Hewlett Packard
Enterprise

Monitoring, Tuning, and Troubleshooting a CSM system

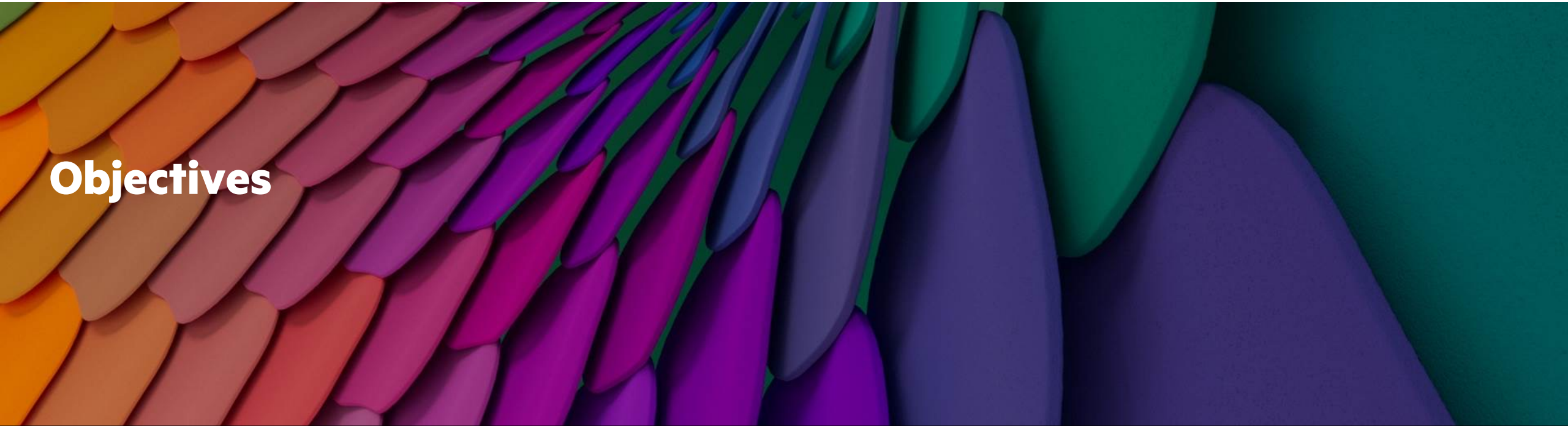
Harold Longley, HPE
Jason Sollom, HPE

May 5, 2024

Agenda

- Objectives
- CSM Architecture Overview
- Monitoring
- Tuning
- Troubleshooting
- Resources
- Q & A

Objectives

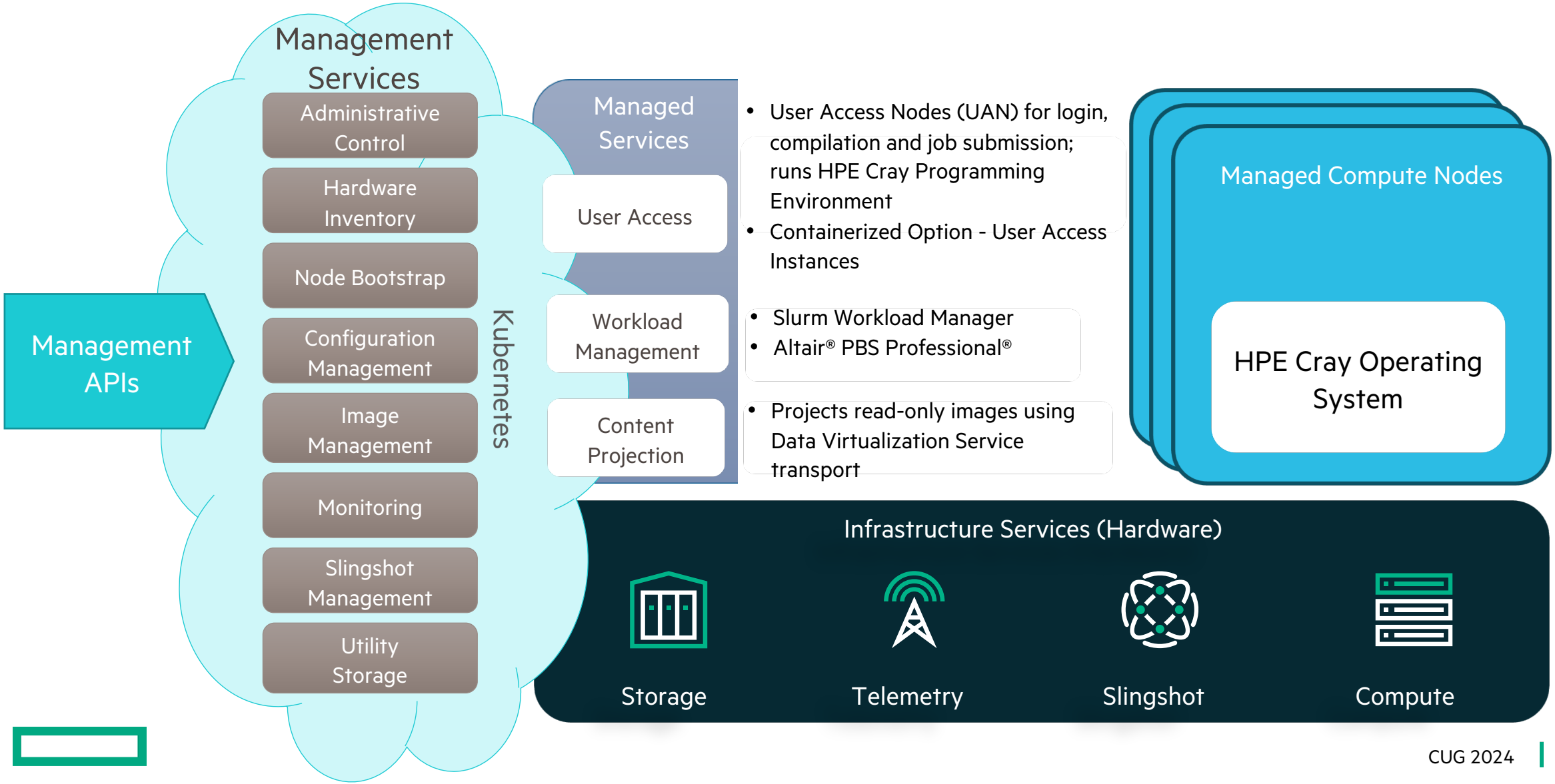


Objectives

- After completing this course, you should be able to:
  - Learn what is in the toolbox for monitoring, logging, and tuning
  - Consider how to apply tools to understand system operation
  - Learn possible approaches to improve system performance
  - Explore troubleshooting methodologies using tools

CSM Architecture Overview

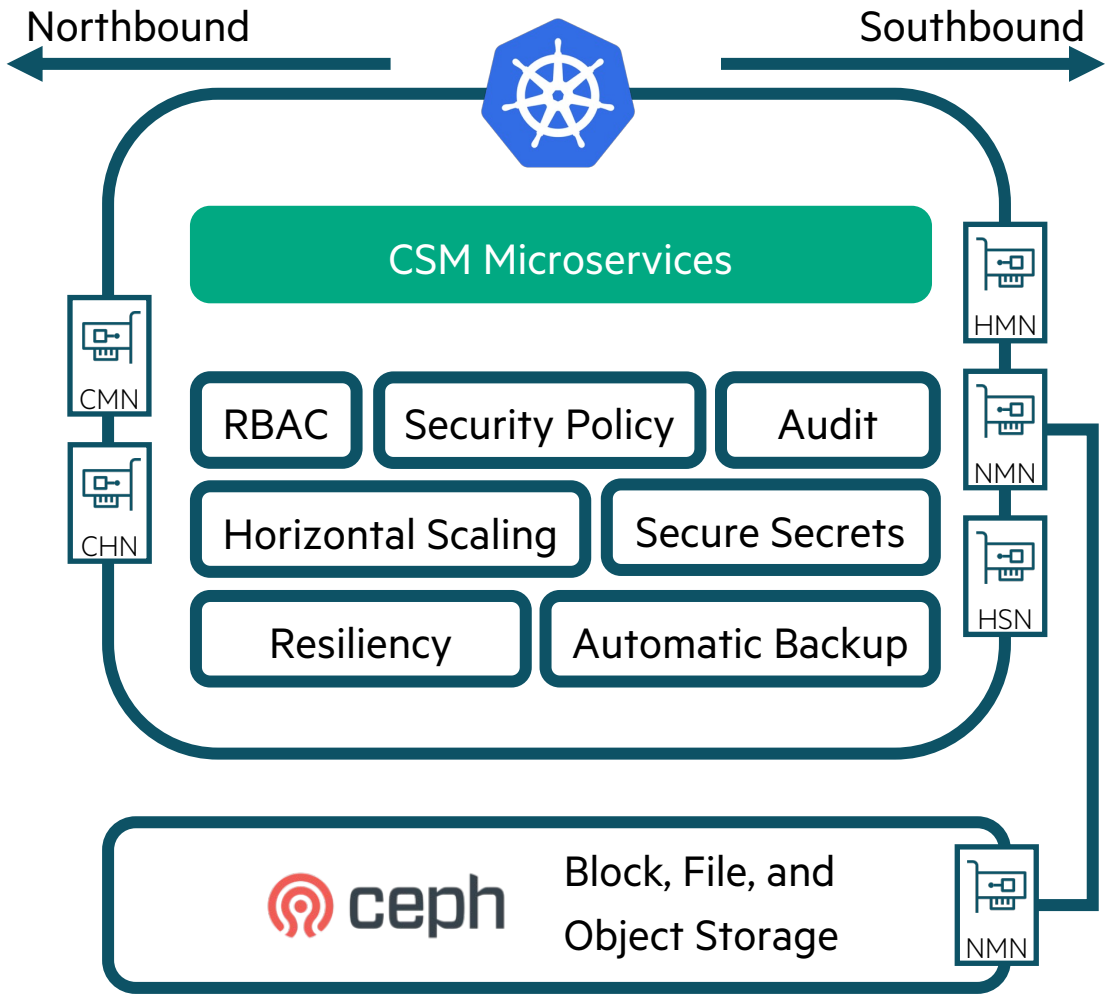
HPE Cray System Management solution overview



System Management Platform

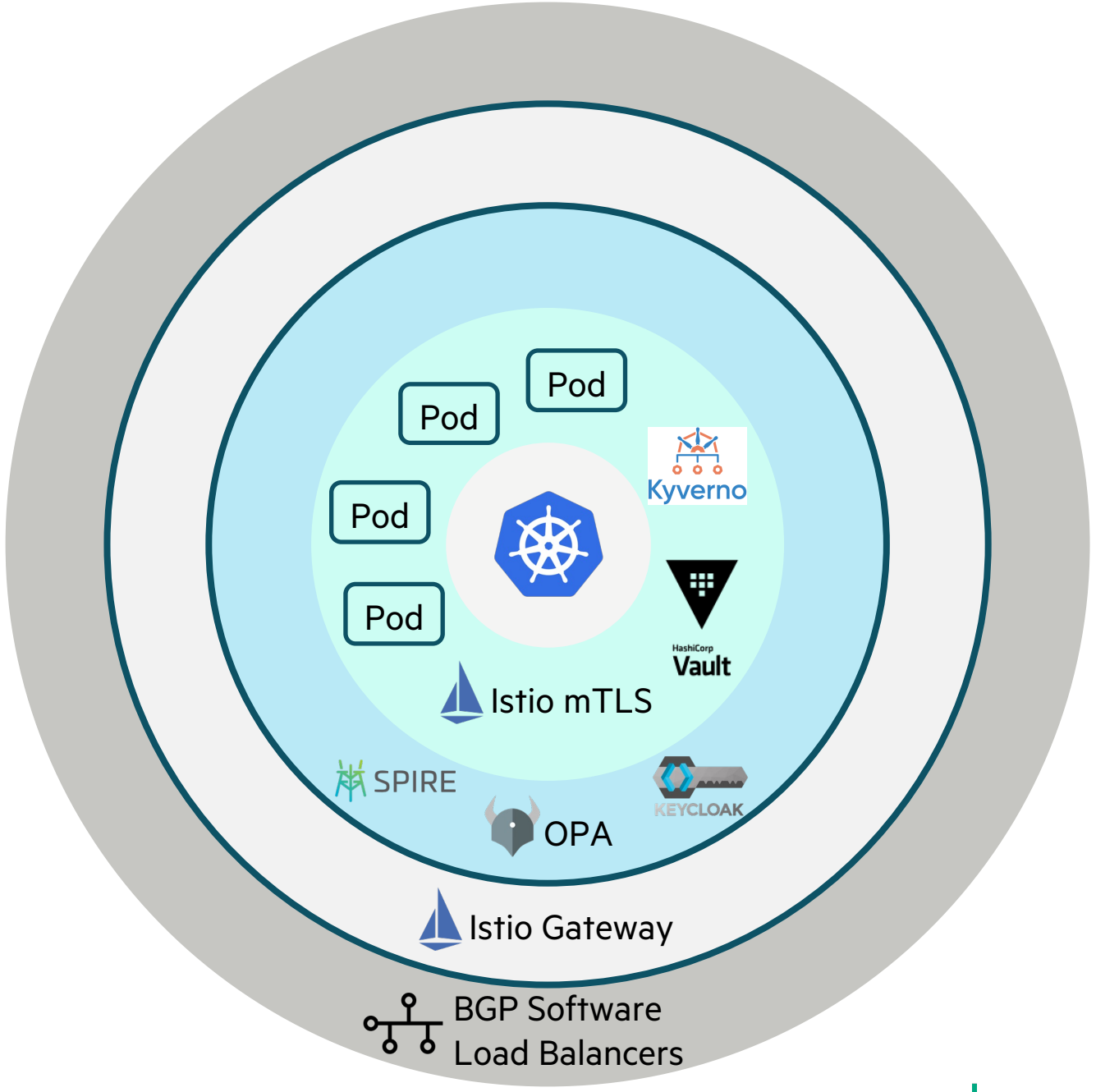
- Kubernete (K8S) as Platform-Building-Platform
- Kubernetes, Istio, and K8S Operators for infrastructure
- Layered microservices for managing HPC clusters
- HPC-enablement only in the upper layers
- Northbound gateways for *system-external* users and admins
- Southbound gateways for *system-internal* NPEs
- MIT licensed
  - GitHub organization: https://github.com/Cray-HPE

All northbound interactions protected by TLS 1.3 and OIDC authentication

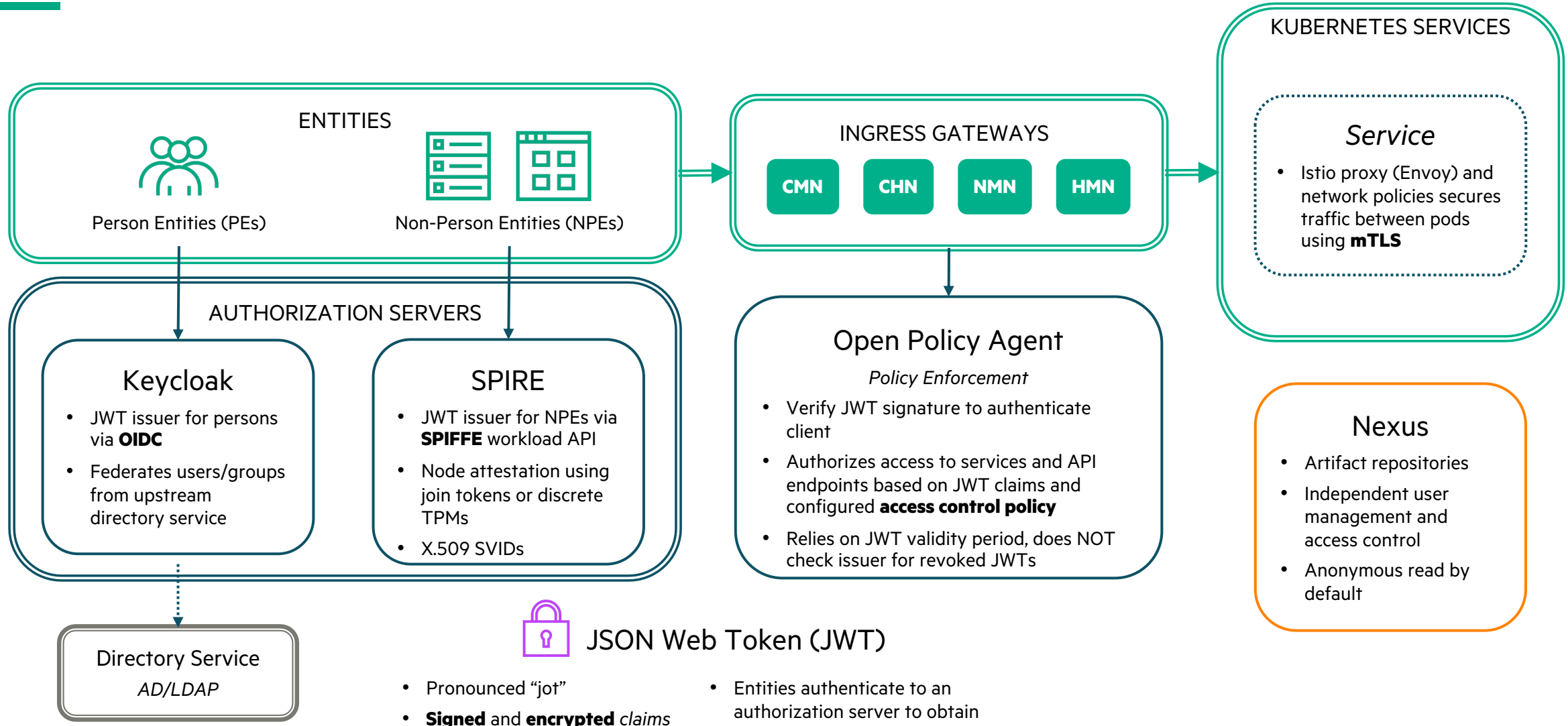


Microservice Security Layers

- Pod to Pod traffic is secured by Istio with mTLS and Kubernetes NetworkPolicy resources
- Kubernetes workload security audited and enforced by Kyverno
- Secrets managed by Hashicorp Vault (OSS, bank-vaults)
- Open Policy Agent (OPA) authenticates clients and make ingress authorization decisions based on policy
- Istio provides gateways to facilitate ingress/egress and enforce OPA authz decisions
- MetalLB allocates Virtual IP addresses that pass traffic to Istio Gateways
- Keycloak and Spire provide OIDC tokens for authentication
- Keycloak federates with upstream LDAP or Kerberos for user directories



IDENTITY AND ACCESS MANAGEMENT



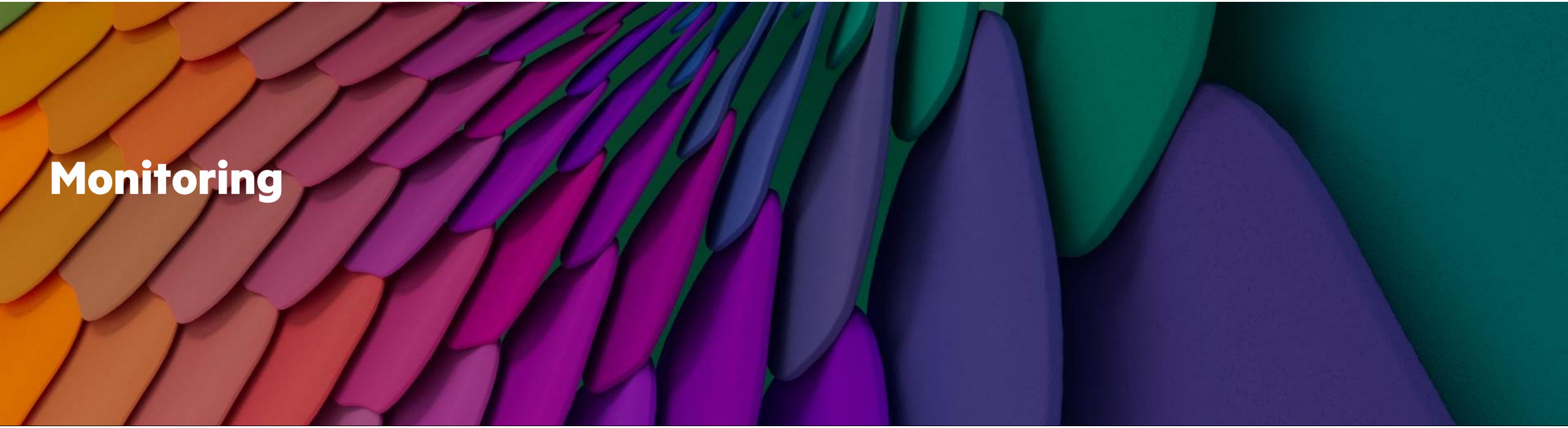
JSON Web Token (JWT)

- Pronounced “jot”
- **Signed** and **encrypted** *claims* about an entity
- Entities authenticate to an authorization server to obtain JWT
- Entities use JWT as bearer token

Kubernetes Runtime Policy Engine
Enabling Ecosystem

- Kubernetes supports dynamic admission control for API requests (e.g., submit a pod for execution)
- Admission control can either *validate* or *mutate* an API request
  - Example Validation Use Cases
    - Verify that the Kubernetes manifest (e.g., pod spec) meets certain security or resource requirements
    - Verify that a container image has provable provenance (e.g., via Sigstore Toolchain) before allowing execution
  - Example Mutation Use Cases
    - Force Kubernetes manifests (via config patching) to adhere to certain security or resource requirements
      - e.g., force execution as non-root, disable privilege escalation or limit to a set of Linux capabilities
    - Translate resource configurations submitted in an older schema version to a newer one
  - Kyverno is a policy engine defined for Kubernetes, that leverages dynamic admission control
    - Comes with a large reference policy library, including ones that align with the Kubernetes Pod Security Standards and guidance such as NIST SP 800-190.

