, job status, dependency readiness).

Job Creation Pipeline:

- Once a node qualifies:
  - A **PBS job** is submitted to the NODETRACKERQ queue to record and track the reboot candidate.
  - This is immediately followed by a job in the ROLLINGREBOOTQ, which handles the actual reboot orchestration.

Execution Layer:

- **Boot Orchestration Services (BOS)** is triggered to perform the reboot.
- Custom scripts run:
  - To initiate the reboot via BOS.
  - To **validate post-reboot success criteria** (e.g., correct OS version, network reachability, service health).
  - To **log state transitions** (pre, in-progress, success, failure) for observability and audit purposes.

PBS-Driven Reboot Orchestration Logic

- The entire reboot process is orchestrated using **PBS queue-based job management**.

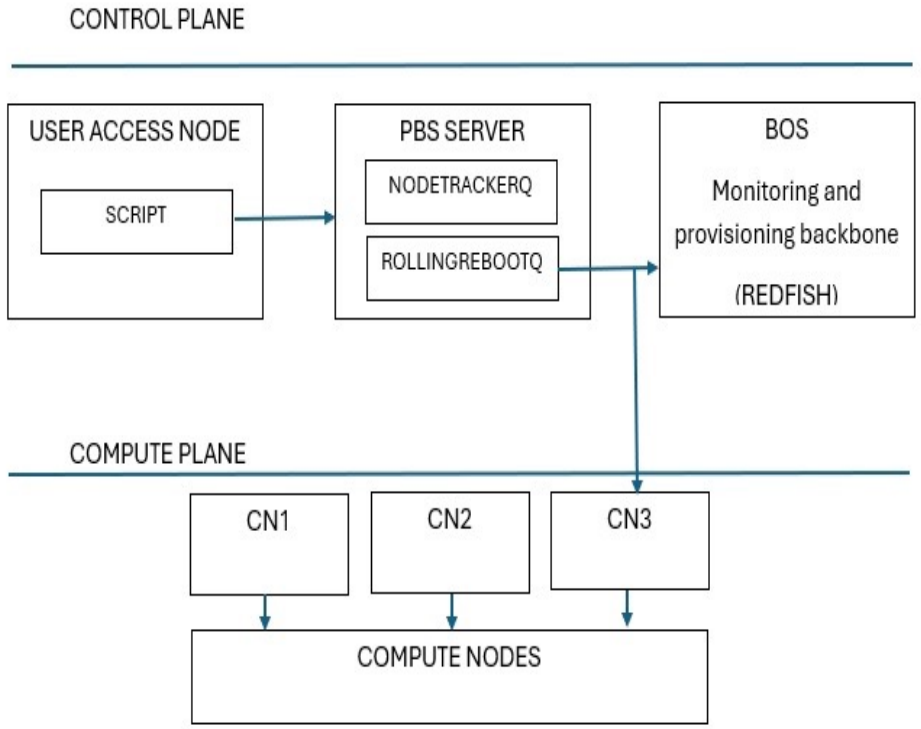
- **NODETRACKERQ**:
  - Acts as a **meta-queue** that monitors node readiness.
  - Only schedules jobs for nodes that are:
    - Idle
    - Not running active jobs
    - Safe for reboot (health checks passed)

- **ROLLINGREBOOTQ**:
  - Handles the **actual reboot orchestration jobs**.
  - Jobs are launched only after the corresponding node passes all readiness checks.

- **Decoupled, Per-Node Execution**:
  - Each node operates independently, with its **own PBS job**.
  - There is **no inter-node dependency**, ensuring:
    - A reboot failure on one node **does not block or delay** others.
    - Robustness and fault isolation.

- **Fine-Grained Control & Concurrency Management**:
  - Queue-level PBS controls such as:
    - max_run – limits parallel job execution (reboot throttling).
    - enabled=False – used to dynamically pause queue dispatching.

- Facilitates safe scaling, auditing, and rollback handling.



Self-Regulating Throttling Mechanism

•Designed to **prevent overload** and avoid **job starvation** during rolling reboots.