Monitoring

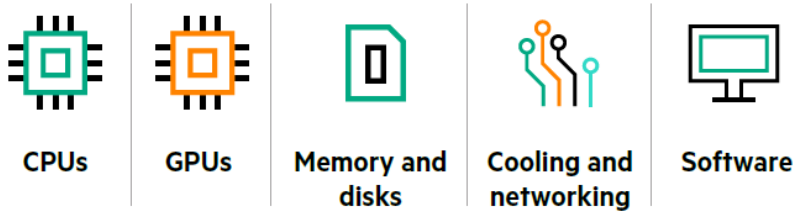


Monitoring Agenda

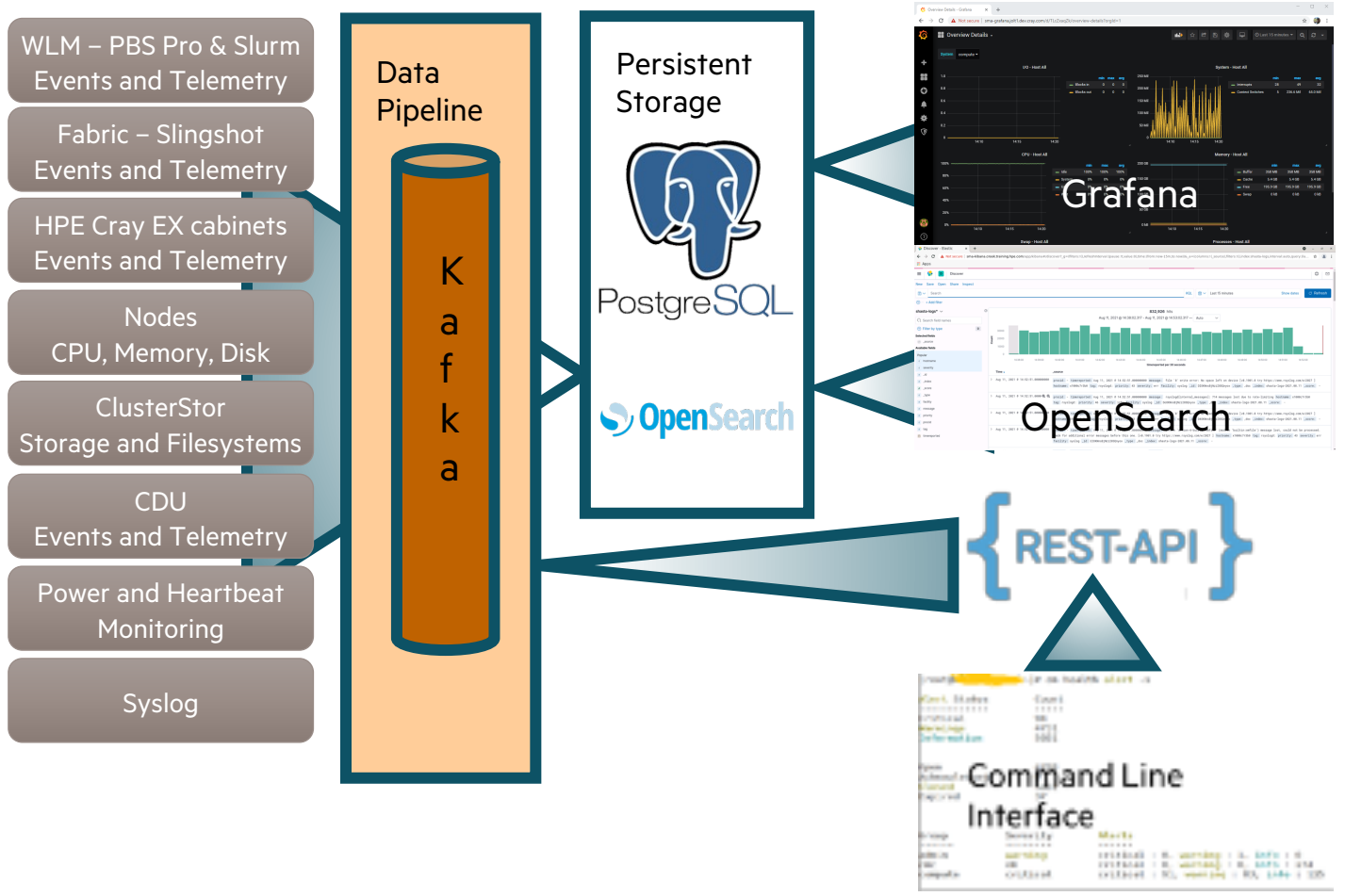
- **Monitoring and logging introduction**
- **Grafana basics**
- **System management health monitoring**
- **SMA monitoring**
- **Alerts**
- **Log analysis**
- **Ensuring continued system health**

Scalable monitoring and management

HPE Cray Systems management offers fine-grained centralized monitoring and management of your Exascale HPC system to keep it performing at its best



- In-band LDMS (Lightweight Distributed Metric Service) and out of band telemetry
- Access metrics and alerts via GUI, CLI, REST APIs
- Customize system telemetry and alerts to best suit your needs
- Set up automatic reactions to events to prevent failures
- Streaming data and persisted data for analytics such as anomaly detection



Grafana basics

- Grafana with CSM
- Refine existing dashboards
- Create new dashboards
- Grafana navigation
- Identify data sources for a dashboard

Grafana documentation

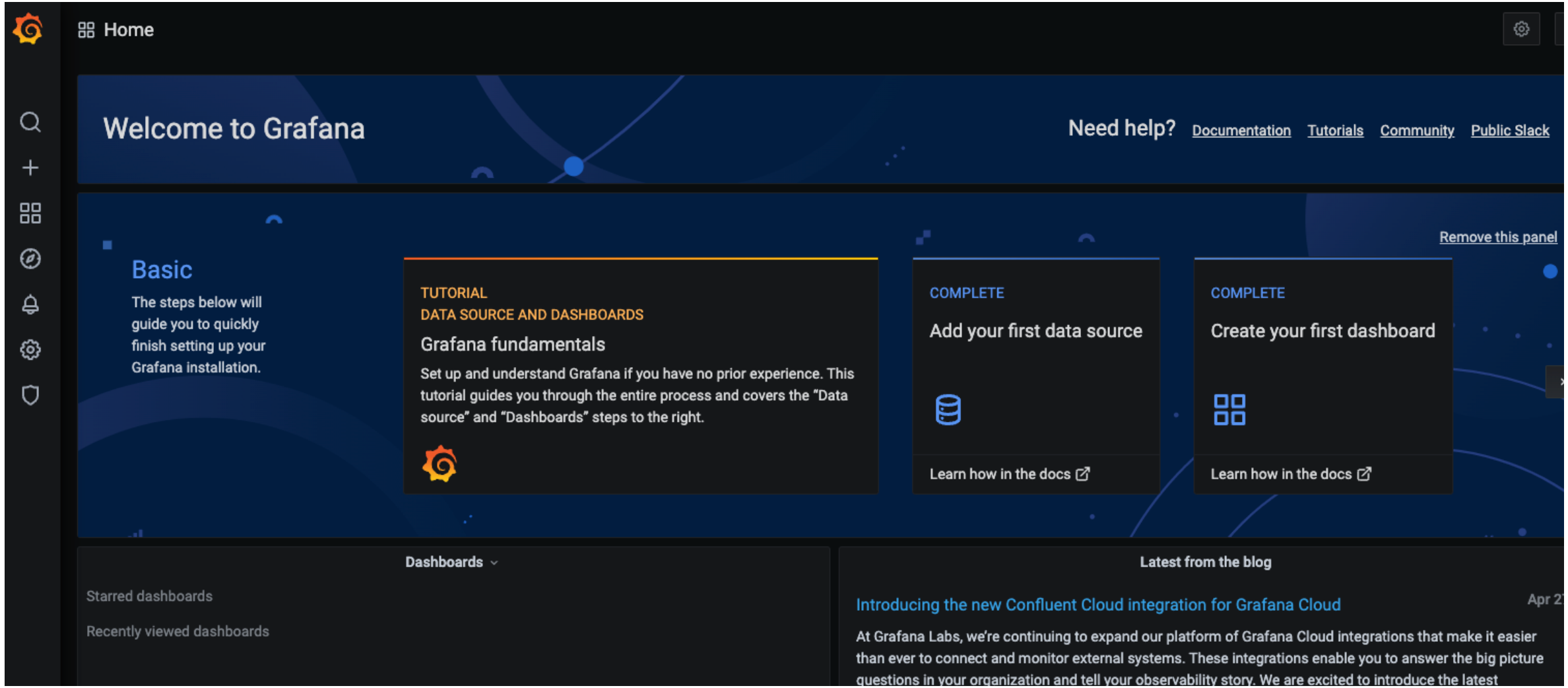
- Two Grafana instances are present on a CSM system
  - System Management Health Grafana
    - Includes numerous dashboards for visualizing metrics from prometheus and prometheus-istio

      https://grafana.cmn.SYSTEM_DOMAIN_NAME/
  - SMA Grafana
    - Includes system metric monitoring from these sources
      - Lightweight Data Monitoring Service (LDMS) statistics
      - HSN fabric performance, errors, congestion, and other statistics
      - Power, temperature and other sensor data from node, cabinet, and switch controllers

    https://sma-grafana.cmn.SYSTEM_DOMAIN_NAME/
  - Determine the external domain name by running the following command on any Kubernetes node:

```
ncn-m# kubectl get secret site-init -n loftsman \
-o jsonpath='{.data.customizations\.yaml}' | base64 -d | grep "external:"
external: SYSTEM_DOMAIN_NAME
```

System Management Health Grafana



Refine existing Grafana dashboards

- Time range of data
  - Upper right corner of screen shows chosen time range
  - Pull down to change time range
- Refresh frequency
  - Can set dashboard to not refresh or refresh at specific interval
- Button to immediately refresh dashboard
- Drilling into data
  - Upper left on some dashboards has datasource and fields that can adjust scope of data being displayed
    - Compute nodes or NCNs
    - Kubernetes namespaces and pods
    - Geolocation by cabinet ID or lower in the hierarchy of components
    - One device or multiple devices or devices in a cabinet ID
    - One or more ports on a switch
    - Etc.

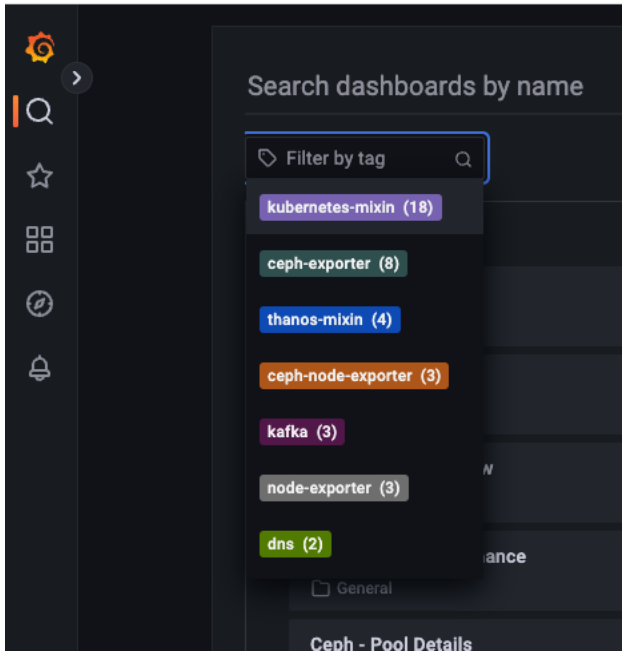
Create new Grafana dashboards

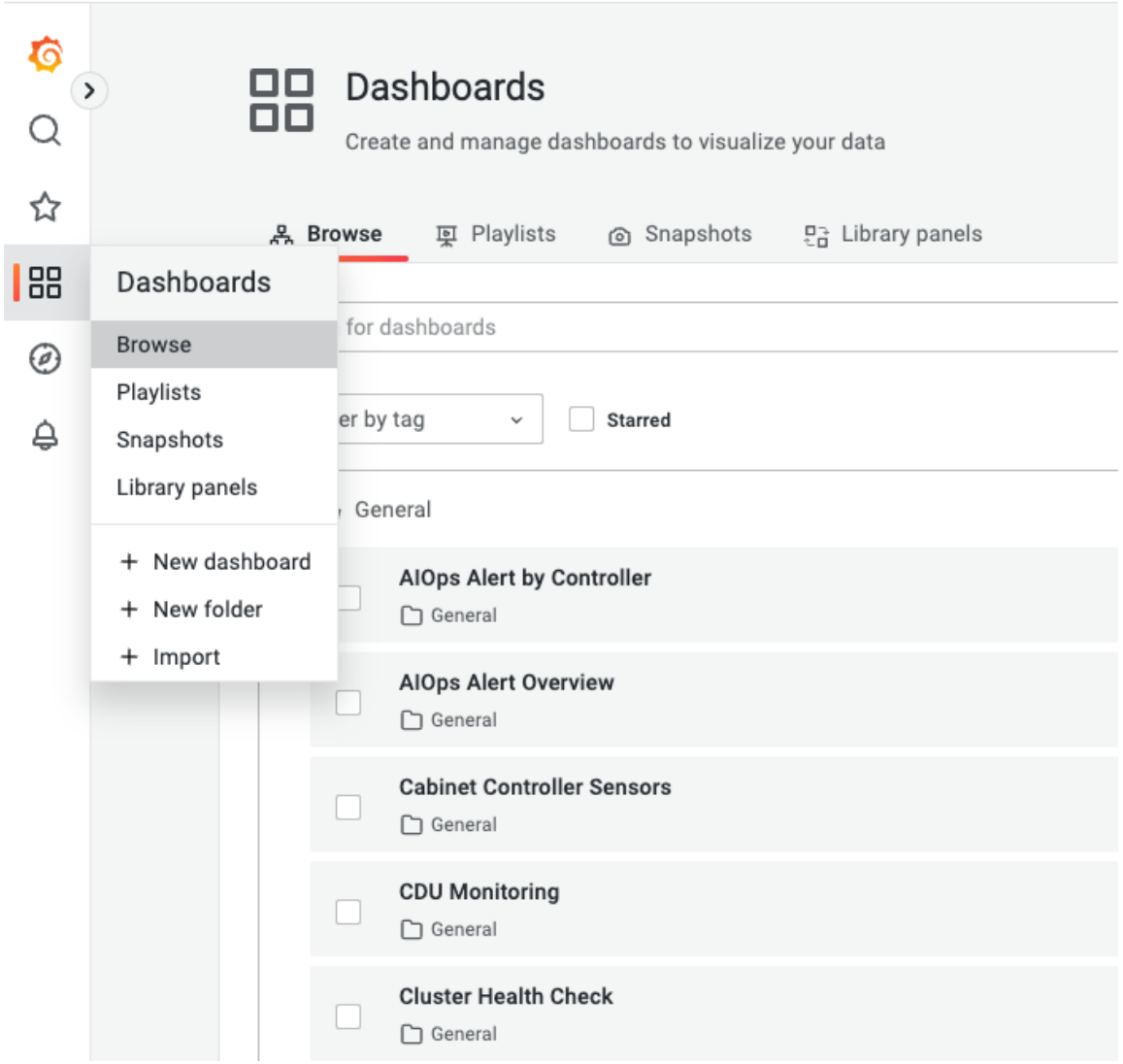
- Grafana's features and dashboard creation
  - https://grafana.com/docs/grafana/latest/
- Grafana panels and visualizations
  - The panel is the basic visualization building block in Grafana
  - Each panel has a query editor specific to the data source selected in the panel
  - The query editor allows you to build a query that returns the data you want to visualize.
  - https://grafana.com/docs/grafana/latest/features/panels/panels/
- Grafana dashboards
  - A dashboard is a set of one or more panels organized and arranged into one or more rows
  - https://grafana.com/docs/grafana/latest/features/dashboard/dashboards/

Grafana navigation

- Navigation bar on left side
  - Select 4 boxes to see list of dashboards
  - Choose browse

  - Select magnifying glass to search dashboards





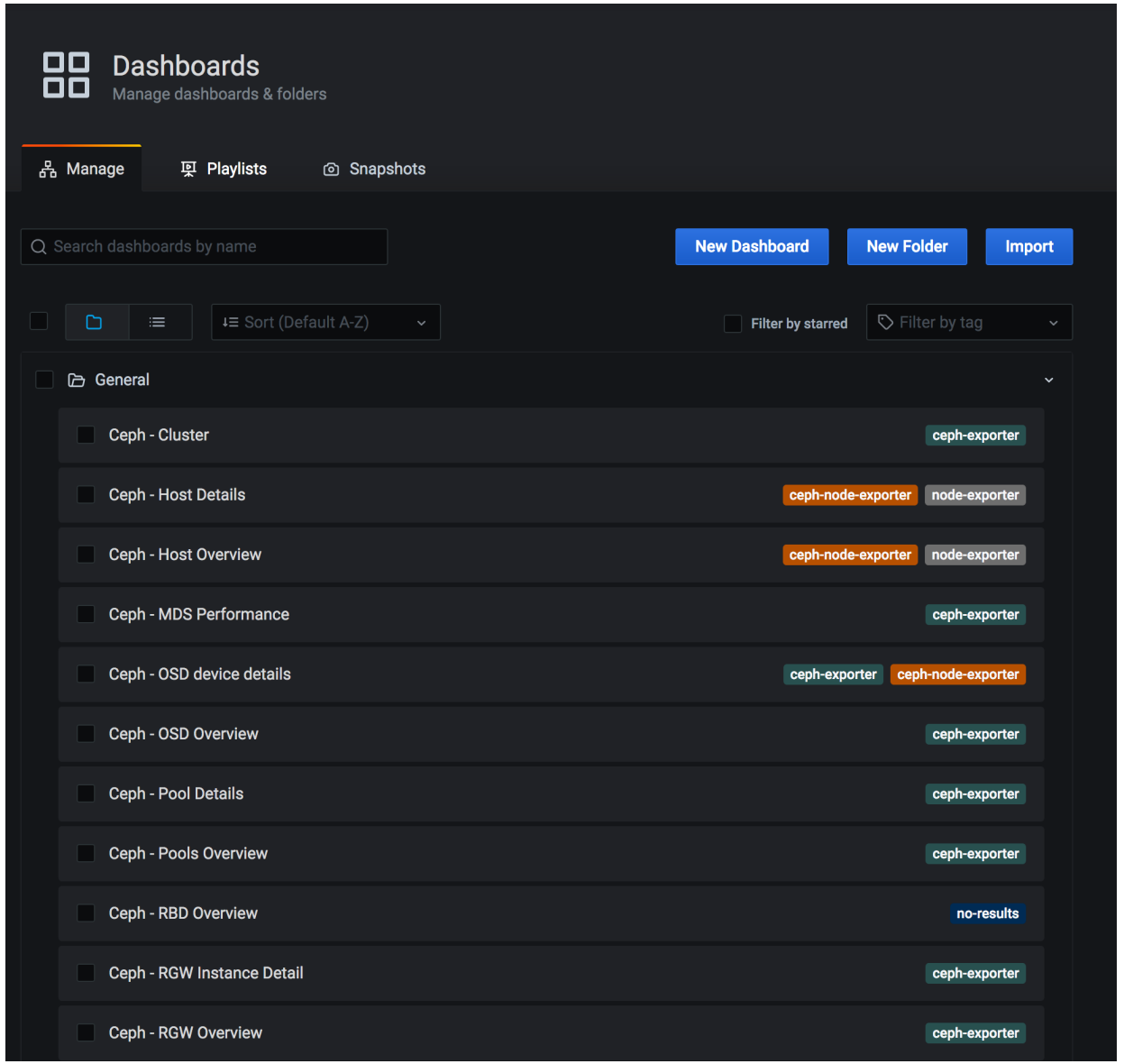
Data sources for Grafana dashboards

System Management Health Grafana dashboard sources

- amperage
- CANU
- ceph-exporter
- ceph-node-exporter
- coredns
- DNS
- fan speed
- GOSS
- hwmon
- IUF
- Kafka
- kea-dhcp
- Kubernetes-mixin
- Linux
- network
- node
- node-exporter
- power
- Prometheus
- resolver
- smartmon
- temperature
- Thanos-mixin
- unbound
- voltage

SMA Grafana dashboard sources

- AIops sub dashboard
- Alerta_dashboard
- crayFabricHealth
- DMTF
- fabric
- HMS
- HSN
- LDMS
- Slingshot
- Sub dashboard



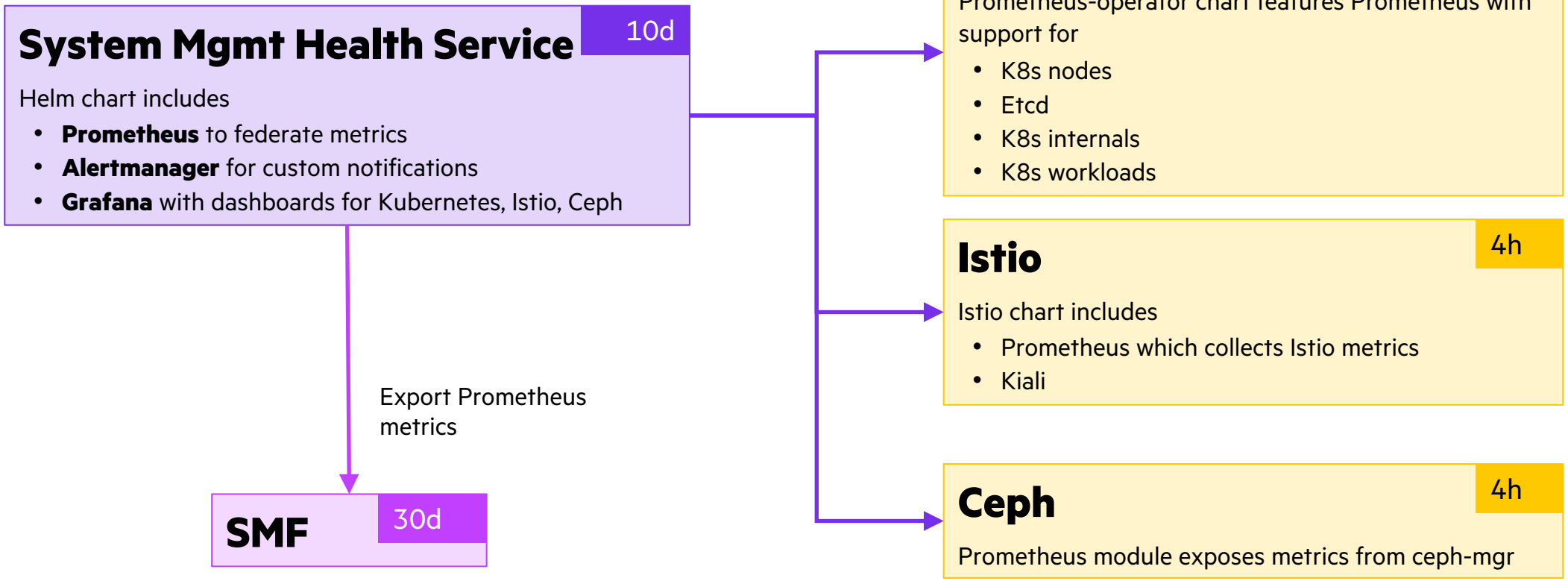
System management health monitoring

- System management health
- Grafana
- Sample Dashboards
- Kiali
- System testing
- CSM health checks
- CSM Diags

System Management Health Service

Is the system healthy?

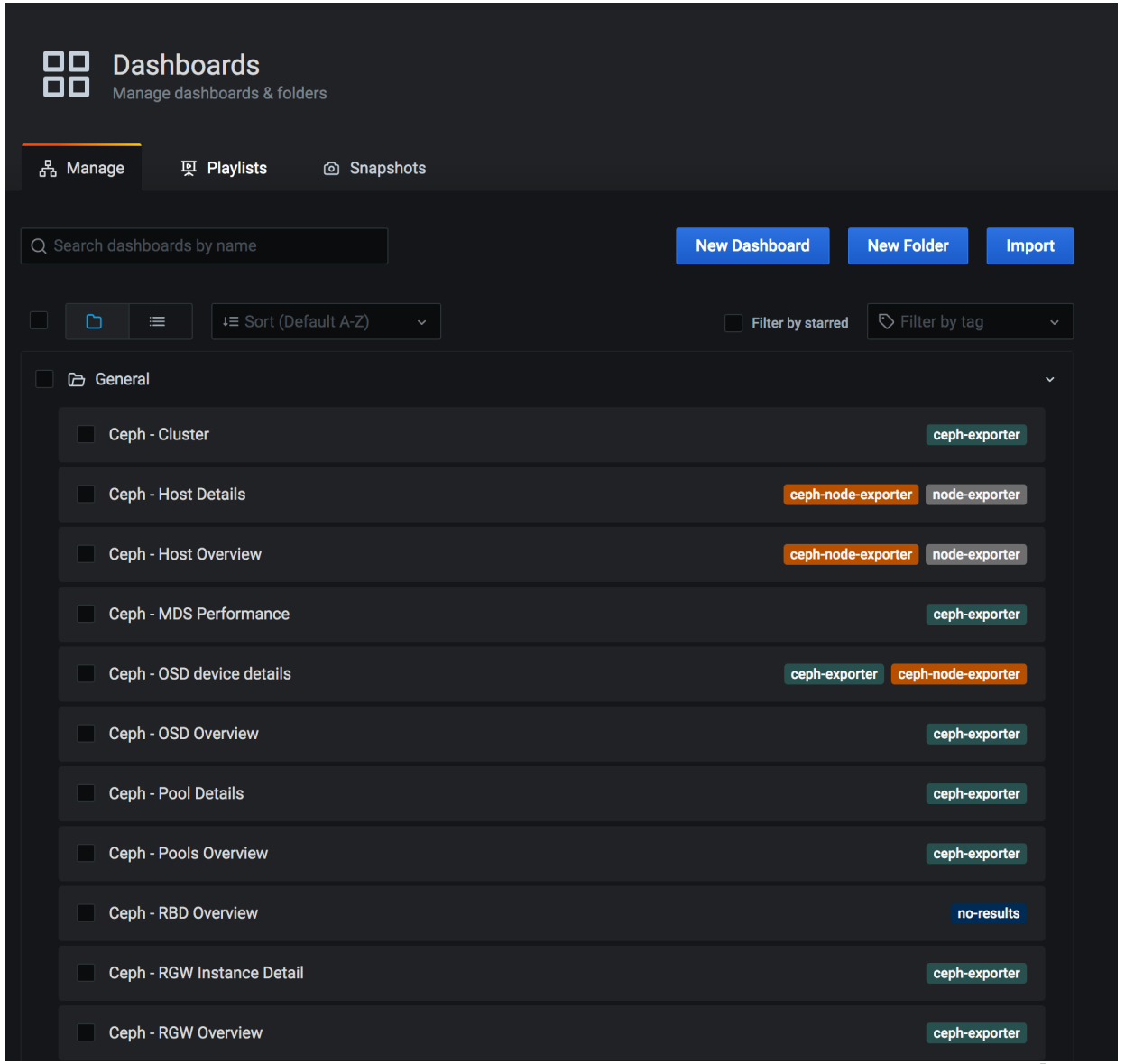
- Independent from the System Monitoring Framework (SMA)
  - Abbreviated on later slides
    - System Mgmt Health
- Does not monitor computes!



System Mgmt Health Grafana Dashboards

- Uses Keycloak authentication/authorization
- Secured with TLS sharing cluster certificate bundle
- About 40 included dashboards
  - Ceph
  - CoreDNS
  - Etcd
  - ETCD Clusters
  - Istio
  - Kea-dhcp
  - Kubernetes
  - Node Exporter
  - Nodes
  - PostgreSQL
  - Prometheus

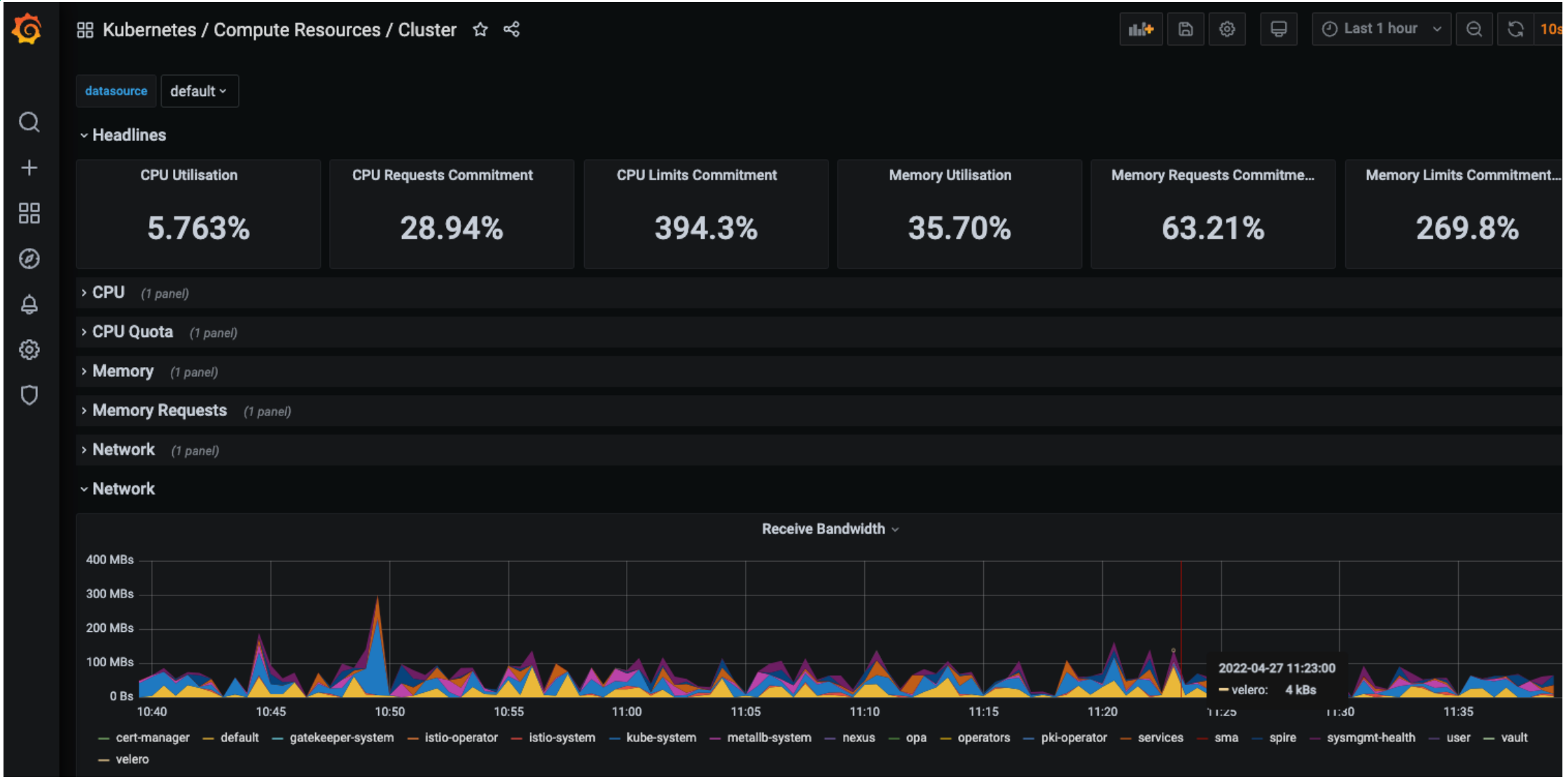
https://grafana.cmn.SYSTEM_DOMAIN_NAME/dashboards



System Mgmt Health Grafana Dashboards: ETCD

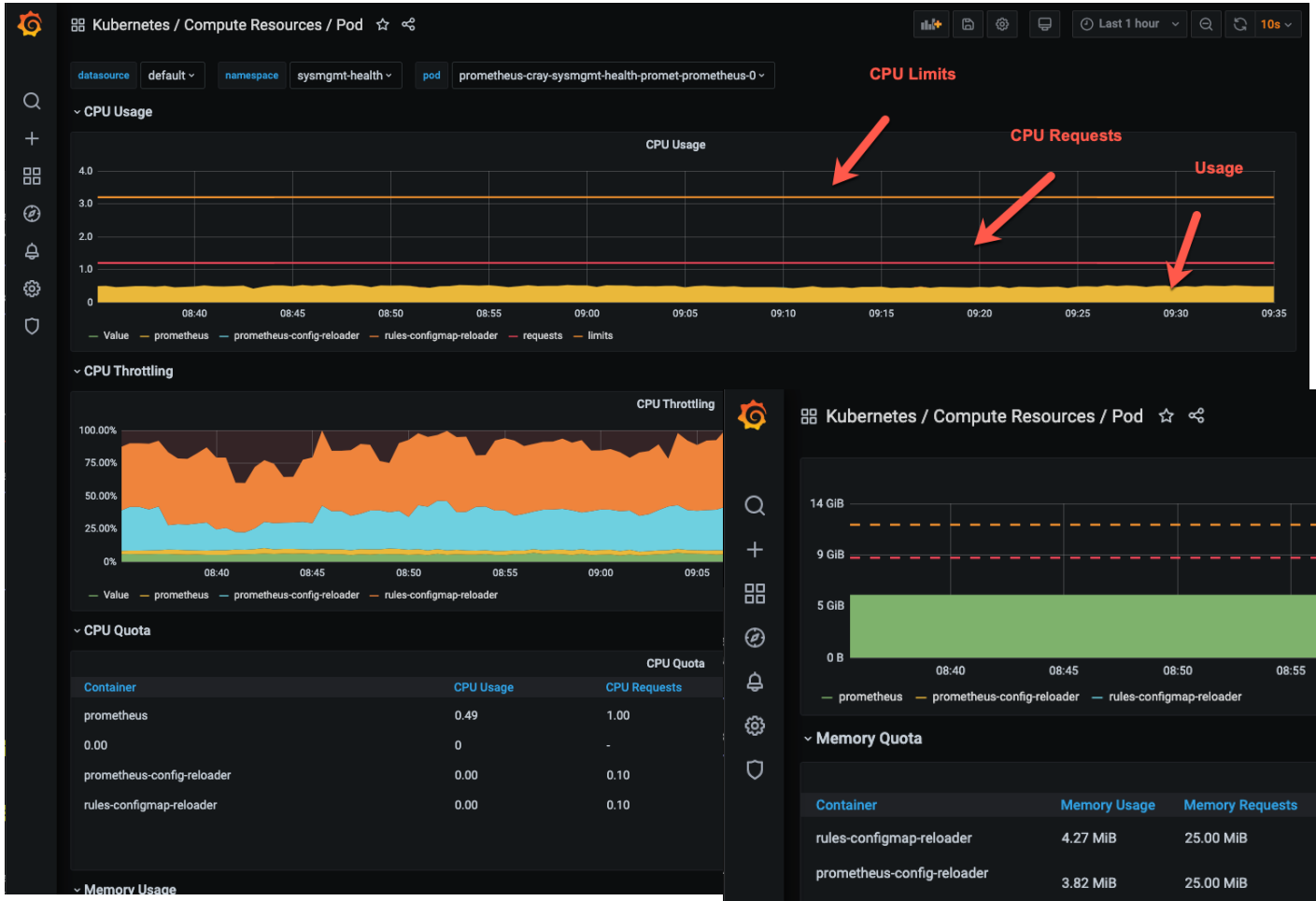
- Nodes up (quorum)
- RPC Rate
- Active Streams
- DB Size
- Disk Sync Duration
- Memory
- Client Traffic in
- Client Traffic Out
- Peer Traffic In
- Peer Traffic Out
- Raft proposals
- Total Leader Elections Per day

System Mgmt Health Grafana Dashboards: Kubernetes Cluster

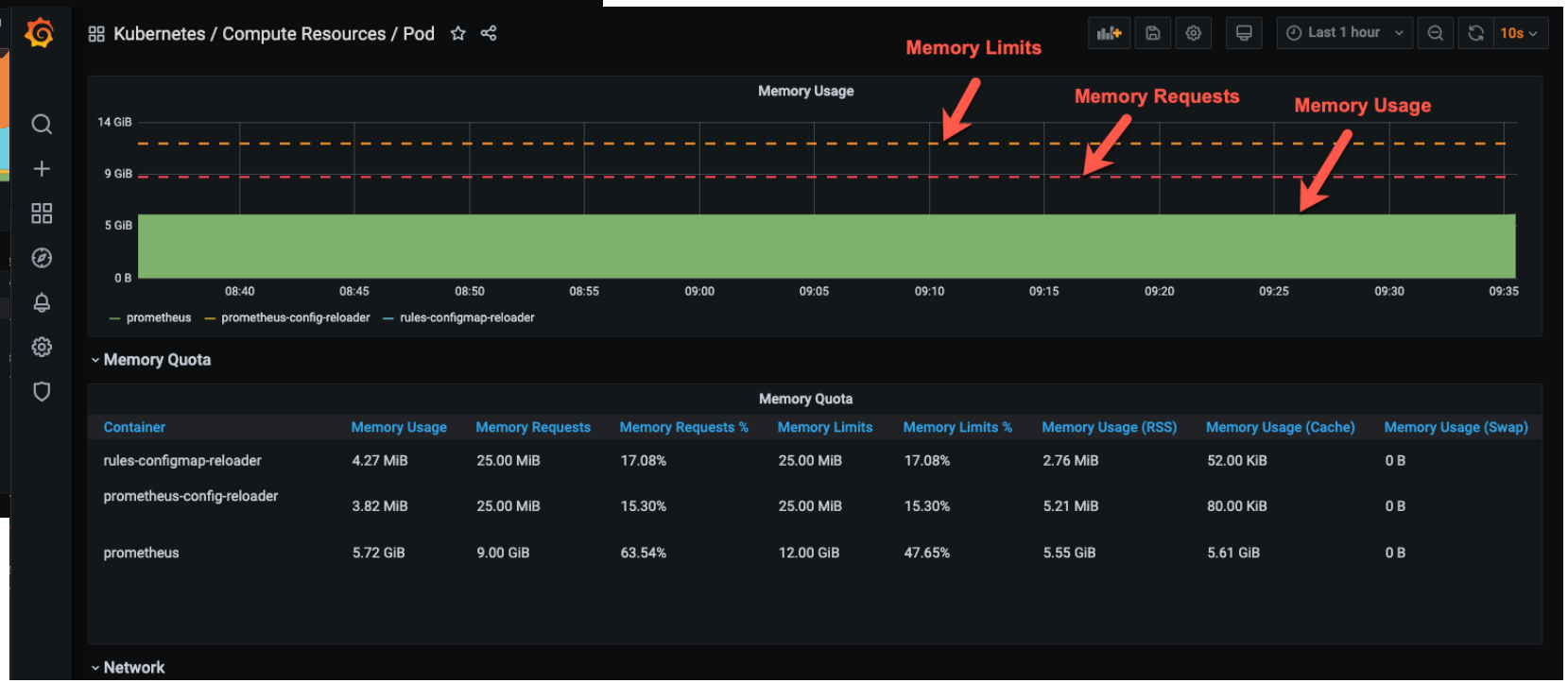


System Mgmt Health Grafana : Kubernetes pod Requests and Limits

CPU usage

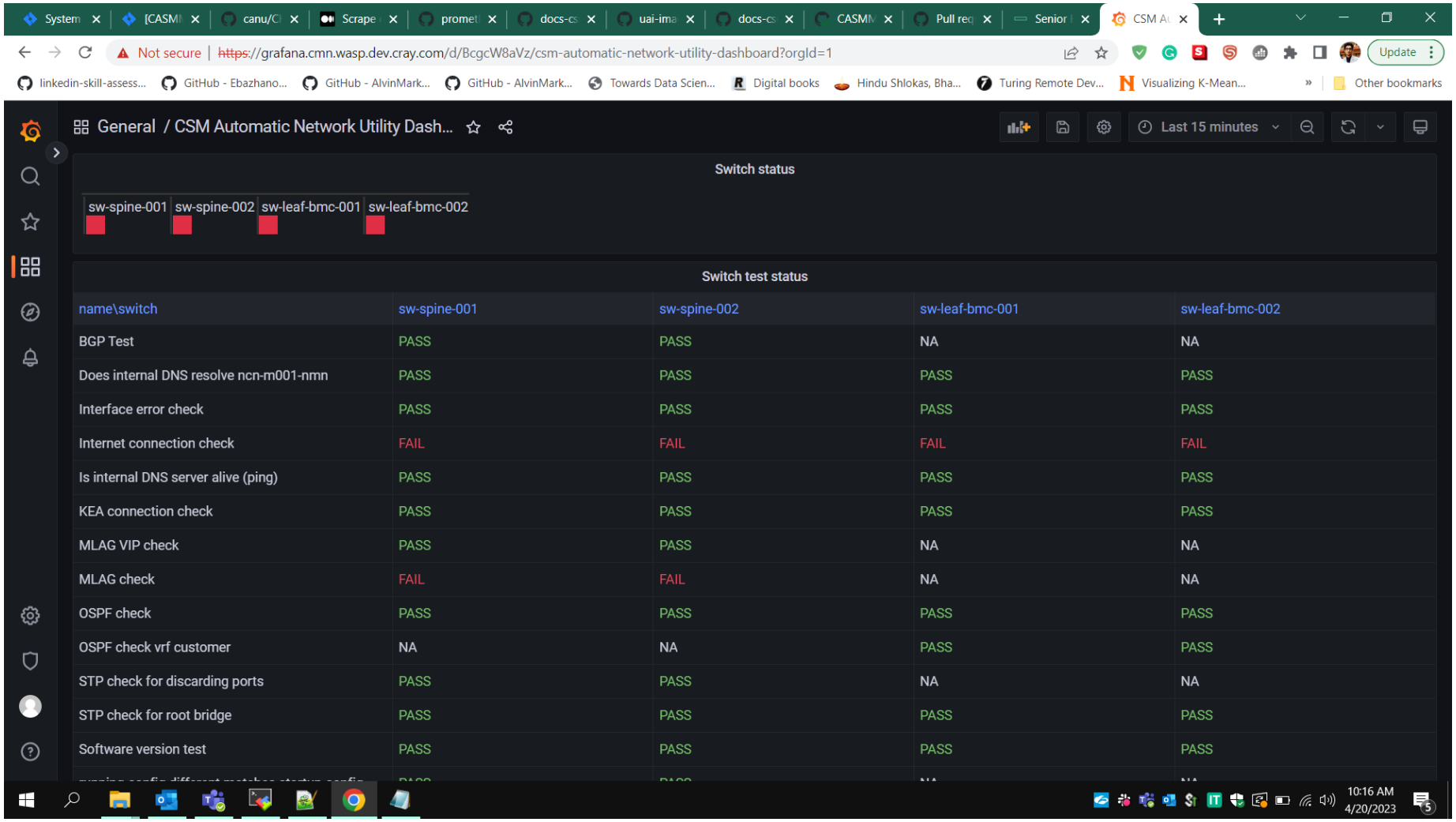


Memory Usage



System Mgmt Health Grafana Dashboards: CANU Dashboard

- CSM Automatic Network Utility (CANU) Dashboard
  - Shows results from CANU tests on management network switches
- Prometheus
  - Search for canu_test



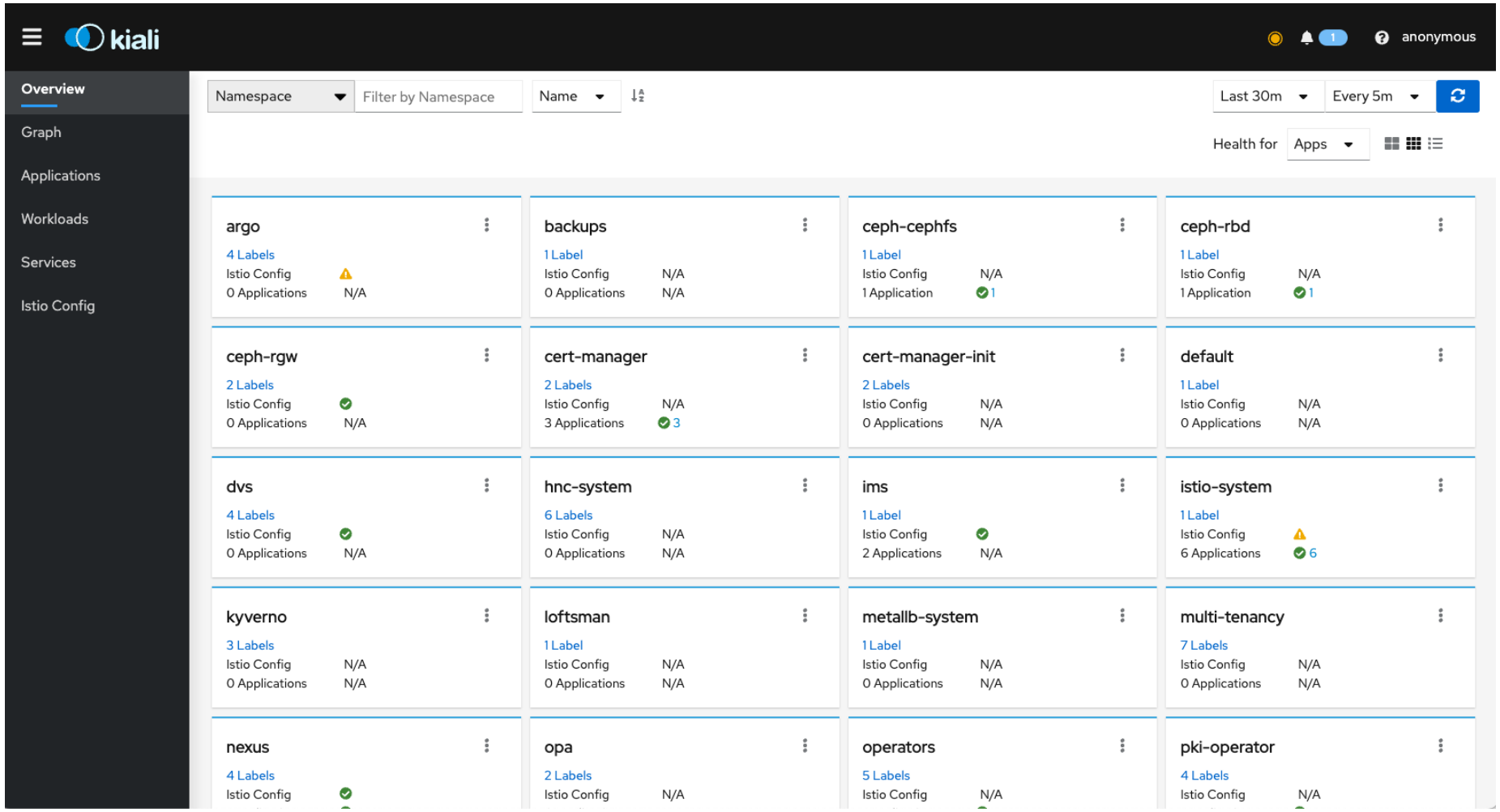
System Mgmt Health Grafana Dashboards: SMARTMON

Kiali

- Kiali provides real-time introspection into the Istio service mesh using metrics from prometheus-istio
  - Observability console for Istio with service mesh configuration and validation capabilities
  - Helps you understand the structure and health of your service mesh by monitoring traffic flow to infer the topology and report errors
  - Provides detailed metrics and a basic Grafana integration, which can be used for advanced queries
    – https://kiali-istio.SYSTEM_DOMAIN_NAME/
  - Documentation
    – https://kiali.io/documentation/

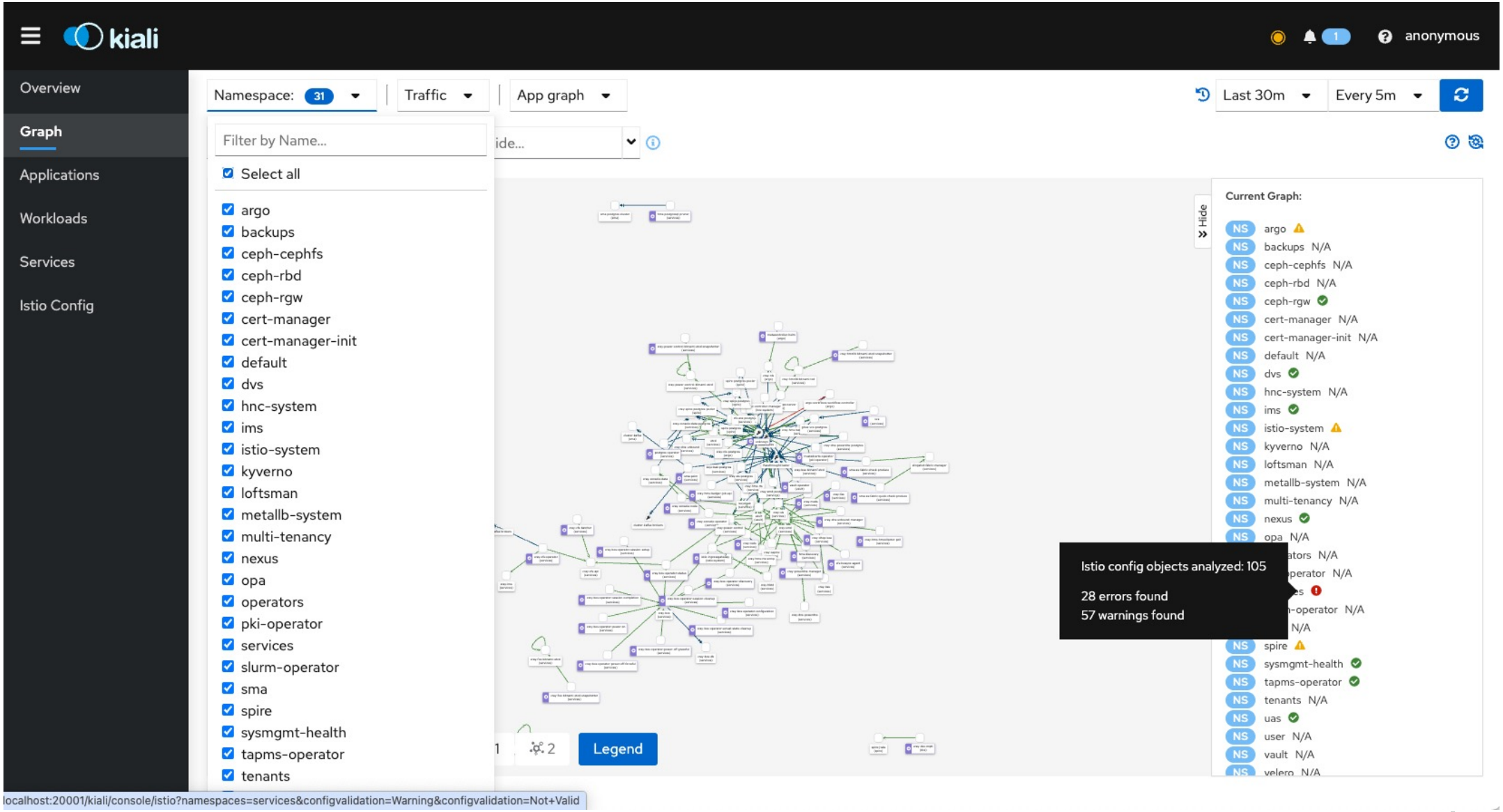
Kiali – Overview

- Identify namespaces with issues
- Summary of configuration health, component health and request traffic health
- Offers various filter, sort and presentation options