•Implements a **dynamic, self-adjusting control loop** using:
- **PBS periodic hooks**
- **CSM (Cray System Management) state monitoring**

•The PBS hook periodically checks:
- The number of nodes currently in a **down or rebooting state**
- The **available system capacity** (i.e., how many nodes are idle or schedulable)
- The PBS queue's configuration, particularly max_run, which limits concurrent jobs

- Based on these checks, the hook **dynamically adjusts execution thresholds**:
  - Throttles or pauses reboots if system load is high
  - Resumes them when safe

- Additional safeguards:
  - **max_concurrent_provision** in BOS ensures only a safe number of nodes are rebooted in parallel.
  - The reboot queue is **disabled by default** and only **re-enabled** once conditions are met.

This approach is **adaptive and autonomous**, ensuring:
- **High availability is maintained**
- Reboots progress safely even during large-scale operations

Key Features

This approach is designed to be resilient, self-correcting, and scalable across multi-thousand-node HPC systems.

- **Resiliency & Reliability:**
  - **Checkpointing with job tracking:** Each node's reboot progress is tracked through PBS jobs, which act as checkpoints in the reboot pipeline.
  - **Self-healing validation logic:** If a node fails to come up cleanly (e.g., hangs or reboots into a failed state), the validation step detects it and either retries or flags it for manual inspection.
  - **Controlled reboot via BOS:** We use Cray's Boot Orchestration Service (BOS) to programmatically reboot nodes using validated boot parameters.
  - **Rollback via STARTING_IMG:** By setting an environment variable, failed reboots can fall back to a known-good OS image, reducing admin overhead during staged rollouts.

- **Availability & Scalability:**
  - **Throttling:** Built-in hooks prevent mass reboots that would degrade service, instead pacing operations using real-time node state awareness.
  - **Script-based automation:** The reboot and validation logic is encapsulated in reusable scripts—portable, testable, and version-controlled.
  - **Parallelism-aware:** The architecture supports multiple reboots in parallel—but only within safety constraints. This ensures faster rollout without risking job preemption or system instability.

Implementation Results

We deployed this solution on a **multi-zoned, CSM-managed HPE Cray EX system**, targeting an OS update across hundreds of nodes.

- **Performance Observations:**
  - **Zero Downtime:** All user-facing jobs continued to run throughout the operation. PBS showed no job preemptions or restarts.
  - **No Scheduling Degradation:** Job throughput remained stable. Nodes were marked offline only during their scheduled reboot phase.
  - **Minimal Impact:** Service-level metrics (e.g., network latency, I/O wait, PBS scheduler performance) showed negligible degradation.

- **Before vs After:**
  - **Before:** Reboots required scheduled maintenance windows, manual coordination between admins and schedulers, and full system drains.
  - **After:** Nodes reboot autonomously under script control; no system-wide scheduling holds or downtime windows.

- **Technical Learnings:**
  - Queue-based orchestration was key to system stability.
  - Throttling parameters needed fine-tuning during scale testing.
  - Post-boot validation was crucial for confidence.

Conclusion & Future Work

- **Key Achievements:**
  - Successfully enabled OS upgrades in production systems **with zero impact on job execution**.
  - Demonstrated a **scalable, decentralized approach** to node-level upgrades that integrates with existing scheduler and provisioning systems.
  - Validated architecture across zones.

- **System Impact:**
  - **Improved availability:** No downtime, no drain events.
  - **Enhanced user experience:** Transparent to scientists and engineers running jobs.
  - **Simplified operations:** Scripts are fully auditable, version-controlled, and reproducible.

- **Next Steps:**
  - Expand this reboot model to firmware upgrades beyond OS (e.g., UEFI, NIC, BMC).
  - Add Chaos Monkey-style failure injection to validate rollback robustness.

Thank you

harold.longley@hpe.com, ashalatha-a.a@hpe.com, ravikanth.nalla@hpe.com, ravi.bissa@hpe.com, jason.coverston@hpe.com, dennis.walker@hpe.com, sirivias.khalsa@hpe.com