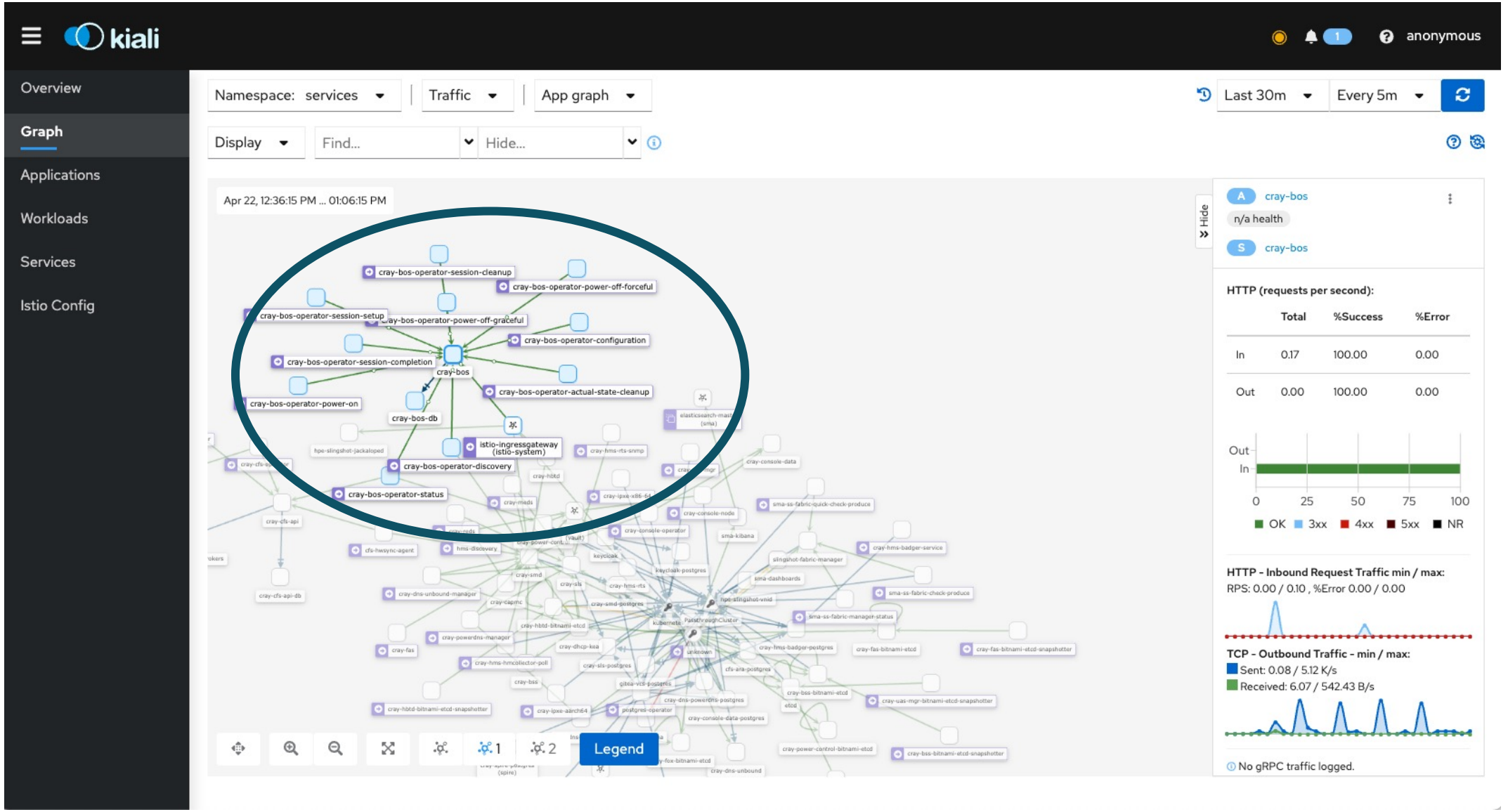
https://kiali.cmn.SYSTEM_DOMAIN_NAME

Kiali – Graph health



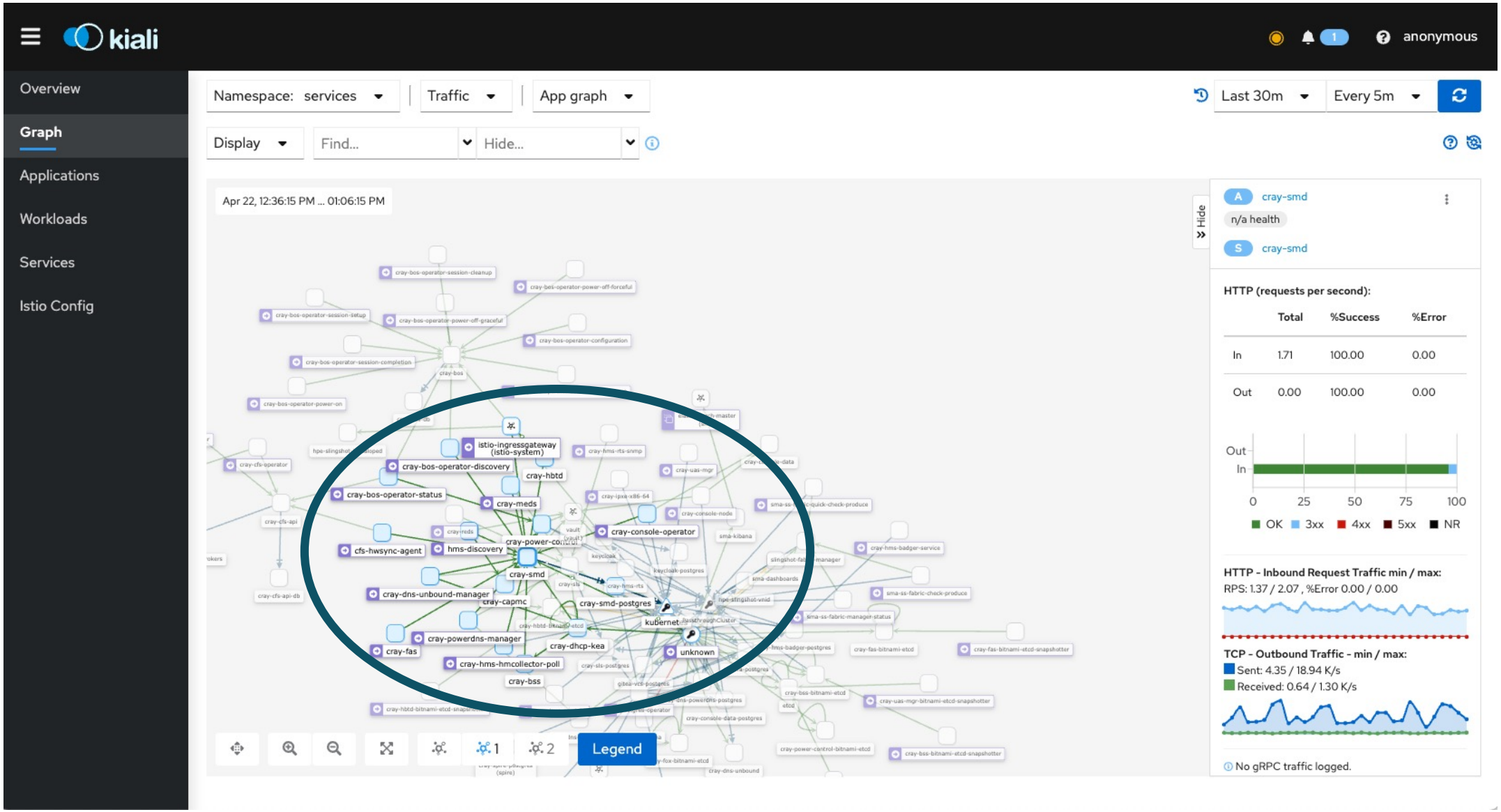
https://kiali.cmn.SYSTEM_DOMAIN_NAME

Kiali – cray-bos



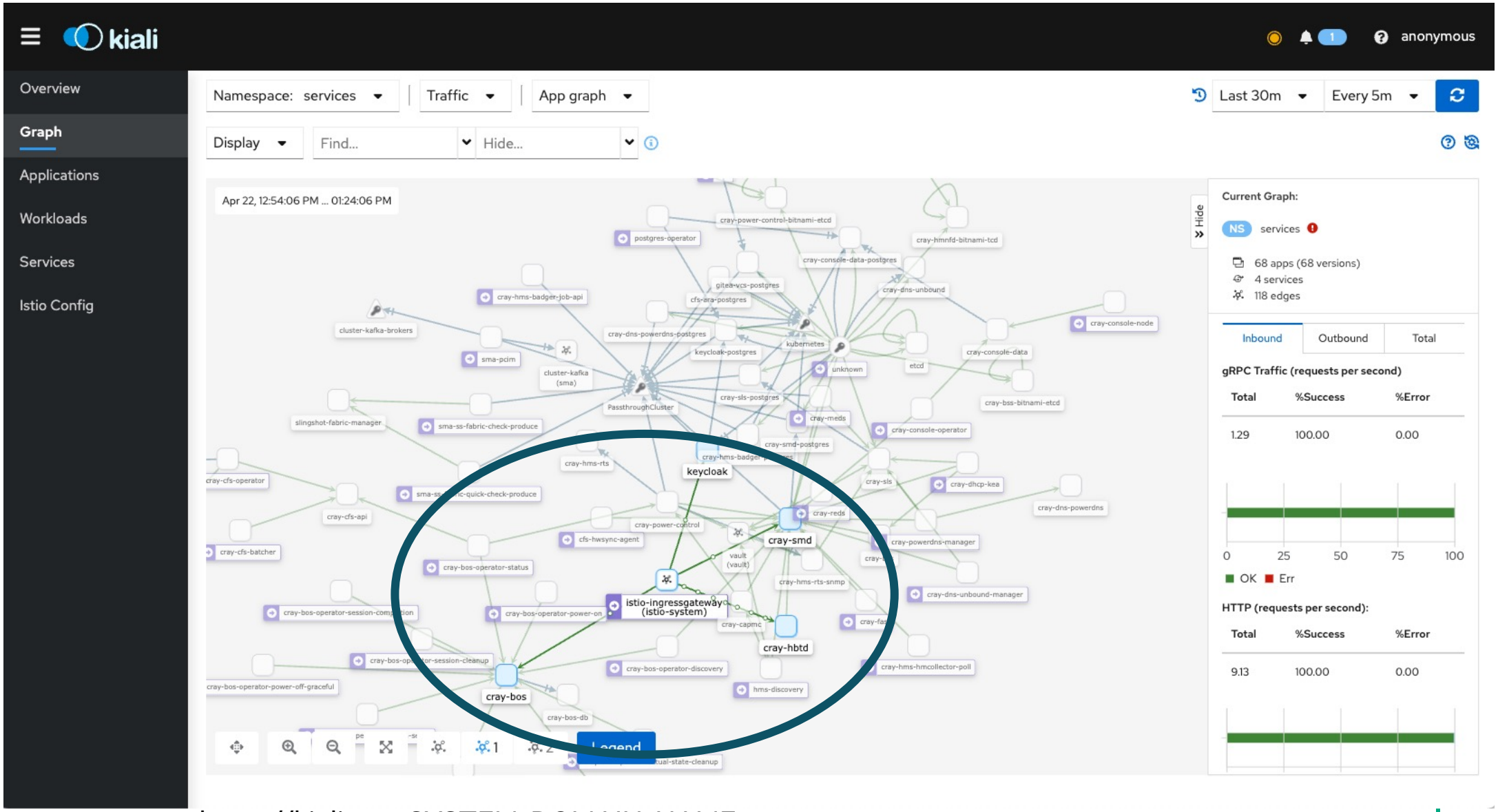
https://kiali.cmn.SYSTEM_DOMAIN_NAME

Kiali – cray-smd



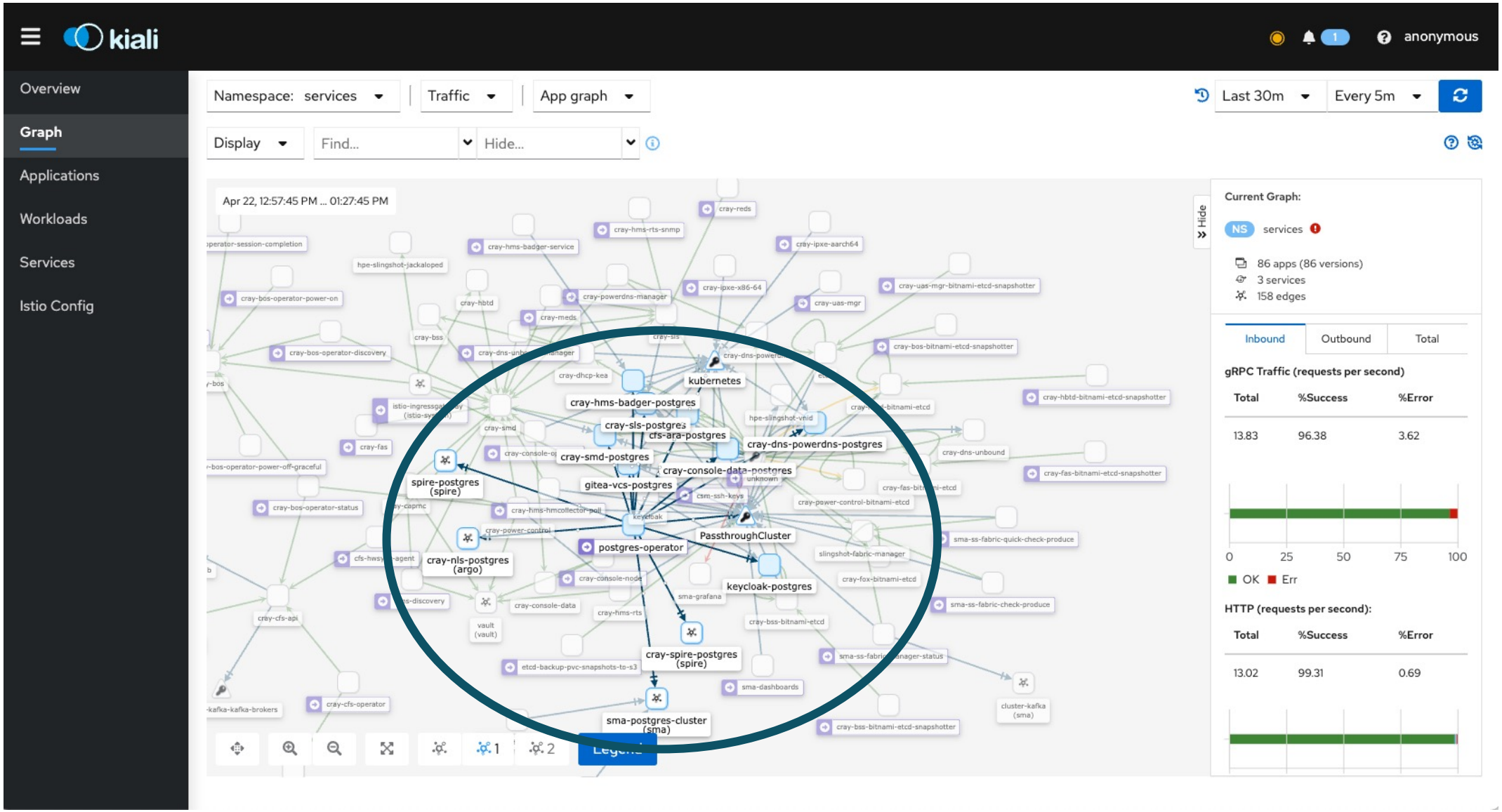
https://kiali.cmn.SYSTEM_DOMAIN_NAME

Kiali – istio-ingressgateway



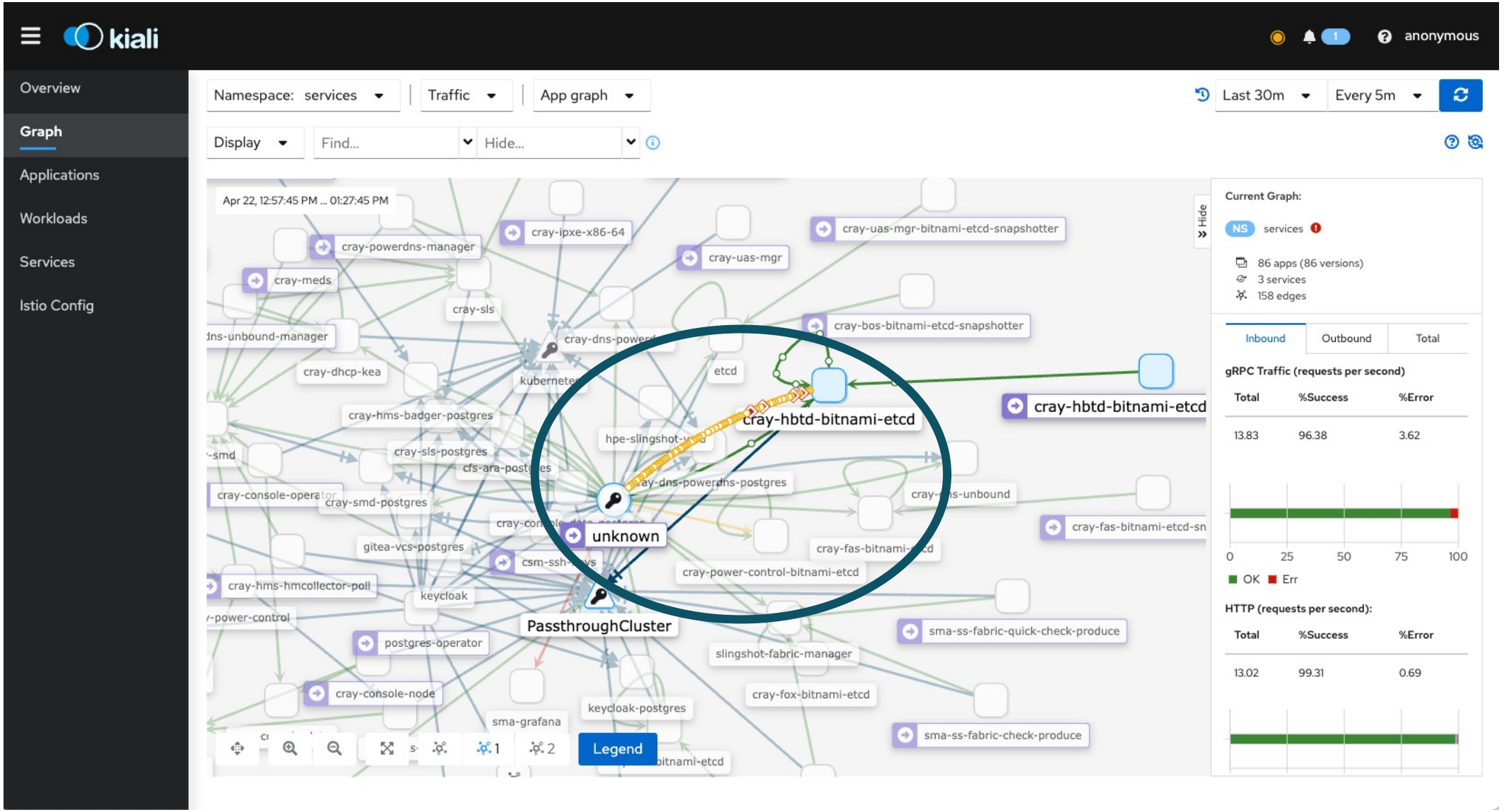
https://kiali.cmn.SYSTEM_DOMAIN_NAME

Kiali – postgres-operator



https://kiali.cmn.SYSTEM_DOMAIN_NAME

Kiali



https://kiali.cmn.SYSTEM_DOMAIN_NAME

System Testing: Validate CSM Health

- CSM documentation describes system health validation
  - See procedures in CSM documentation for your CSM release
    - CSM 1.3
      - https://cray-hpe.github.io/docs-csm/en-13/operations/validate_csm_health/
    - CSM 1.4
      - https://cray-hpe.github.io/docs-csm/en-14/operations/validate_csm_health/
    - CSM 1.5
      - https://cray-hpe.github.io/docs-csm/en-15/operations/validate_csm_health/
- When should you do this?
  - Run before rebooting or rebuilding a management node
  - Run before complete system graceful shutdown
  - Run as part of the complete system graceful startup
  - Run during complete system non-graceful startup
  - Run as part of troubleshooting toolbox

Goss testing

- Goss is a testing framework written in Golang
  - https://github.com/aelsabbahy/goss
- A Goss server runs on each Kubernetes management node
  - Some tests are used during installation
  - Some tests are used to validate system health post-installation by running script
    - Runs several tests in parallel across the management nodes
    - Reports failures in detail and overall pass/fail status for all test
- Passing test example

  Result: PASS

  Test Name: Etcd Cluster Endpoint Health

  Description: Checks the endpoint health of the etcd clusters.

  Severity: 0

  Test Summary: Command: etcd_database_health: exit-status: matches expectation: [0]

  Execution Time: 6.67716910 seconds

  Node: ncn-m003

Goss failing test example and summary

- Failing test will provide description and manual test method and, sometimes, remediation or link to troubleshooting documentation

  Result: FAIL

  Test Name: All Kubernetes Etcd Clusters have a Backup Created within the Last 24 Hours.

  Description: If this test fails, run the script "goss-testing/scripts/etcd_backups_check.sh -p" to see a printed description of the errors. Check that cron jobs are running and creating backups periodically with the command "kubectl -n operators get cronjob kube-etcd-periodic-backup-cron". To see the last scheduled time, run the command "kubectl -n operators get cronjob kube-etcd-periodic-backup-cron -o jsonpath='{.status}'". To restart the cronjob, execute "kubectl -n operators get cronjob kube-etcd-periodic-backup-cron -o json | jq 'del(.spec.selector)' | jq 'del(.spec.template.metadata.labels."controller-uid")' | jq 'del(.status)' | kubectl replace --force -f -".

  Severity: 0

  Test Summary: Command: k8s_etcd_backups_check: stdout: patterns not found: [PASS]

  Execution Time: 0.00001914 seconds

  Node: ncn-m003

- Overall summary
  - Total number of automated tests to be run increases from CSM 1.3 to CSM 1.5

    Total Tests: 1008, Total Passed: 1005, Total Failed: 3, Total Execution Time: 38.8861 seconds

System Testing: CSM Health checks

- Platform Health Checks
  - CSM 1.5, CSM 1.4, CSM 1.3
    - Automated combined NCN and Kubernetes checks using Goss servers on each NCN
      `/opt/cray/tests/install/ncn/automated/ncn-k8s-combined-healthcheck`
  - CSM 1.2
    - Automated NCN checks using Goss servers on each NCN
      `/opt/cray/tests/install/ncn/automated/ncn-healthcheck`
    - Automated Kubernetes check using Goss servers on each NCN
      `/opt/cray/tests/install/ncn/automated/ncn-kubernetes-check`
  - Manual ncnHealthChecks
    `/opt/cray/platform-utils/ncnHealthChecks.sh -s SUBOPTION`
    `ncn_uptimes, node_resource_consumption, pods_not_running, etcd_health_status, etcd_cluster_balance, etcd_database_health`
  - Manual ncnPostgresHealthChecks
    `/opt/cray/platform-utils/ncnPostgresHealthChecks.sh`
  - Check that system management monitoring tools are able to display data
    - Prometheus, Alertmanager, Grafana, Kiali
  - BGP Peering Status and Reset

System Testing: More CSM Health checks

- Hardware Management Services
  - HMS Test execution

    `/opt/cray/csm/scripts/hms_verification/run_hms_ct_tests.sh`
  - HSM Discovery Validation

    – CSM 1.5

    `/opt/cray/csm/scripts/hms_verification/hsm_discovery_status_test.sh`

    – CSM (all versions)

    `/opt/cray/csm/scripts/hms_verification/verify_hsm_discovery.py`
  - Check for node Warning and Alert states in HSM

    – CSM 1.5

    `/opt/cray/csm/scripts/hms_verification/run_hardware_checks.sh`
- Software Management Services
  - BOS, TFTP, cray-console, IMS, CFS, VCS, CRUS

    `-/usr/local/bin/cmsdev test -q all`
- Gateway health and SSH access checks
  - Gateway health tests from NCN

    `/usr/share/doc/csm/scripts/operations/gateway-test/ncn-gateway-test.sh`
  - Gateway health tests from outside the system
  - Internal SSH access

    `/usr/share/doc/csm/scripts/operations/pyscripts/start.py test_bican_internal`
  - External SSH access
- Booting CSM Barebones image
  - Tests whether the booting services infrastructure is functional to boot a compute node
- UAS/UAI tests
  - Validate basic UAS installation
  - Validate UAI creation
  - Troubleshooting UAS/UAI

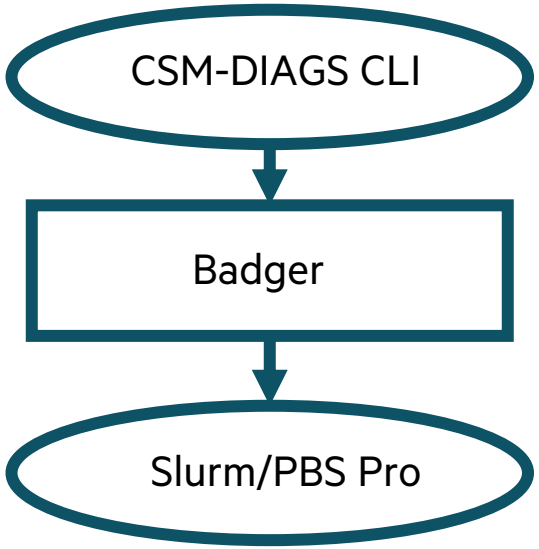
System Testing: CSM Diags

A set of diagnostic tools to perform various node level and system wide tests on compute nodes

- Functional test suites and performance test suites with both MPI and non MPI test suites
- Tests initiated using cray-hms-badger service to submit WLM jobs on compute nodes
  - Consistency checks
  - System Level Diagnostics
  - Nvidia GPU Diagnostics
  - AMD GPU diagnostics
  - Fabric diagnostics
  - OSU Benchmarks
  - CANU checks (CSM 1.5)
- sdiag_run.py using cray-hms-badger
  - Execute multiple diagnostics (MPI, NON_MPI, GPU, Slingshot) in one shot on multiple compute nodes

CSM Diags - CLI

- A CLI (which uses Badger framework) has been provided on the worker nodes to execute multiple diagnostics (MPI, NON_MPI, Slingshot) in a single instance on multiple compute nodes
  - Admin needs to modify the configuration files, with the list of diagnostics that need to be executed
    - sdiag-list.json (List of diagnostics which Admin needs to run)
    - sdiag-arguments.json (Argument Values for each Diagnostic Test)
    - sdiag-gumball.json (Badger Information, Session directory)
    - nodes (file with the list of node xnames or nids)
    - nodes_gpu (file with the list of GPU node xnames or nids)



- Can be run by:

```
ncn-w# /opt/cray/csm-diags/sdiag_run.py
  out_02:12:02.txt output file has been created in /var/log/cray/shasta-diag
  -Execution completed
       gpu-burn: cray badger sessions describe e5d5b58a-f63e-45a5-b3cd-3b383ecdd1df'
```

```
rocket-ncn-w001:~ # cray badger sessions describe "e5d5b58a-f63e-45a5-b3cd-3b383ecdd1df"
notFound = []
loopSuiteUntilTimestamp = ""
analysisStatus = "PASSED"
finishTimestamp = "2020-09-08T04:33:31.977125Z"
underUtilizedNodes = []
cleaned = false
```

```
output_533.0_0_nid001034: TEST has PASSED
GPU 0 - Max Gflops : 16249 , Max Temp : 61 C , Health : OK , Errors: 0
GPU 1 - Max Gflops : 16414 , Max Temp : 52 C , Health : OK , Errors: 0
GPU 2 - Max Gflops : 16172 , Max Temp : 59 C , Health : OK , Errors: 0
GPU 3 - Max Gflops : 17499 , Max Temp : 62 C , Health : OK , Errors: 0
output_533.1_0_nid001033: TEST has PASSED
GPU 0 - Max Gflops : 16271 , Max Temp : 56 C , Health : OK , Errors: 0
GPU 1 - Max Gflops : 16300 , Max Temp : 52 C , Health : OK , Errors: 0
GPU 2 - Max Gflops : 16130 , Max Temp : 58 C , Health : OK , Errors: 0
GPU 3 - Max Gflops : 16492 , Max Temp : 50 C , Health : OK , Errors: 0
```

SMA Monitoring

- System Monitoring Framework (SMF)
- SMA Grafana

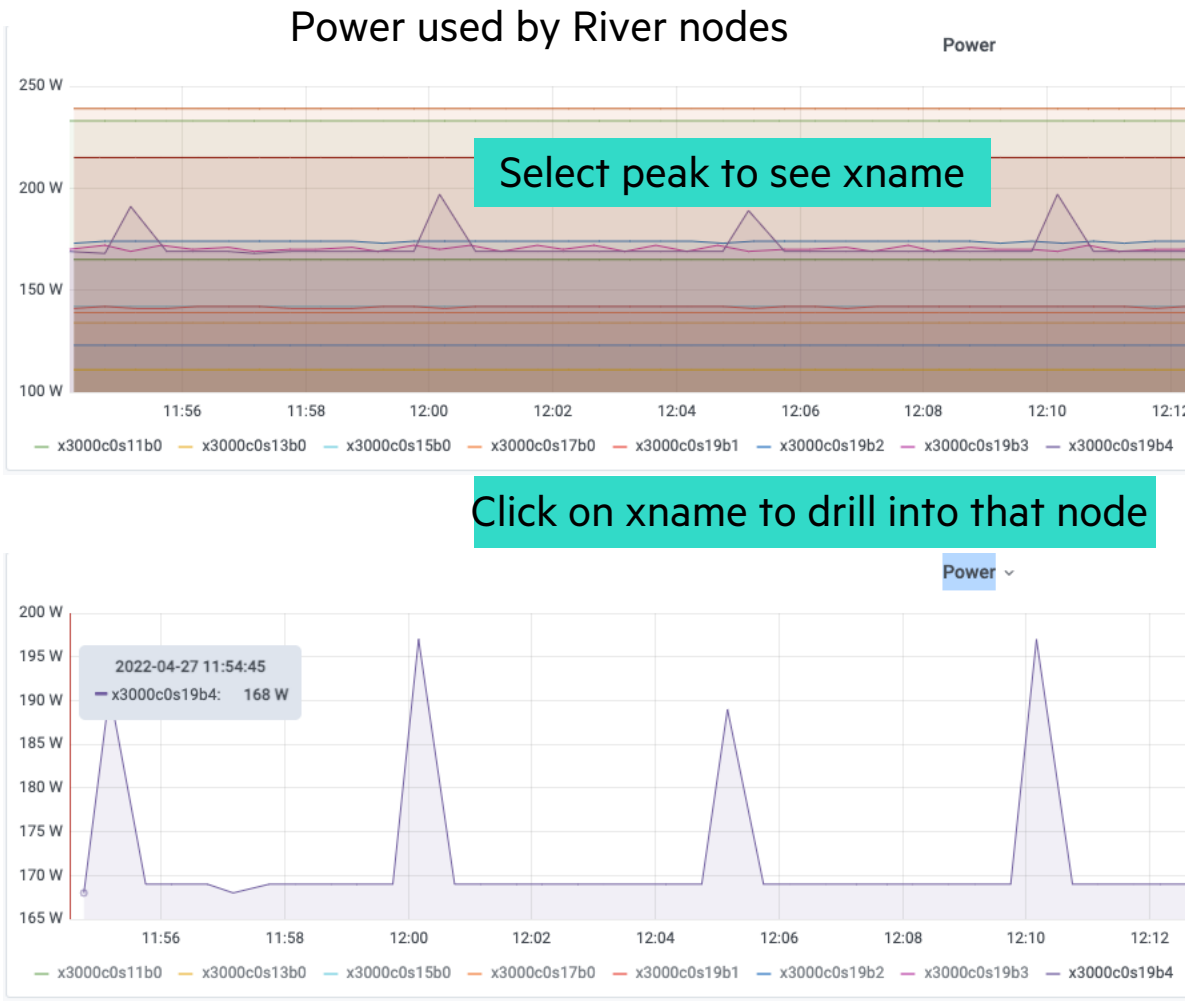
System monitoring Framework

- Tightly-integrated monitoring system
- Provides detailed telemetry information from multiple subsystems:
  - Fabric
  - Environmental
  - Network
  - Storage
  - Operating systems (vmstat and iostat metrics)
- Incorporates the context necessary to understand telemetry data
- Feeds into a common message bus (Kafka), persistence, and minimal UI infrastructure
- SMA alarms and notifications subsystem monitors metric data
  - Provides a way to notify administrators when select metric data is outside of normal operating values
  - SMA includes several pre-defined alarms
  - Can be extended with site defined alarms

SMA-Grafana Dashboards

- About 20 included dashboards
  - System CPU, I/O, Kernel, Memory, Processes, Swap
  - Cabinet Controller Sensors
  - CDU Monitoring
  - Fabric Telemetry
  - Fabric Performance Telemetry
  - Fabric Critical Telemetry
  - Fabric Switch Hardware Telemetry
  - Node Controller Sensors
  - Overview Details
  - Overview Device I/O Stats
  - PDU Monitoring
  - Redfish Events
  - River Sensors
  - Switch Controller Sensors
  - System Monitoring Dashboard
  - Cluster Health Check (Alerta alerts)

https://sma-grafana.cmn.SYSTEM_DOMAIN_NAME/dashboards



SMA New Grafana dashboards

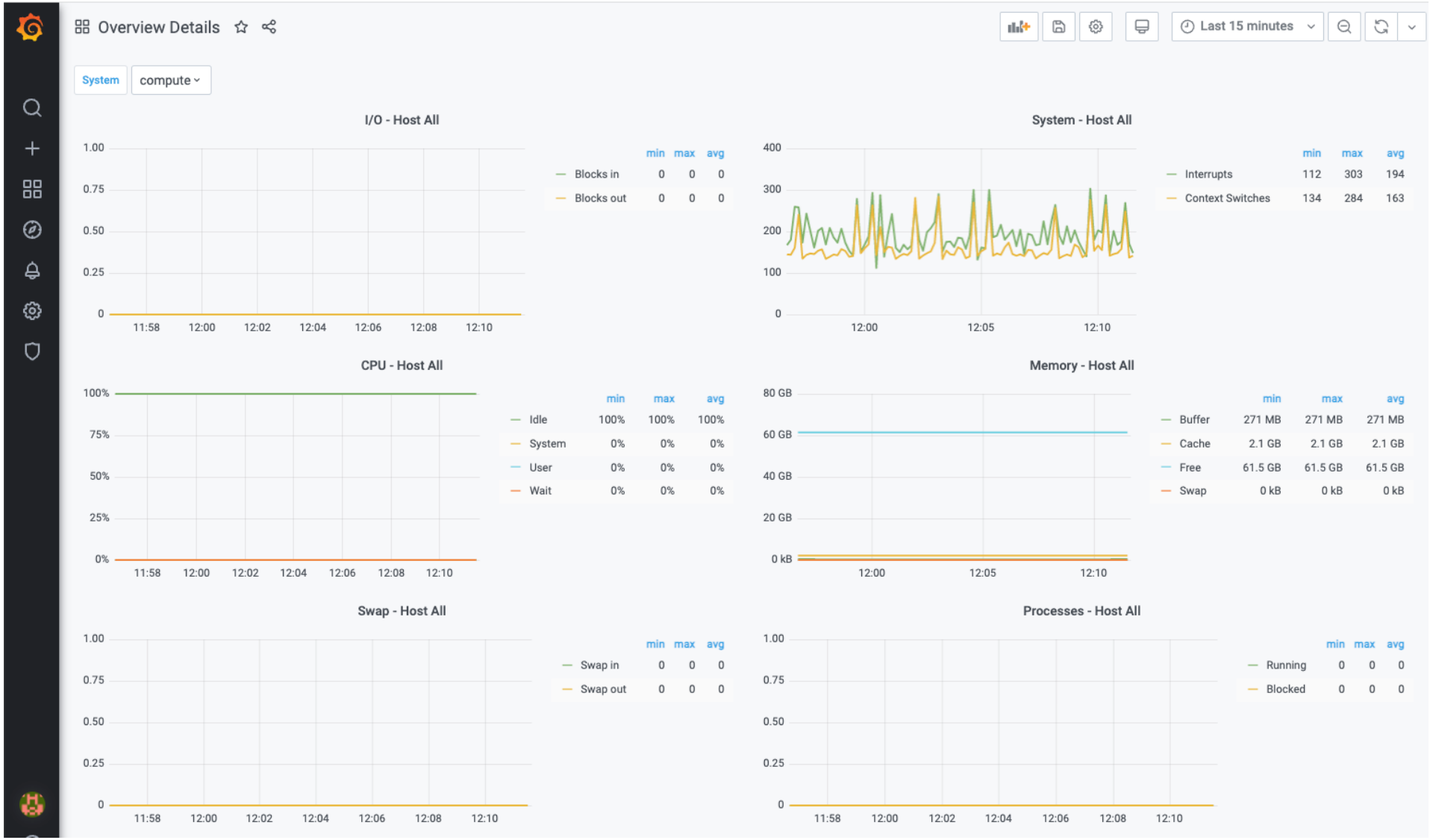
SMA 1.9/CSM 1.5

- See Monitoring Cooling Devices with Artificial Intelligence for IT operations
  - AIOps Anomaly Forecast
  - AIOps Slingshot Physical Context Congestion
  - AIOps Slingshot Physical Context Congestion Details
  - AIOps Slingshot Physical Context Temperature Details
  - AIOps Univariate Dashboard

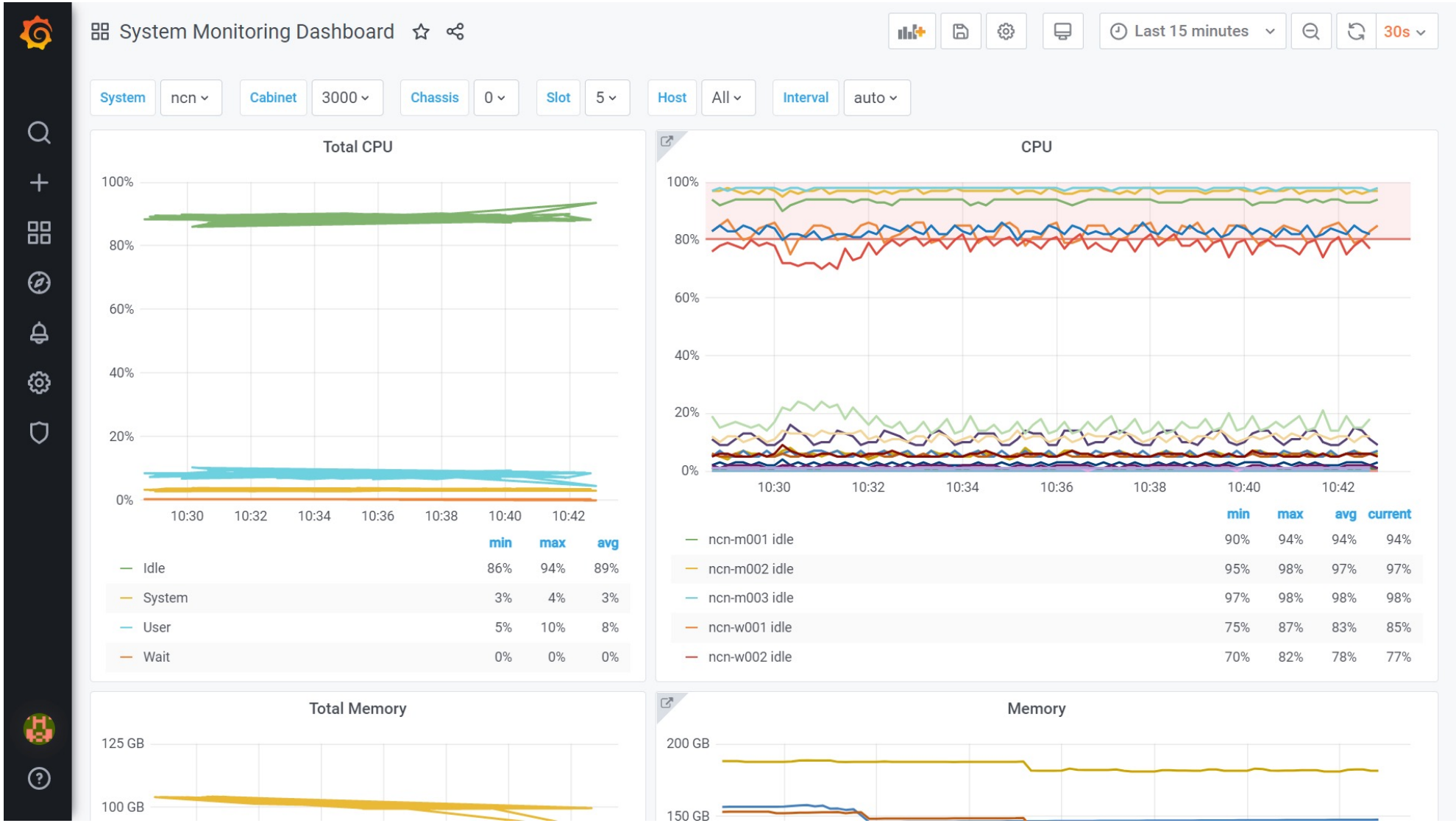
SMA 1.8/CSM 1.4

- CDU Monitoring
- CPU Power Monitoring
- CPU Temperature Monitoring
- GPU Temperature Monitoring
- Prometheus Alerts
- Slingshot Port Flap
- Slingshot Port State
- Slingshot Bit Error Rate (BER)
- Slingshot rxCongestion
- Slingshot rxBW/txBW (Receive/Transmit Bandwidth)
- Slingshot Routing Error
- Slingshot Hard Error

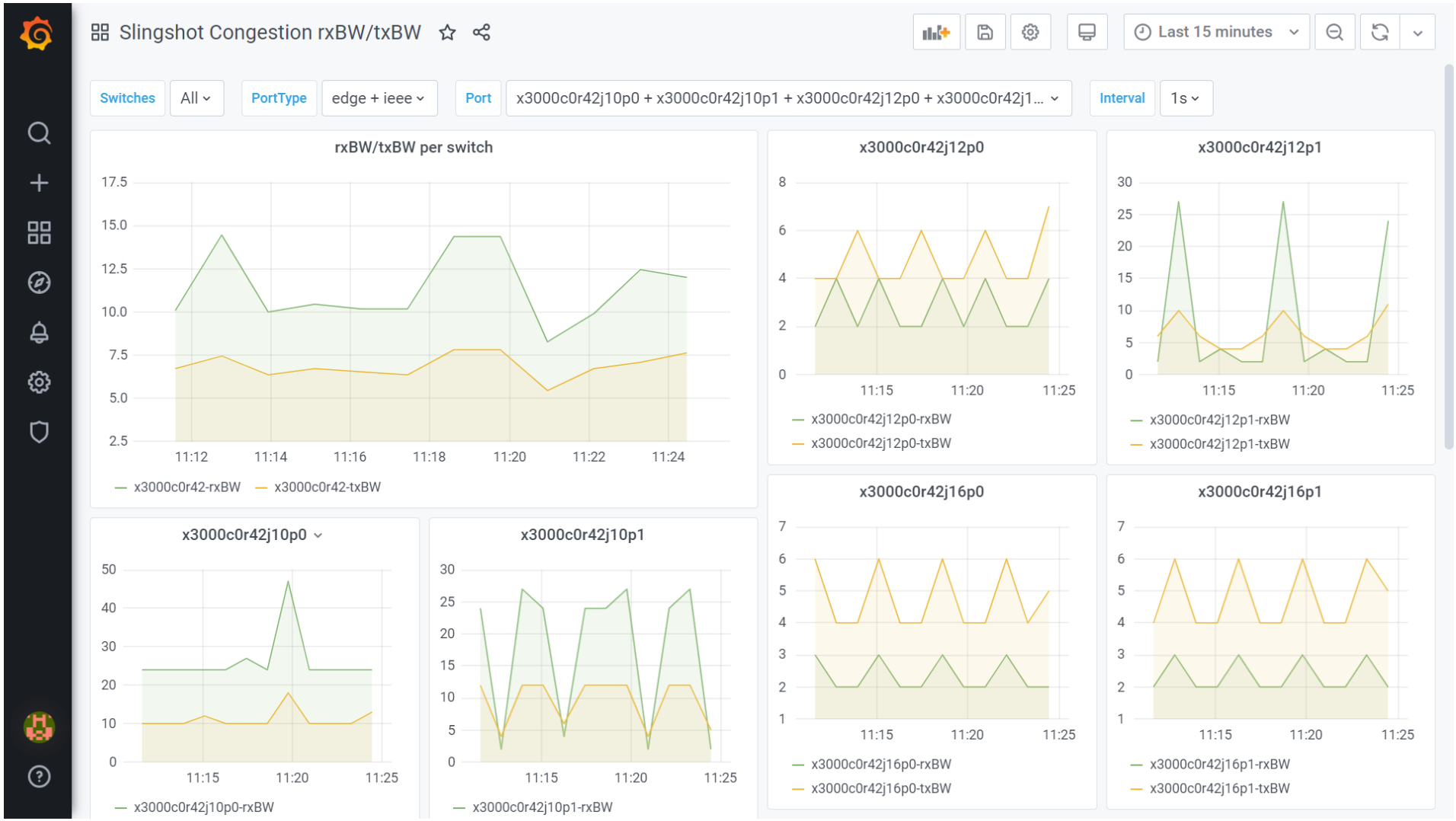
SMA-Grafana overview details



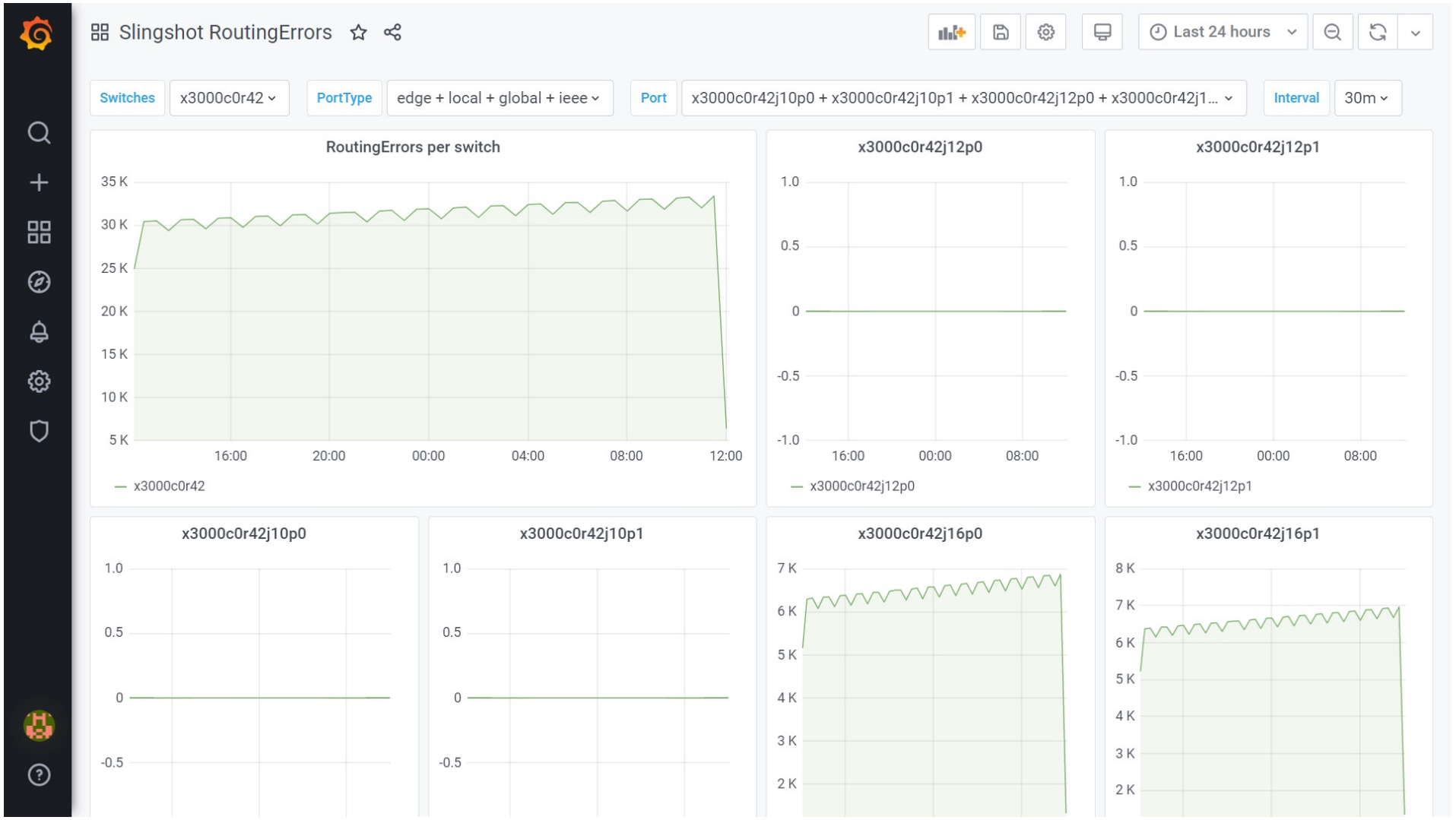
SMA-Grafana System monitoring dashboard



SMA-Grafana Slingshot congestion receive/transmit bandwidth



SMA-Grafana slingshot routing Errors



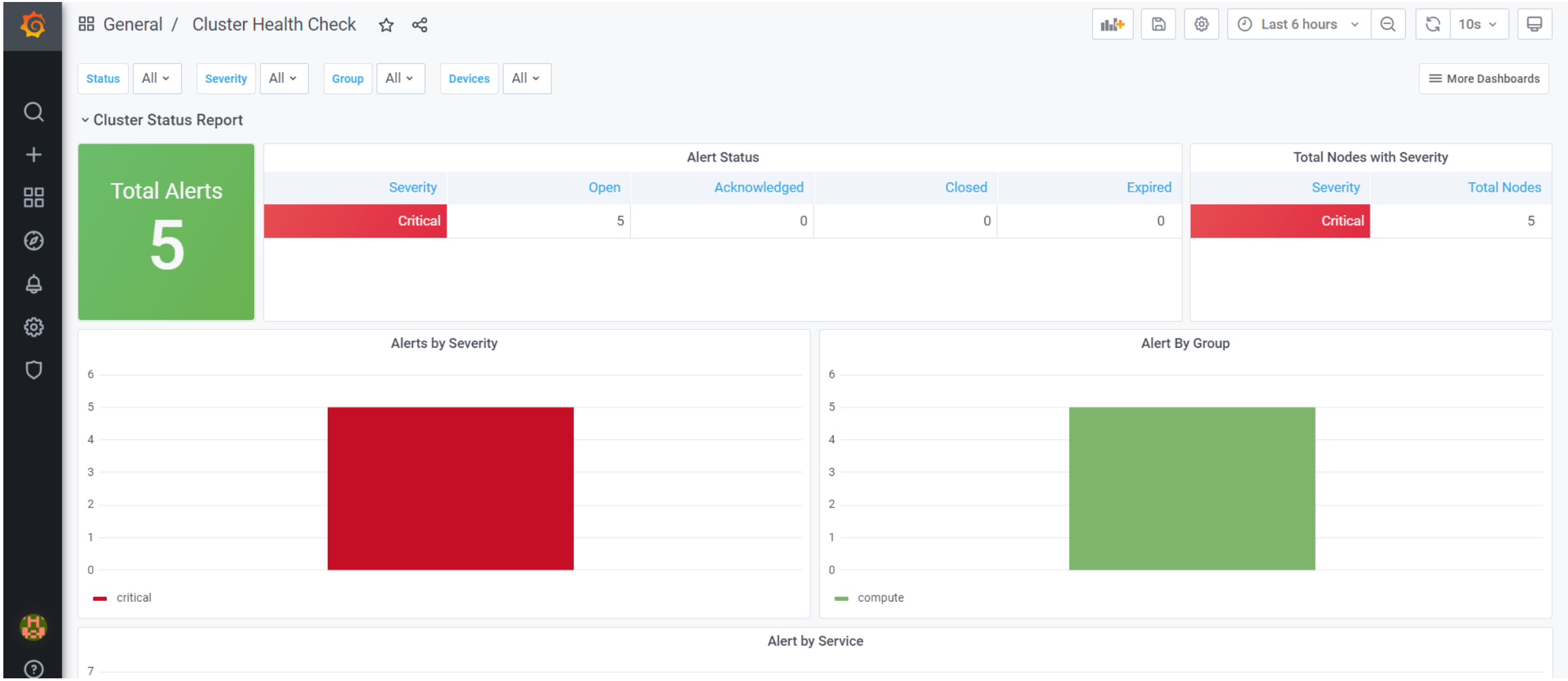
SMA-Grafana switch controller sensors

SMA-Grafana cabinet controller sensors

SMA-Grafana node controller sensors

CDU Monitoring

Cluster Health Check

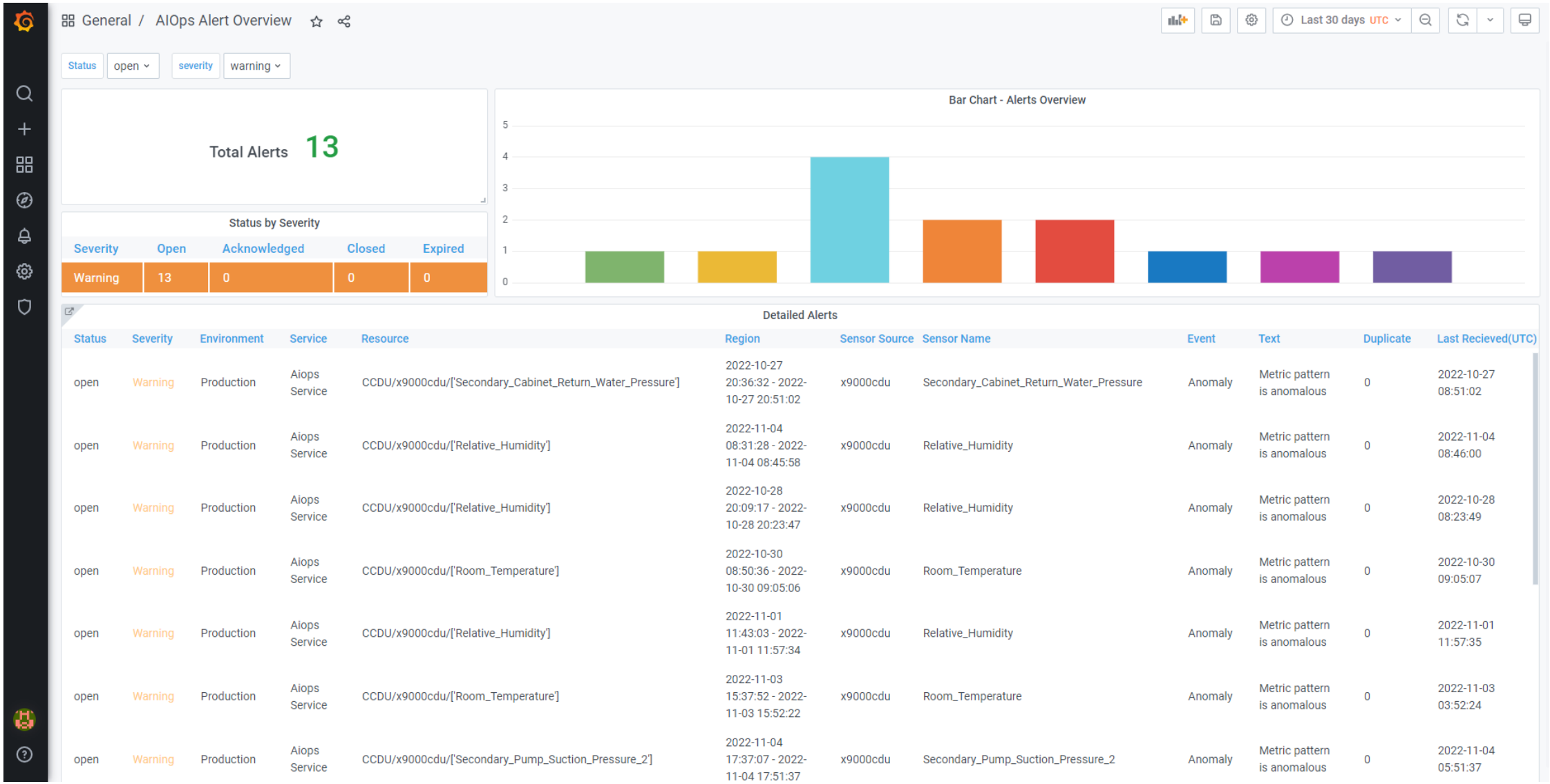


AIOps

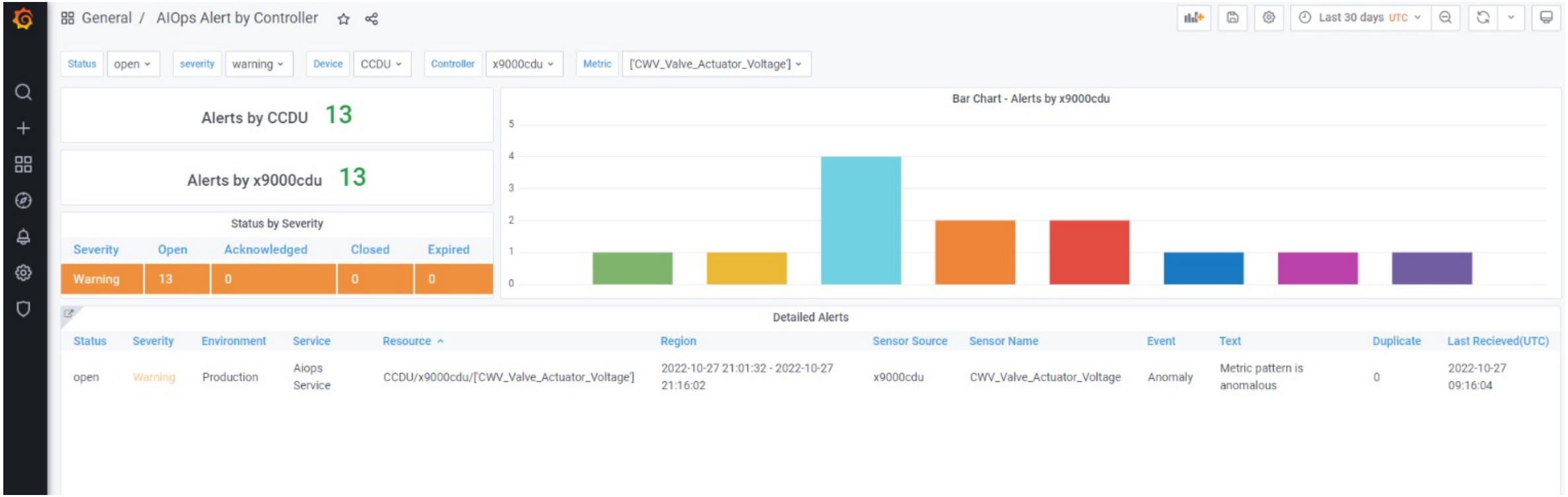
- Typical monitoring systems are based on thresholds
  - IT operations require administrators to monitor dashboards
  - The dashboards consolidate data from multiple monitoring systems based on established thresholds
- AIOps offers the following features:
  - Anomaly detection and processing
    - AIOps issues notifications for critical anomalies detected in the metrics derived from the cooling distribution units (CDUs)
    - AIOps simplifies data center management by reducing the number of false alarms, surfacing only anomalous results, limiting the number of dashboards needed, and providing other features
  - Default cooling device monitoring
    - Rather than rely on established thresholds, the default AIOps cooling device monitor uses dynamic thresholds for monitoring cooling devices
    - These dynamic thresholds are calculated automatically and are based on the latest data used to train the AI models
    - The data from the cooling systems can change over time for a number of reasons, and this approach makes alerting relevant to the latest data
  - Alert processing
    - You can display AIOps data in Grafana
    - Within Grafana, AIOps provides several dashboards in JSON format
- Enable AIOps monitoring for CDU forecast and Slingshot fabric by setting enabled=true in config map

```
ncn# kubectl edit cm -n sma aiops-enable-disable-models
```

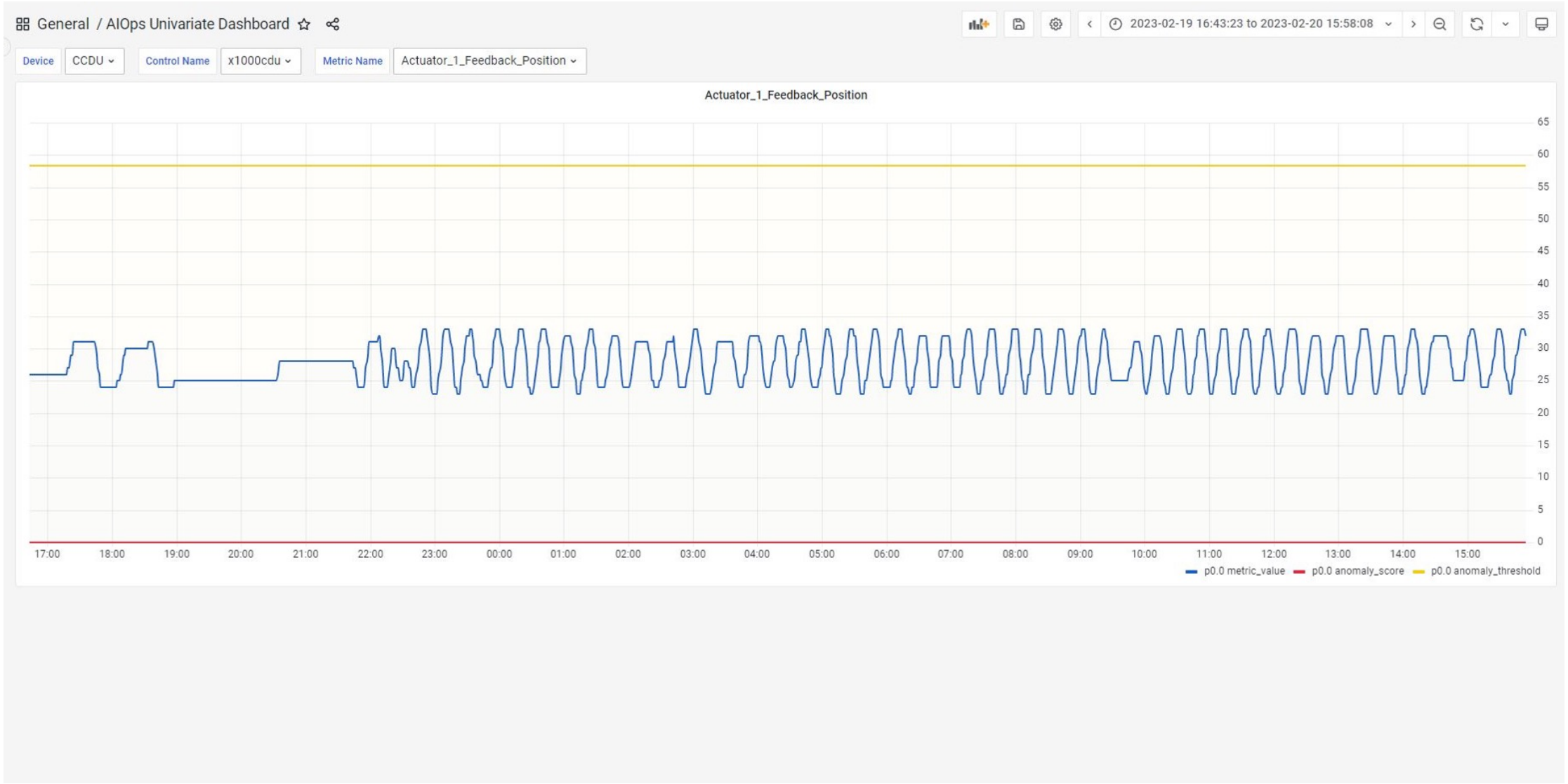
AIOps Alert Overview



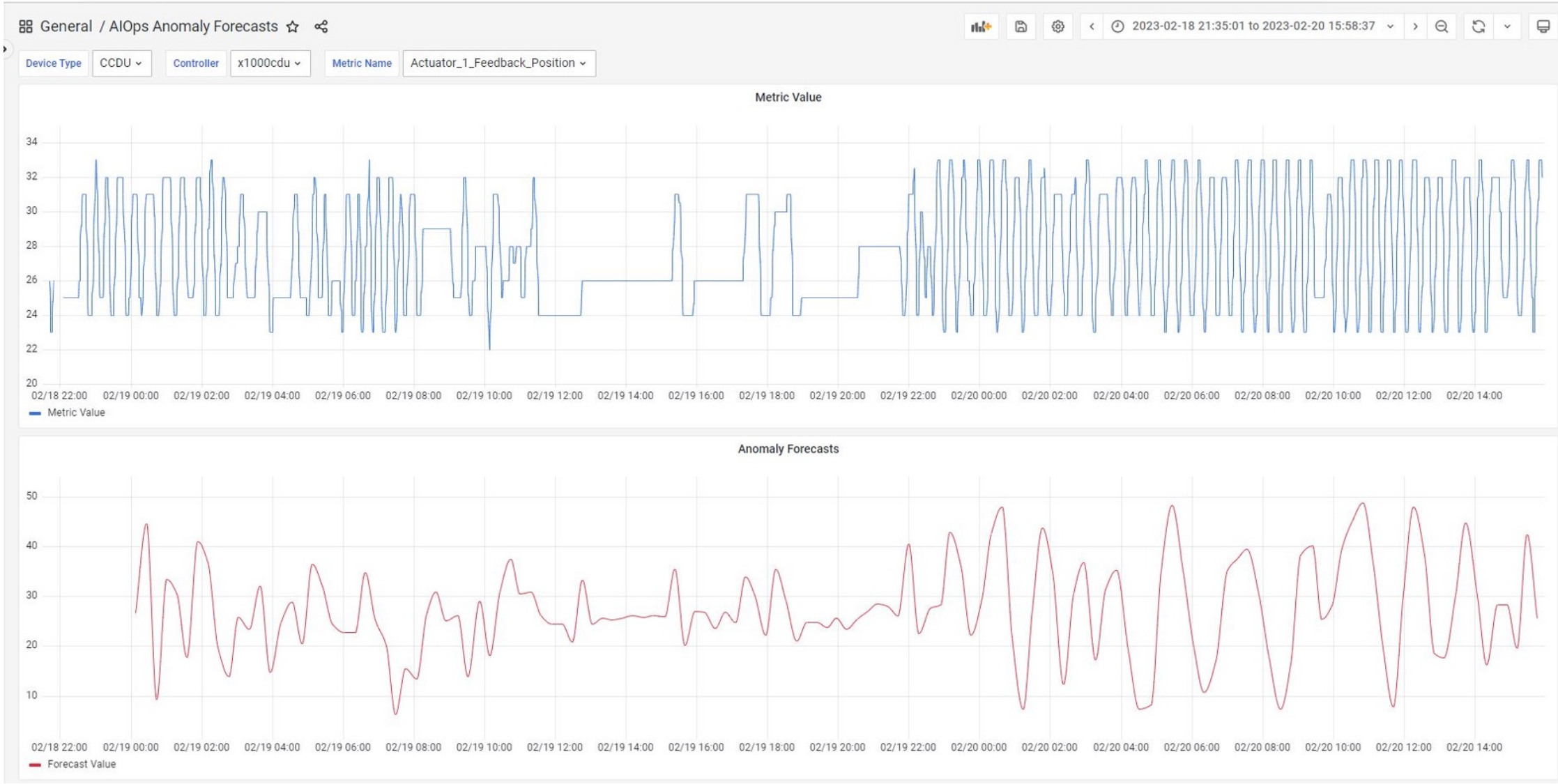
AIOps Alert by controller



AIOps univariate dashboard



AIOps anomaly forecast dashboard

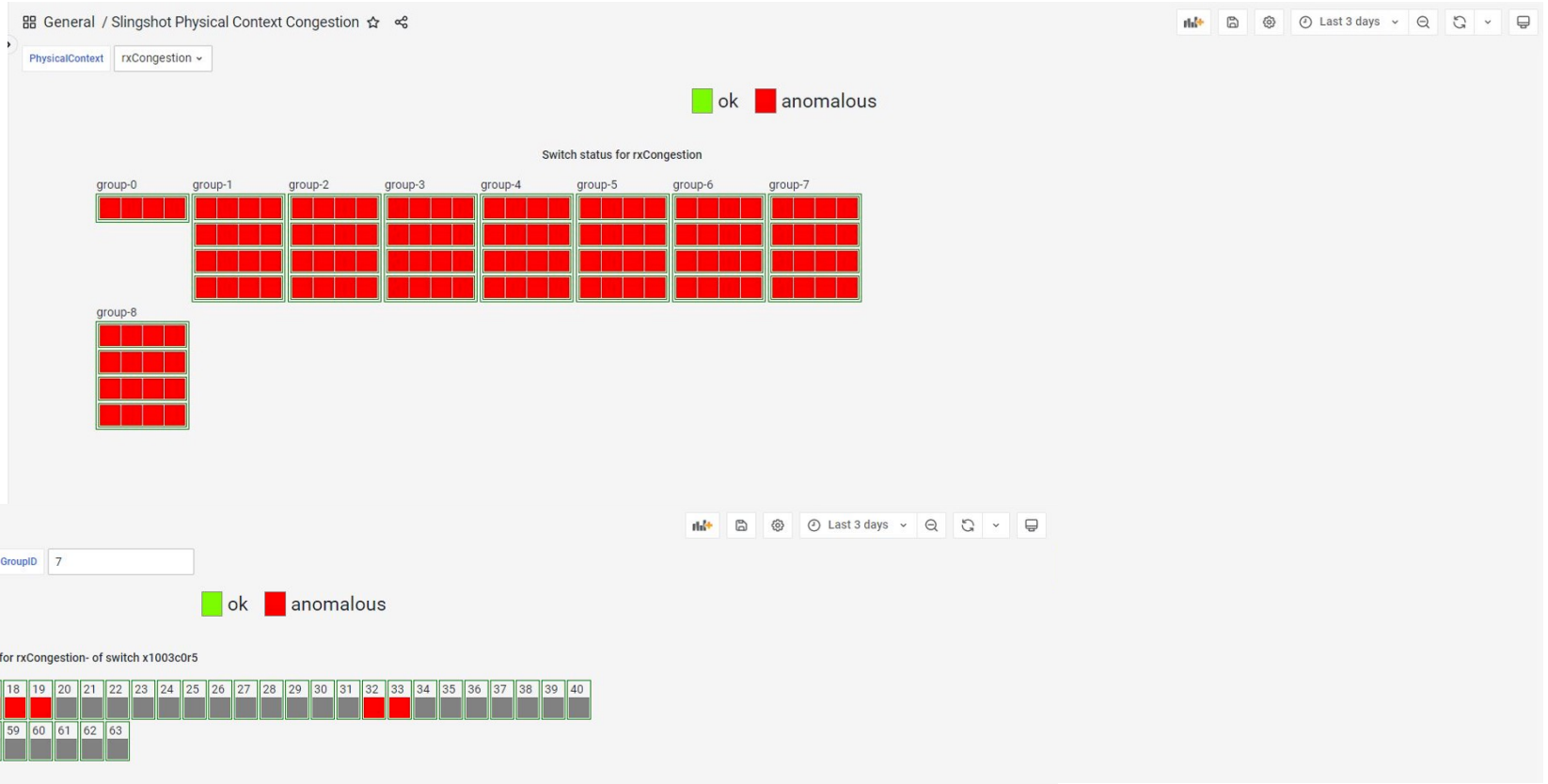


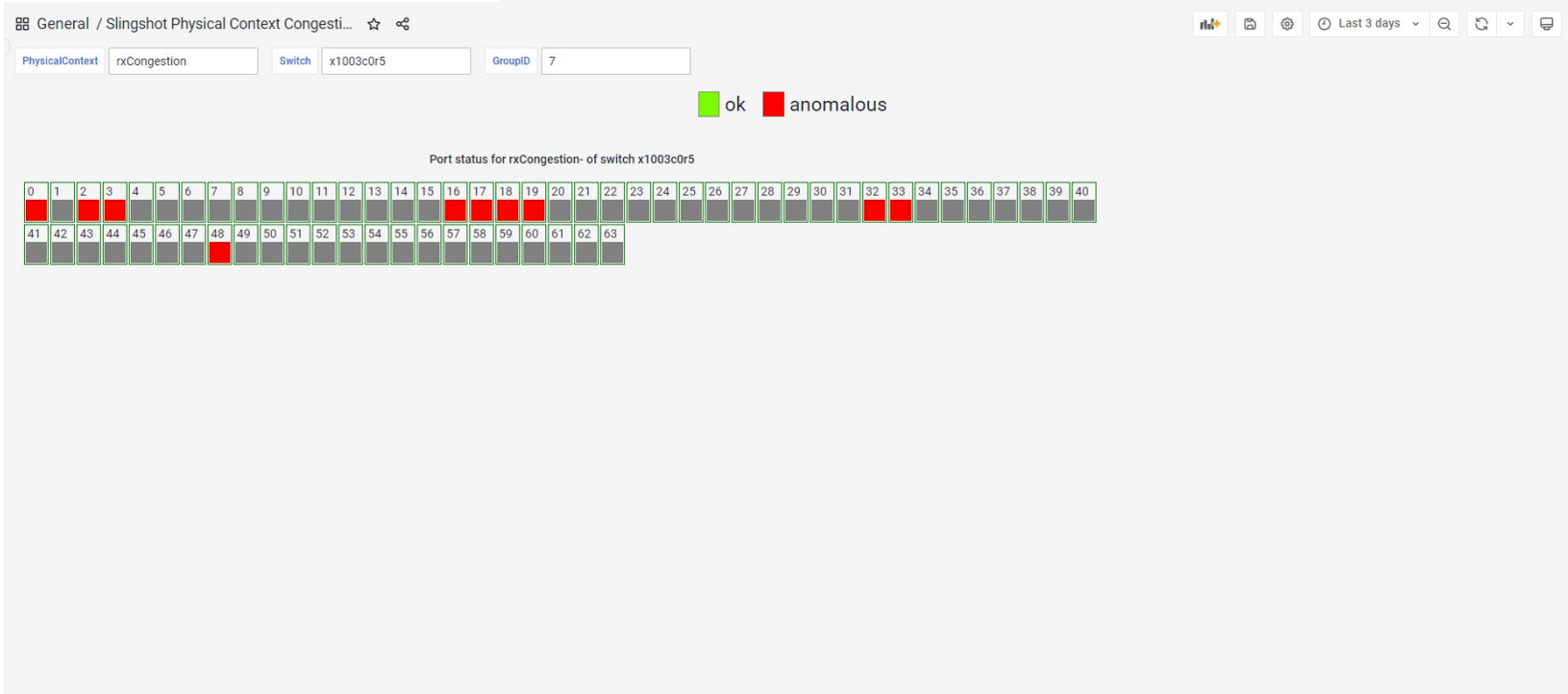
AIOps anomaly detection slingshot temperature

- Metrics used for cray-telemetry-temperature
  - ASIC
  - NetworkingDevice
  - SystemBoard
  - VoltageRegulator
  - Chassis
  - PowerSupply

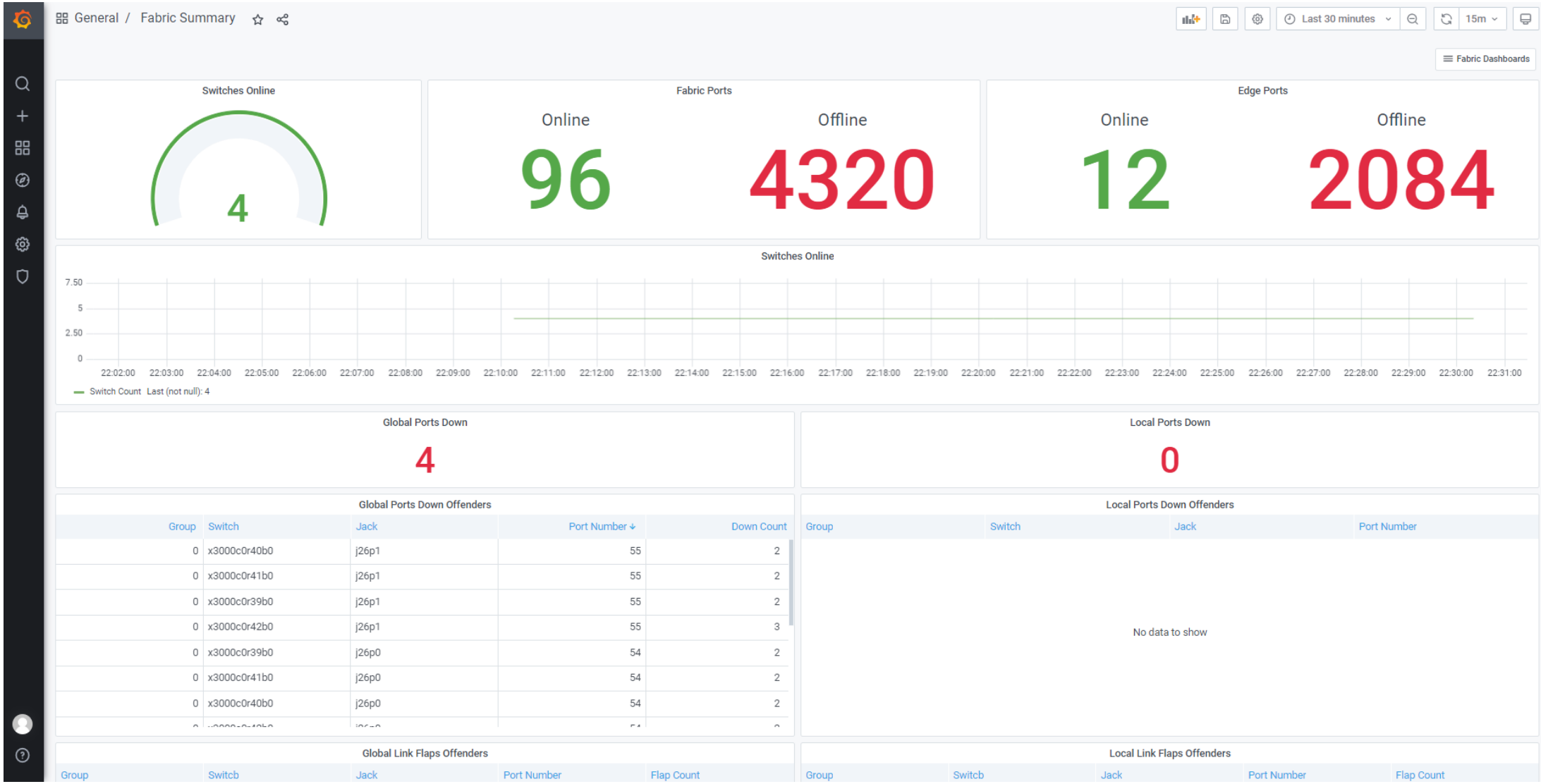
AIOps anomaly detection Slingshot congestion

- Metrics for cray-fabric-perf-telemetry
  - rxPausePercent
  - txPausePercent
  - rxCongestion

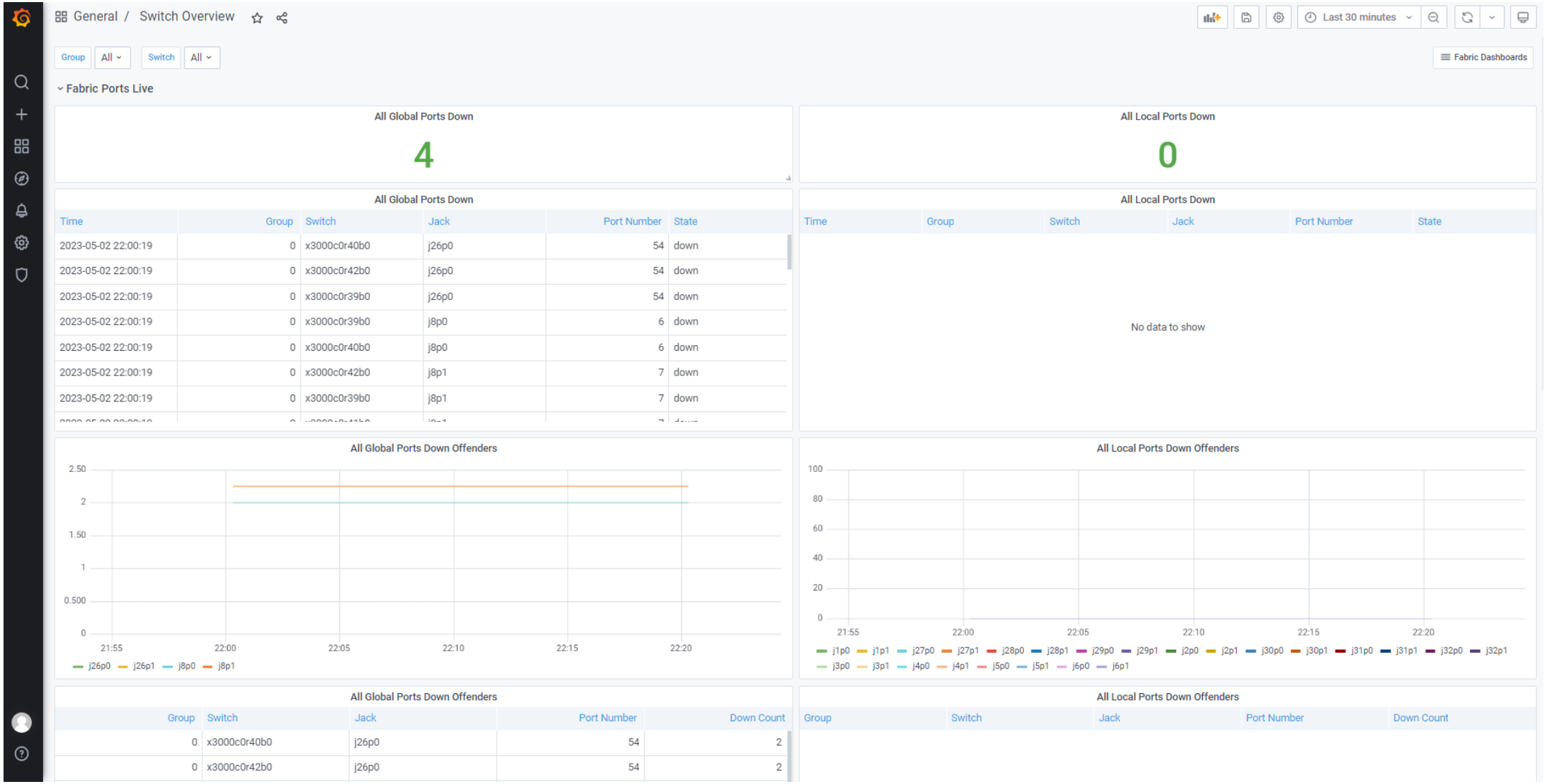




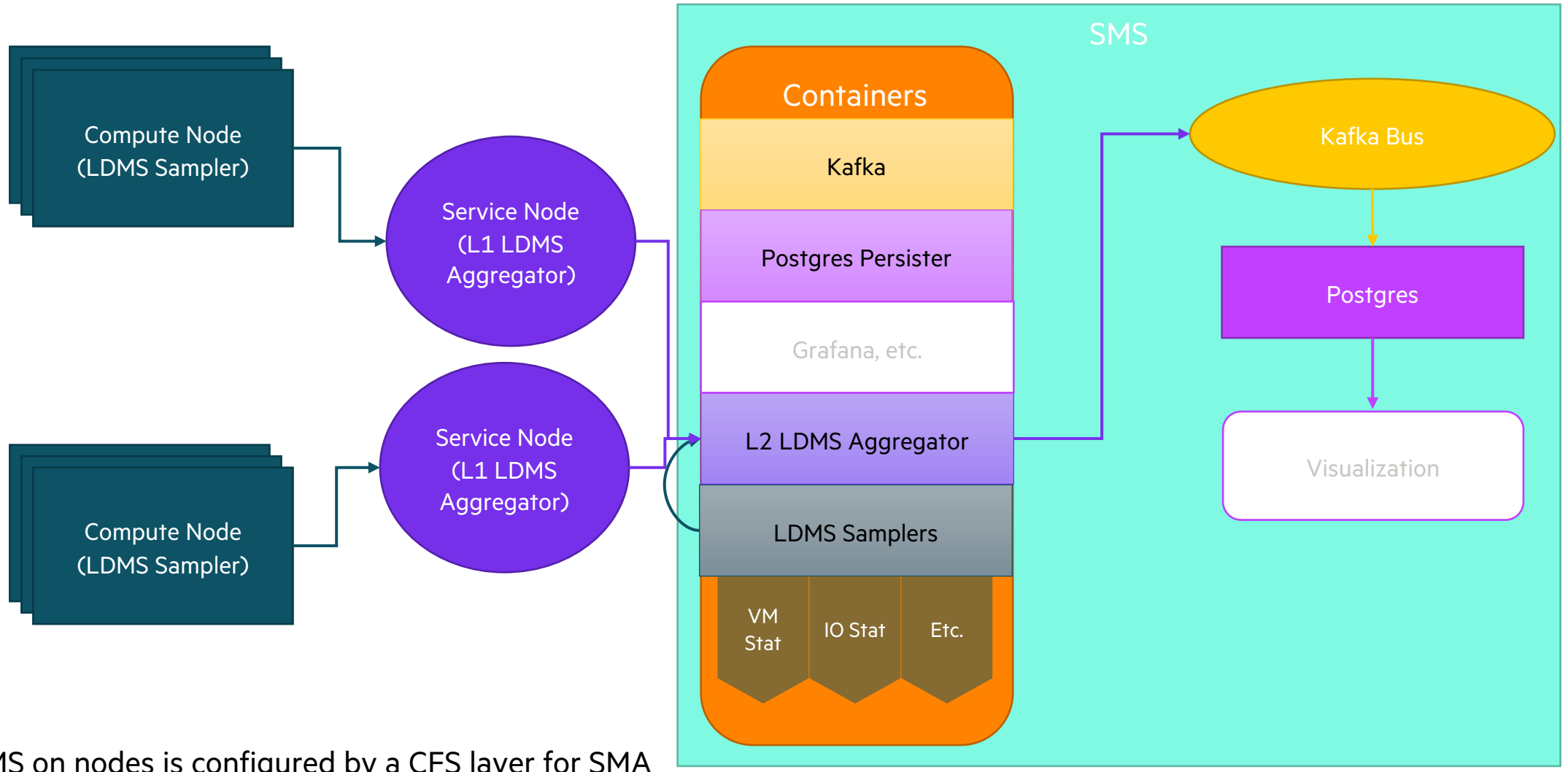
Slingshot Fabric summary



Slingshot switch overview



Lightweight Distributed Metric Service (LDMS)



- LDMS on nodes is configured by a CFS layer for SMA

Alerts

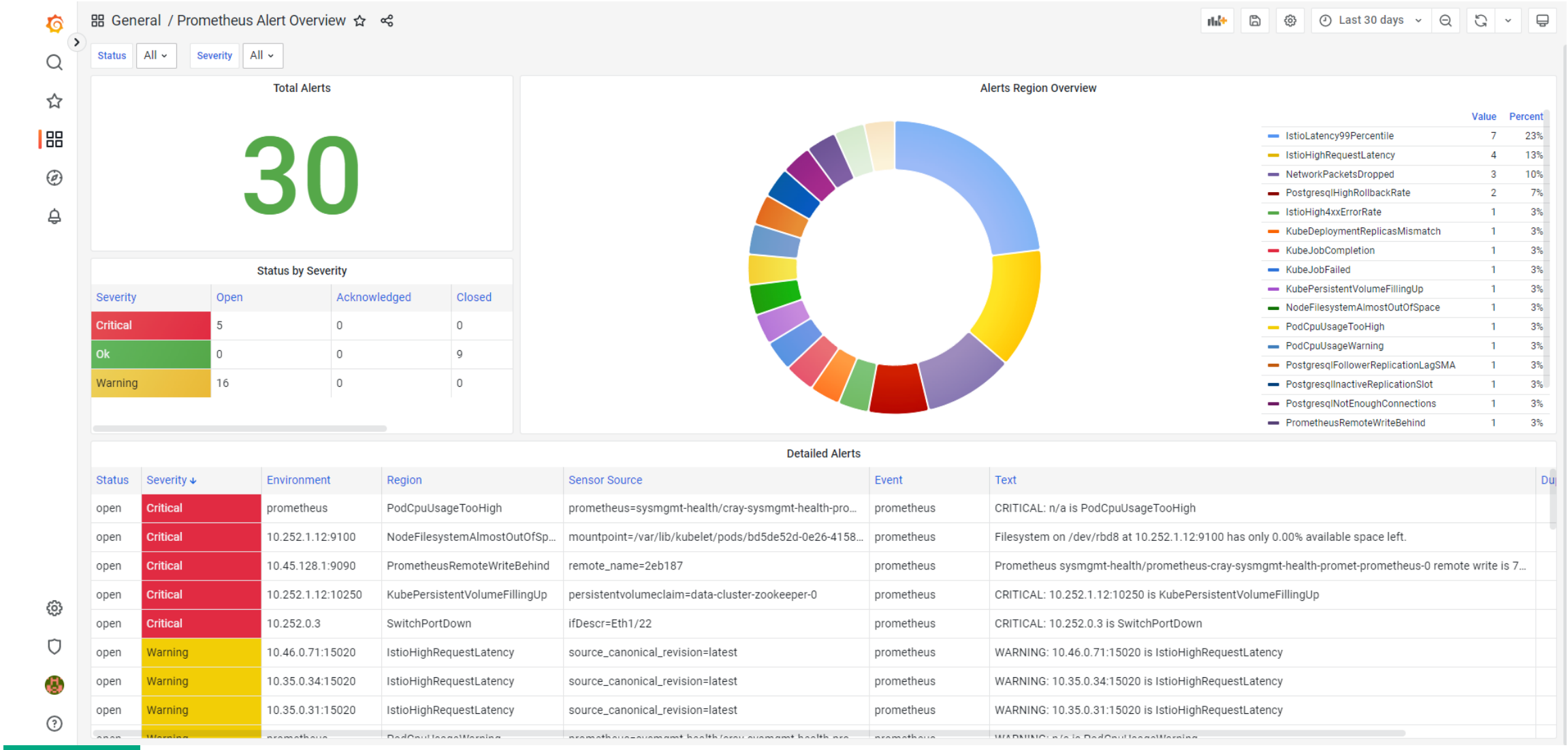
- Prometheus alerts
- Alertmanager
- Monasca alarms and notifications
- cm health alert
- sat sensors

Prometheus Health Checks

- Prometheus alerts provide coverage across infrastructure and platform
- Coarse-grained and comprehensive, as opposed to fine-grained and exhaustive
- Supports preventive and diagnostic use cases

NON-COMPUTE NODES	UTILITY STORAGE	CONTAINER ORCHESTRATION	SERVICE MESH	WORKLOADS
• CPU and memory utilization • Local storage utilization • Network I/O errors and latency • Clock skew	• Ceph status • Storage utilization • Disk I/O errors and latency	• Kubernetes status • API errors • CPU and memory overcommitments	• Istio status • Service availability • Service request rates • Service response statuses and latency	• Status of pods, deployments, stateful sets, daemon sets, jobs • CPU, memory, network, and storage utilization and errors

System Mgmt Health Grafana: Prometheus Alerts Overview



Retrieving Alerts from Prometheus

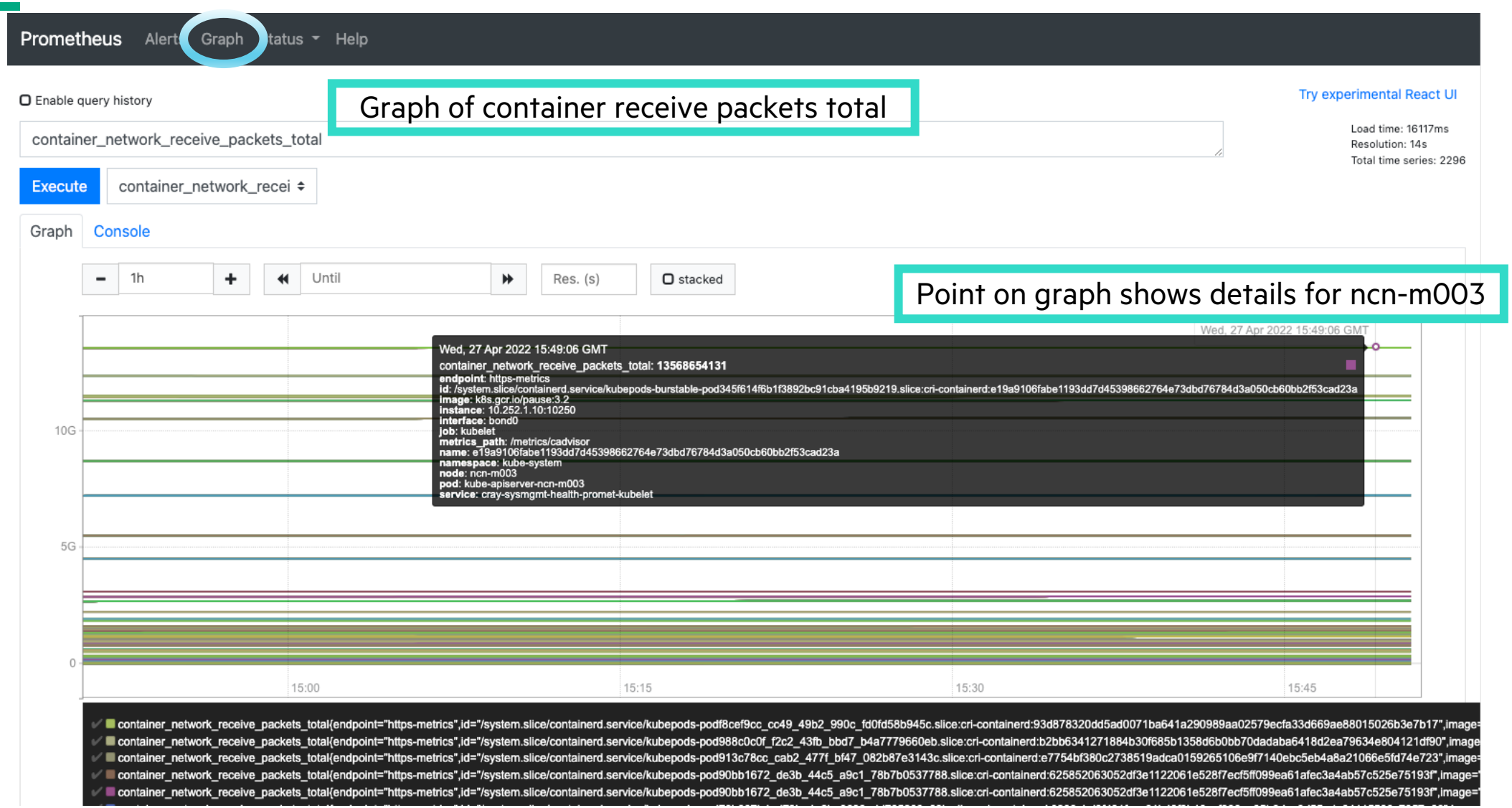
- CSM 1.4 and earlier

```
ncn# kubectl -n sysmgmt-health get svc cray-sysmgmt-health-promet-prometheus
```

- CSM 1.5

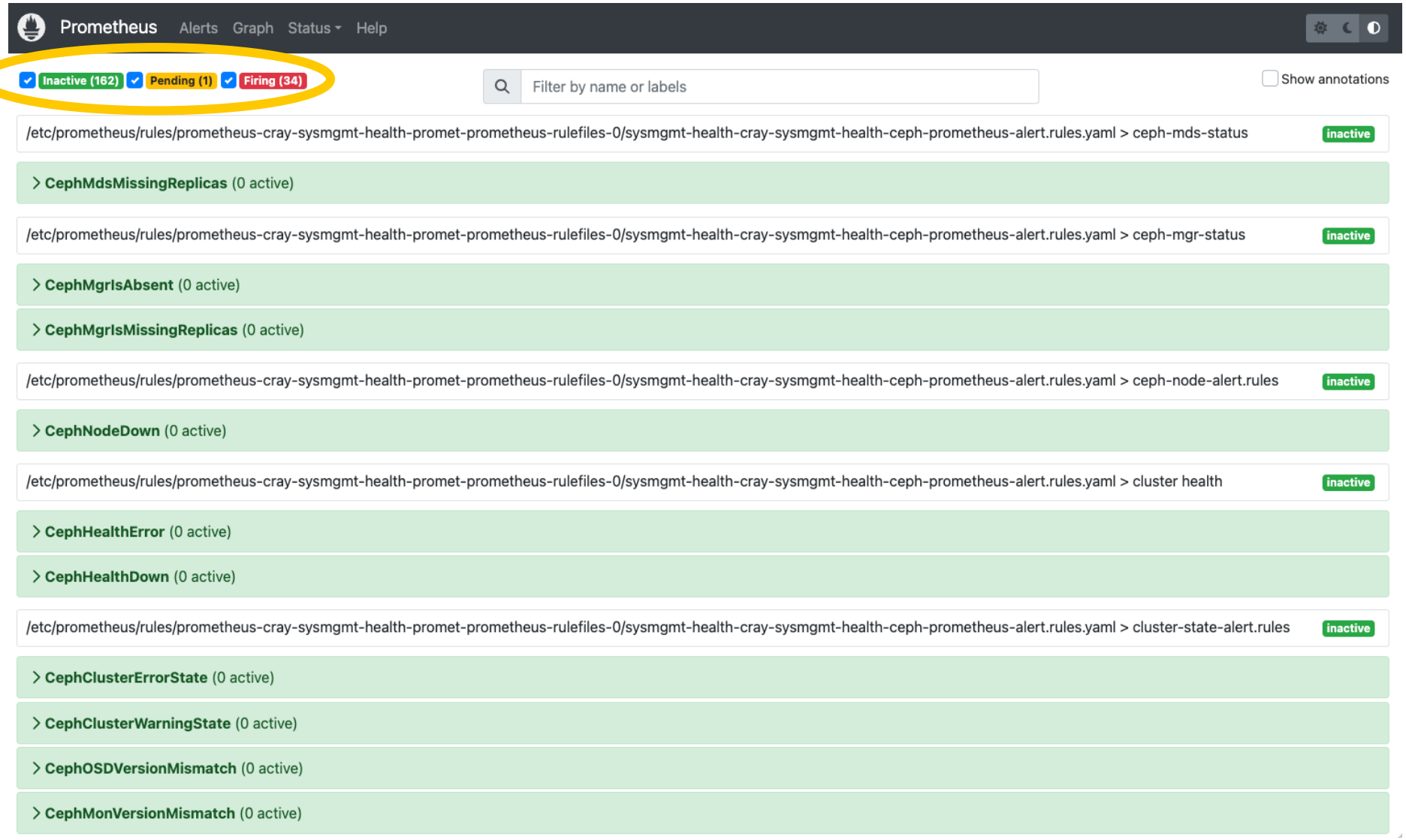
```
ncn# kubectl -n sysmgmt-health get svc cray-sysmgmt-health-kube-p-prometheus
NAME                                      TYPE        CLUSTER-IP     EXTERNAL-IP   PORT(S)    AGE
cray-sysmgmt-health-kube-p-prometheus   ClusterIP   10.21.141.187   <none>        9090/TCP   34d
ncn# curl -s http://10.21.141.187:9090/api/v1/alerts |jq -j '.data' | grep alertname | sort | uniq -c
     12          "alertname": "CPUThrottlingHigh",
      1          "alertname": "etcdInsufficientMembers",
      1          "alertname": "etcdMembersDown",
    108          "alertname": "IstioHighRequestLatency",
    103          "alertname": "IstioLatency99Percentile",
      1          "alertname": "IstioLowTotalRequestRate",
      1          "alertname": "KubeAPIErrorBudgetBurn",
      1          "alertname": "KubeDeploymentReplicasMismatch",
    131          "alertname": "KubeJobCompletion",
    130          "alertname": "KubeJobFailed",
      2          "alertname": "KubePersistentVolumeFillingUp",
      1          "alertname": "KubePodCrashLooping",
      1          "alertname": "NodeClockNotSynchronising",
      1          "alertname": "NodeMemoryUsageWarning",
      6          "alertname": "PodLivenessProbeFailure",
      1          "alertname": "PodReadinessProbeFailure",
      1          "alertname": "PostgresqlFollowerReplicationLagSMA",
      2          "alertname": "PostgresqlHighRollbackRate",
      1          "alertname": "PostgresqlInactiveReplicationSlot",
      3          "alertname": "PostgresqlNotEnoughConnections",
     14          "alertname": "SmartExpired",
      1          "alertname": "SMARTMonitorNotRunning",
      3          "alertname": "TargetDown",
      1          "alertname": "Watchdog",
```

Prometheus - graph



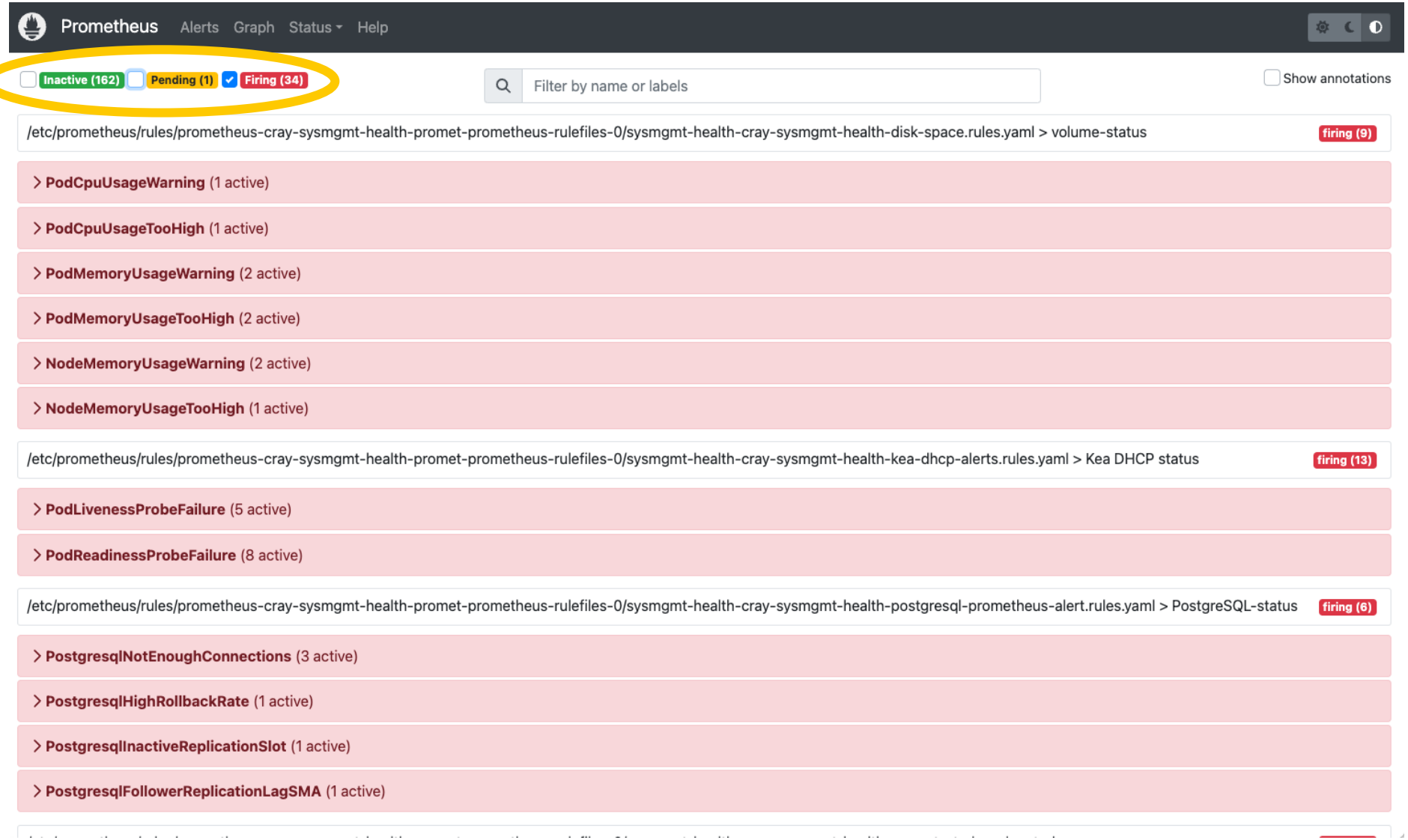
https://prometheus.cmn.SYSTEM_DOMAIN_NAME

Prometheus – alerts (all)



https://prometheus.cmn.SYSTEM_DOMAIN_NAME

Prometheus – alerts (only Firing)



https://prometheus.cmn.SYSTEM_DOMAIN_NAME

Prometheus – alerts (PostgresqlFollowerReplicationLagSMA)

Prometheus Alerts Graph Status Help

MdRaidDegradedOlderNodeExporter (0 active)

MdRaidDiskFailure (0 active)

/etc/prometheus/rules/prometheus-cray-sysmgmt-health-promet-prometheus-rulefiles-0/sysmgmt-health-cray-sysmgmt-health-postgresql-prometheus-alert.rules.yaml > PostgreSQL-status

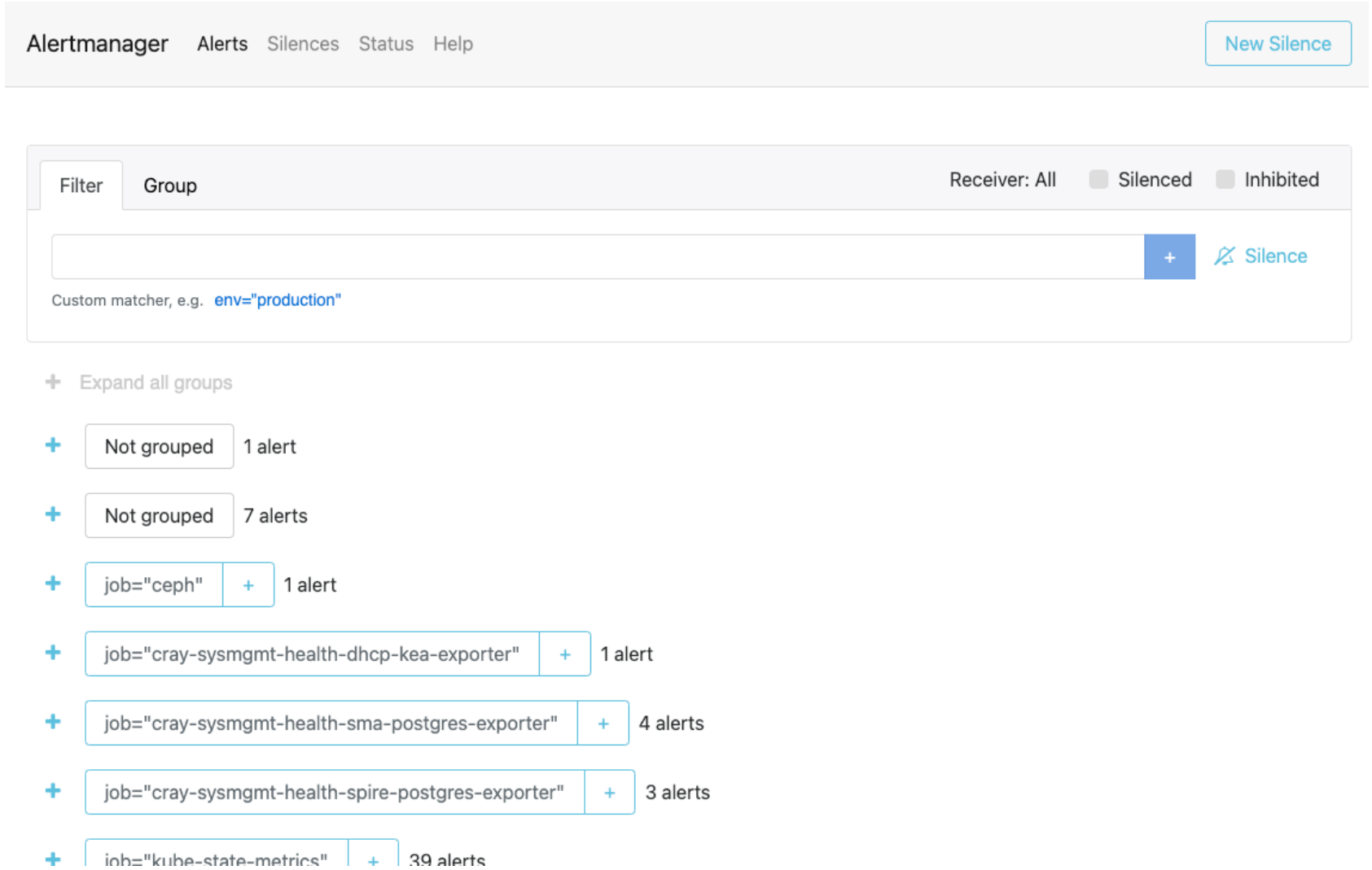
PostgresqlFollowerReplicationLagSMA (2 active)

```
alert: PostgresqlFollowerReplicationLagSMA
expr: pg_replication_slots_pg_wal_lsn_diff{namespace="sma"}
  > 1e+09
for: 5m
labels:
  severity: warning
annotations:
  description: Replica from follower "{{ $labels.application_name }}" is lagging
    behind master "{{ $labels.pod }}" by "{{ $value }}" bytes.
  summary: Postgresql replication lag from follower on replica "{{ $labels.application_name
    }}"
```

Labels	State	Active Since	Value
alertname="PostgresqlFollowerReplicationLagSMA" endpoint="exporter" instance="10.45.1.112:9187" job="cray-sysmgmt-health-sma-postgres-exporter" namespace="sma" pod="sma-postgres-cluster-1" server="localhost:5432" service="cray-sysmgmt-health-sma-postgres-exporter" severity="warning" slot_name="permanent_physical_1"	FIRING	2022-04-22 20:07:12.869288317 +0000 UTC	1.01669652776e+11
alertname="PostgresqlFollowerReplicationLagSMA" endpoint="exporter" instance="10.45.1.112:9187" job="cray-sysmgmt-health-sma-postgres-exporter" namespace="sma" pod="sma-postgres-cluster-1" server="localhost:5432" service="cray-sysmgmt-health-sma-postgres-exporter" severity="warning" slot_name="sma_postgres_cluster_0"	FIRING	2022-04-22 20:07:12.869288317 +0000 UTC	1.01669652776e+11

https://prometheus.cmn.SYSTEM_DOMAIN_NAME

Alertmanager



https://alertmanager.cmn.SYSTEM_DOMAIN_NAME

Monasca email notifications

- SMA monitors metric data that is transmitted on the main telemetry bus
  - Provides a way to notify users when select metric data is outside of normal operating values
  - Includes several pre-defined alarms
- SMA configmap `sma-monasca-alarms-configdata-cm`
  - email_destination
  - sendmail_server
  - email_source

```
ncn# kubectl -n sma edit cm sma-monasca-alarms-configdata-cm
ncn# kubectl -n sma describe cm sma-monasca-alarms-configdata-cm
```

- Changes require deleting pods and job and creating new job

```
ncn# kubectl -n sma delete pod -l component=notification
ncn# kubectl -n sma delete job -l component=alarms-init-job
ncn# kubectl -n sma delete pod -l component=alarms-init-job
ncn# vi alarms-init-job.yaml
ncn# kubectl -n sma apply -f alarms-init-job.yaml
```

Monasca local alarms

- Local alarms can be created that send email notifications
  - Create local alarm definitions

    ```
    ncn# vi customer-alarms-configmap.yaml
    ```
  - Deploy configmap

    ```
    ncn-# kubectl -n sma apply -f customer-alarms-configmap.yaml
    ```
  - Create job definition

    ```
    ncn# vi customer-alarms-init-job.yaml
    ```
  - Execute the SMA alarm initialization job

    ```
    ncn# kubectl -n sma apply -f customer-alarms-init-job.yaml
    ```
  - Verify job succeeds

    ```
    ncn# kubectl -n sma get po -l component=customer-alarms-init-job
    NAME READY STATUS RESTARTS AGE
    customer-alarms-init-job-dtrw5 0/1 Completed 0 5m
    ```

CLI for alarms

- List the state of all defined alarms

```
ncn# kubectl -n sma exec -it sma-monasca-agent-0 -c collector -- sh -c 'monasca alarm-list'
+--------------------------------------+--------------------------------------+-------------------+
| id                                   | alarm_definition_id                  | alarm_definition_name |
+--------------------------------------+--------------------------------------+-------------------+
| 0881af14-5659-4468-813a-d99ac7f415c5 | cd72e681-995c-4f0a-9d29-c6a0a0e0dde8 | validation1Alarm |
| 64bbb62d-3cb1-466c-bed3-e7012f742683 | cd72e681-995c-4f0a-9d29-c6a0a0e0dde8 | validation1Alarm |
| b571cb91-2e1f-486e-9dc7-8b1e112cb530 | cd72e681-995c-4f0a-9d29-c6a0a0e0dde8 | validation1Alarm |
+--------------------------------------+--------------------------------------+-------------------+
```

- List all defined alarms

```
ncn# kubectl -n sma exec -it sma-monasca-agent-0 -c collector -- sh -c 'monasca alarm-definition-list'
+-----------------------+--------------------------------------+-------------------------------------------------------------------------+
| name                  | id                                   | expression                                                              |
+-----------------------+--------------------------------------+-------------------------------------------------------------------------+
| validation1Alarm      | 791d8c5a-f217-456c-9223-53976dfc5cdf | last(kubelet.health_status) < 0                                         |
| metricsTestAlarm      | a8ae3ac5-de01-4019-b1bb-090faf9c8c51 | avg(cray_test.other_test) < 20                                          |
| validation2Alarm      | ccc9cbb1-ad79-49a6-94ff-30c465a4479b | last(monasca.thread_count) < 0                                          |
| Critical Redfish Event | d98b6394-28b6-4ff2-a0d3-214fe5dc636e | count(dmtf.redfish_event{severity=Critical}, deterministic) >= 1 |
| vmstatTestAlarm       | e11d3234-3bdb-4318-8d5b-3cee64affb7f | avg(cray_test.vmstat_test) < 20                                         |
+-----------------------+--------------------------------------+-------------------------------------------------------------------------+
```

- List all defined notifications

```
ncn# kubectl -n sma exec -it sma-monasca-agent-0 -c collector -- sh -c 'monasca notification-list'
+----------------+--------------------------------------+---------+-----------------------------------------------------------+
| name           | id                                   | type    | address                                                   |
+----------------+--------------------------------------+---------+-----------------------------------------------------------+
| defaultEmail   | 5f326486-5667-4afe-999f-1a65fe9ca7b0 | EMAIL   | email-distribution-list@customer.com                      |
| defaultWebhook | 8bb6cd0c-5674-42c4-aab1-cf1db78e164d | WEBHOOK | http://sma-alerta.sma.svc.cluster.local:8080/webhooks/monasca |
+----------------+--------------------------------------+---------+-----------------------------------------------------------+
```

Mon-alert (CSM 1.4 and earlier)

- Mon-alert manages the life cycle of each alert
  - Looks for events in the data
  - Analyzes each event
  - Alerts the user
  - Stores the event in the alert dashboard
  - Retrieve alerts, process alerts, and close alerts
  - Disable alert mechanism during maintenance

- Display a summary of all alerts

```
ncn# mon-alert -s
Alert Status Count
------------ -----
Critical     0
Warnings     6288
Information  1
Open         6288
Acknowledged 0
Closed       2
Expired      0
```

cm health alert (CSM 1.5)

```
ncn# cm health alert -s
Alert Status        Count
------------        -----
Critical            2
Warnings            0
Information         0

Open                2
Acknowledged        0
Closed              0
Expired             0

Group                Severity        Alerts
------               --------        ------
compute              critical        critical : 2, warning : 0, info : 0
fabric               ok              critical : 0, warning : 0, info : 0
slingshothsn         ok              critical : 0, warning : 0, info : 0
slingshotswitch      ok              critical : 0, warning : 0, info : 0
prometheus           ok              critical : 0, warning : 0, info : 0
aiops                ok              critical : 0, warning : 0, info : 0
```

- Manage alerts from many sources: Prometheus Alertmanager, Monasca, Slingshot
  - Looks for events in the data
  - Constantly analyzes each event
  - Alerts the user regarding the event
  - Stores the event in the alert dashboard
- Manage the life cycle of alerts
  - Retrieve alerts
  - Process alerts
  - Close alerts
  - Disable during maintenance periods and re-enable after maintenance ends

```
ncn# cm health alert query
ID        STATUS    SEVERITY    GROUP    ENV           SERVICE      RESOURCE            EVENT          VALUE  DESCRIPTION                                  DUPL  LAST RECEIVED
--------  --------  ----------  -------  ------------  -----------  ------------------  ------------  -------  -------------------------------------------  ----  -------------------
2809d804  open      critical    compute  x3700c0r41b0  SensorEvent  dmtf.redfish_event  PSU1-Voltage        0  Sensor _PSU1 Voltage_ reading of 0 _V_ is  0     2024/03/05 19:13:21
                                                                                                              below the 11.16 lower critical threshold.
85082bef  open      critical    compute  x3700c0r39b0  SensorEvent  dmtf.redfish_event  PSU1-Voltage        0  Sensor _PSU1 Voltage_ reading of 0 _V_ is  0.    2024/03/05 19:17:35
                                                                                                              below the 11.16 lower critical threshold.
```

Check sensors

- Obtain sensor readings from BMCs (ChassisBMC, NodeBMC, RouterBMC)
  - Limit the telemetry topics queried to the topics listed
  - The default is to query all topics:
    - cray-telemetry-temperature, cray-telemetry-voltage, cray-telemetry-power, cray-telemetry-energy, cray-telemetry-fan, cray-telemetry-pressure

```
ncn-m# sat sensors -x x1003c2s6b1 -t NodeBMC -b 2 --timeout 10 --topic cray-telemetry-temperature
Telemetry data being collected for x1003c2s6b1
Please be patient...
Waiting for metrics for all requested xnames from cray-telemetry-temperature.
Receiving metrics from stream: cray-telemetry-temperature...
Telemetry data received from cray-telemetry-temperature for all requested xnames.
+------------+---------+---------------------------+-------------------------------+-------------+----------+-----------+----------+------+
| xname      | Type    | Topic                     | Timestamp             | Location | Parental Context | Physical Context | Index |Value |
+------------+---------+---------------------------+-------------------------------+-------------+----------+-----------+----------+------+
| x1003c2s6b1 | NodeBMC | cray-telemetry-temperature | 2022-04-01T18:17:57.079525696Z | x1003c2s6b1n0 | Chassis | VoltageRegulator | 0 | 55.4 |
| x1003c2s6b1 | NodeBMC | cray-telemetry-temperature | 2022-04-01T18:17:56.585058025Z | x1003c2s6b1n0 | Chassis | VoltageRegulator | 2 | 45.8 |
| x1003c2s6b1 | NodeBMC | cray-telemetry-temperature | 2022-04-01T18:17:57.081500532Z | x1003c2s6b1n1 | Chassis | VoltageRegulator | 0 | 51.2 |
| x1003c2s6b1 | NodeBMC | cray-telemetry-temperature | 2022-04-01T18:17:56.580577726Z | x1003c2s6b1n1 | Chassis | VoltageRegulator | 2 | 45.8 |
| x1003c2s6b1 | NodeBMC | cray-telemetry-temperature | 2022-04-01T18:17:57.072975044Z | x1003c2s6b1n0 | MISSING | CPU       | 0   | 30.875000 |
| x1003c2s6b1 | NodeBMC | cray-telemetry-temperature | 2022-04-01T18:17:57.072913765Z | x1003c2s6b1n0 | MISSING | CPU       | 1   | 26.500000 |
| x1003c2s6b1 | NodeBMC | cray-telemetry-temperature | 2022-04-01T18:17:57.073033042Z | x1003c2s6b1n1 | MISSING | CPU       | 0   | 29.750000 |
| x1003c2s6b1 | NodeBMC | cray-telemetry-temperature | 2022-04-01T18:17:57.073074561Z | x1003c2s6b1n1 | MISSING | CPU       | 1   | 27.500000 |
+------------+---------+---------------------------+-------------------------------+-------------+----------+-----------+----------+------+
```

Logs

- Kubernetes logs
- Console logs
- SMA-OpenSearch

Kubernetes – View pod logs

- View the pod's log

```
ncn# kubectl -n NAMESPACE logs PODNAME
```

- Follow log continuously with the -f|–follow=true option

```
ncn# kubectl -n NAMESPACE logs PODNAME -f
```

- You may want to view the logs of more than one pod when there are multiple replicas of the same pod
  - Find the pods using labels

```
ncn# kubectl -n services get pods -l app.kubernetes.io/name=cray-bos
NAME                            READY   STATUS    RESTARTS   AGE
cray-bos-67c5f989f8-kb4rt       2/2     Running   0          18d
cray-bos-67c5f989f8-qf7ff       2/2     Running   0          18d
```

  - Watch their logs

```
ncn# kubectl -n services logs -l app.kubernetes.io/name=cray-bos
```

Containerized console access

- ConMan is a serial console management program designed to support a large number of console devices and simultaneous users
- cray-console uses ConMan for interactive remote console access and console log collection
  - Automatically detects nodes which have been added or removed
  - Shared filesystem in Ceph for all cray-console pods to easily view log data
  - Console log data sent to SMA for other log processing
  - Dynamic autoscaling number of cray-console-node pods for size of system
    - Minimally, two pods are started
    - The number of PODs is scaled on
      - 750 Liquid-cooled nodes and/or 2000 “River” nodes
        - The Liquid-cooled nodes each require an ssh connection, so numbers are different
- Log locations:
  - Logs visible in any `cray-console-node-x` pod
  - Node logs: `/var/log/conman/console.XNAME`
  - ConMan daemon logs: `/var/log/conman.log`

```
ncn# kubectl get pods -A |grep cray-console
services            cray-console-data-5cd59677d9-lf4f4
services            cray-console-data-postgres-0
services            cray-console-data-postgres-1
services            cray-console-data-postgres-2
services            cray-console-node-0
services            cray-console-node-1
services            cray-console-operator-7f9894f657-5psn5
```

Console logs with cray-console-node

```
ncn# kubectl get pods -A |grep console-node
services            cray-console-node-0        3/3     Running       1     62d
services            cray-console-node-1        3/3     Running       0     68d
ncn# kubectl -it exec -n services cray-console-node-1 -c cray-console-node -- ls /var/log/conman
console.x1000c0s1b0n0  console.x1000c3s3b0n0   console.x3000c0s20b4n0
console.x1000c0s1b0n1  console.x1000c3s3b0n1   console.x3000c0s23b1n0
console.x1000c0s1b1n0  console.x1000c3s3b1n0   console.x3000c0s23b2n0
console.x1000c0s1b1n1  console.x1000c3s3b1n1   console.x3000c0s23b3n0
console.x1000c0s5b0n0  console.x1000c5s5b0n0   console.x3000c0s23b4n0
console.x1000c0s5b0n1  console.x1000c5s5b0n1   console.x3000c0s25b1n0
console.x1000c0s5b1n0  console.x1000c5s5b1n0   console.x3000c0s25b2n0
console.x1000c0s5b1n1  console.x1000c5s5b1n1   console.x3000c0s25b3n0
console.x1000c0s7b0n0  console.x1000c7s7b0n0   console.x3000c0s25b4n0
```

Each pod sees all the console files, only one cray-console-node pod is managing that node and writing its log file

```
ncn# kubectl -it exec -n services cray-console-node-1 -c cray-console-node -- \
tail -f /var/log/conman/console.x1000c0s1b0n0
```

Can view log without entering pod

```
ncn# kubectl -it exec -n services cray-console-node-1 -c cray-console-node -- /bin/bash
cray-console-node-1-pod# grep -i error /var/log/conman/console.x1000c0s1b0n0
```

Can view log by entering pod

Interactive console example (long)

- To join the console, use `conman -j`
  - Retrieve the `cray-console-operator` pod ID

```
ncn# CONPOD=$(kubectl get pods -n services \
    -o wide|grep cray-console-operator|awk '{print $1}')
ncn# echo $CONPOD
cray-console-operator-79bf95964-qpcpp
```

  - Set the `XNAME` variable to the xname of the node whose console you wish to open

```
ncn# XNAME=x1000c0s0b0n0
```

  - Find the `cray-console-node` pod that is managing that node

```
ncn# NODEPOD=$(kubectl -n services exec $CONPOD -c cray-console-operator \
-- sh -c "/app/get-node $XNAME" | jq .podname | sed 's/"//g')
ncn# echo $NODEPOD
cray-console-node-1
```

  - Connect to the node's console using ConMan on the `cray-console-node` pod you found

```
ncn# kubectl exec -it -n services $NODEPOD -- conman -j $XNAME
<ConMan> Connection to console [x1000c0s0b0] opened.
nid000001 login:
```

- To exit console use `&.` command

Interactive console example (short)

- Bash function to join console

```
ncn# ConsoleJ ()
{
    XNAME=$@;
    CONPOD=$(kubectl get pods -n services -o wide|grep cray-console-operator|awk '{print $1}');
    NODEPOD=$(kubectl -n services -c cray-console-operator exec $CONPOD -- sh -c "/app/get-node $XNAME" | jq .podname | tr -d '"');
    echo conpod = $CONPOD nodepod = $NODEPOD;
    kubectl exec -it -n services $NODEPOD -c cray-console-node -- conman -j $XNAME
}
ncn# ConsoleJ x1000c0s0b0n0
<ConMan> Connection to console [x1000c0s0b0n0] opened.
nid000001 login:
```

- To exit console use `&.` command
- To view the console read-only instead of joining it read-write, use `conman –m $XNAME`

NodeBMC console logs

- In addition to console logs available via conman in cray-console-node pods, check the NodeBMC
  - Console logs for nodes on blades in (liquid-cooled) Olympus cabinets can be accessed from the nodeController (BMC)
    - For node x1000c0s0b1n0, connect to its nodeController

```
ncn# ssh x1000c0s0b1
x1000c0s0b1> cd /var/log/n0
x1000c0s0b1> tail current
x1000c0s0b1> grep –i error current
```

  - Console logs for nodes in (air-cooled) River cabinets can be accessed from the node BMC using a Web GUI to the BMC

SMA OpenSearch

- SMA OpenSearch enables
  - Viewing all logs from CNs, NCNs, and Kubernetes pods
  - Sorting and searching through log information from multiple sources to help troubleshoot issues
- View and analyze system logs in the web UI
- Access sma-dashboards
  1. Determine the external domain name by running the following command on any NCN:

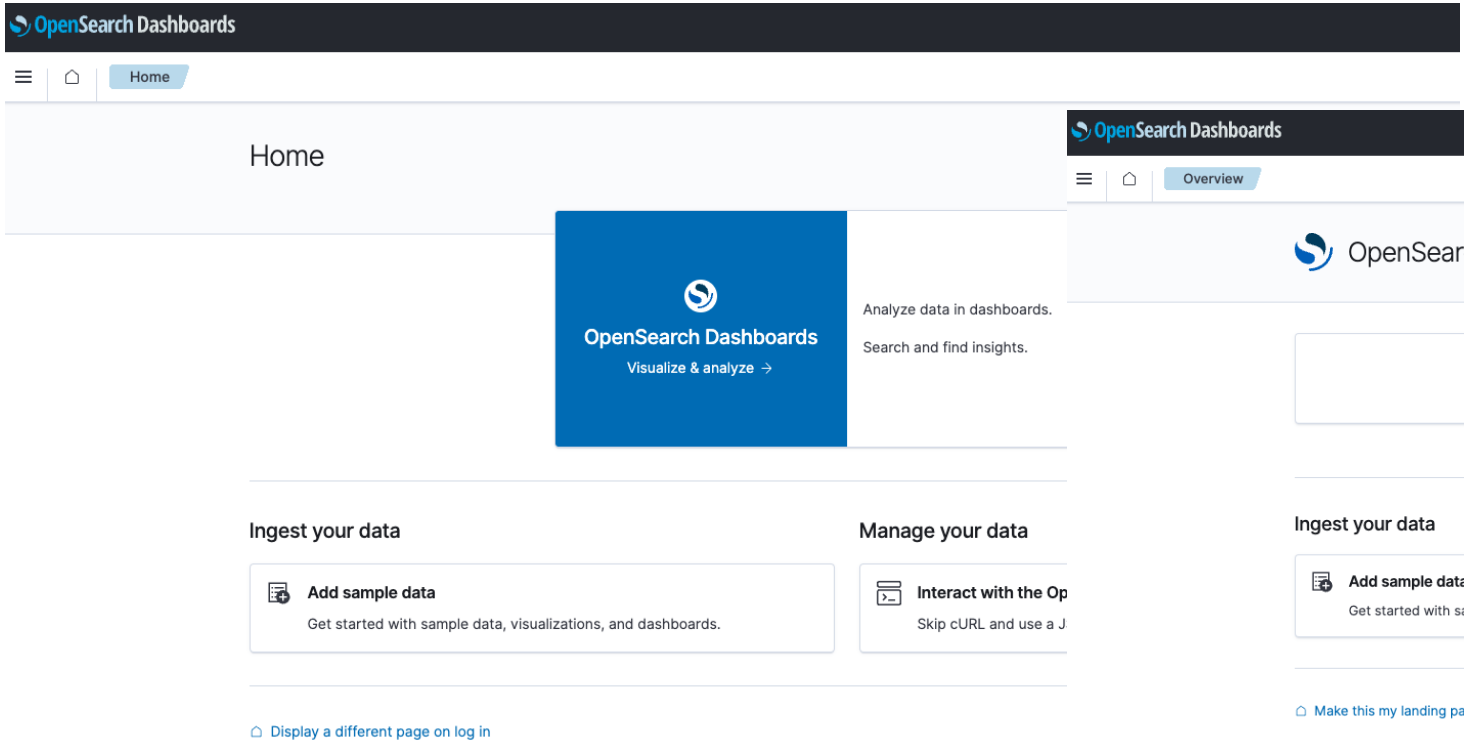
     ```
     ncn-m001# kubectl get secret site-init -n loftsman \
     -o jsonpath='{.data.customizations\.yaml}' | base64 -d | grep "external:"
     external: SYSTEM_DOMAIN_NAME
     ```

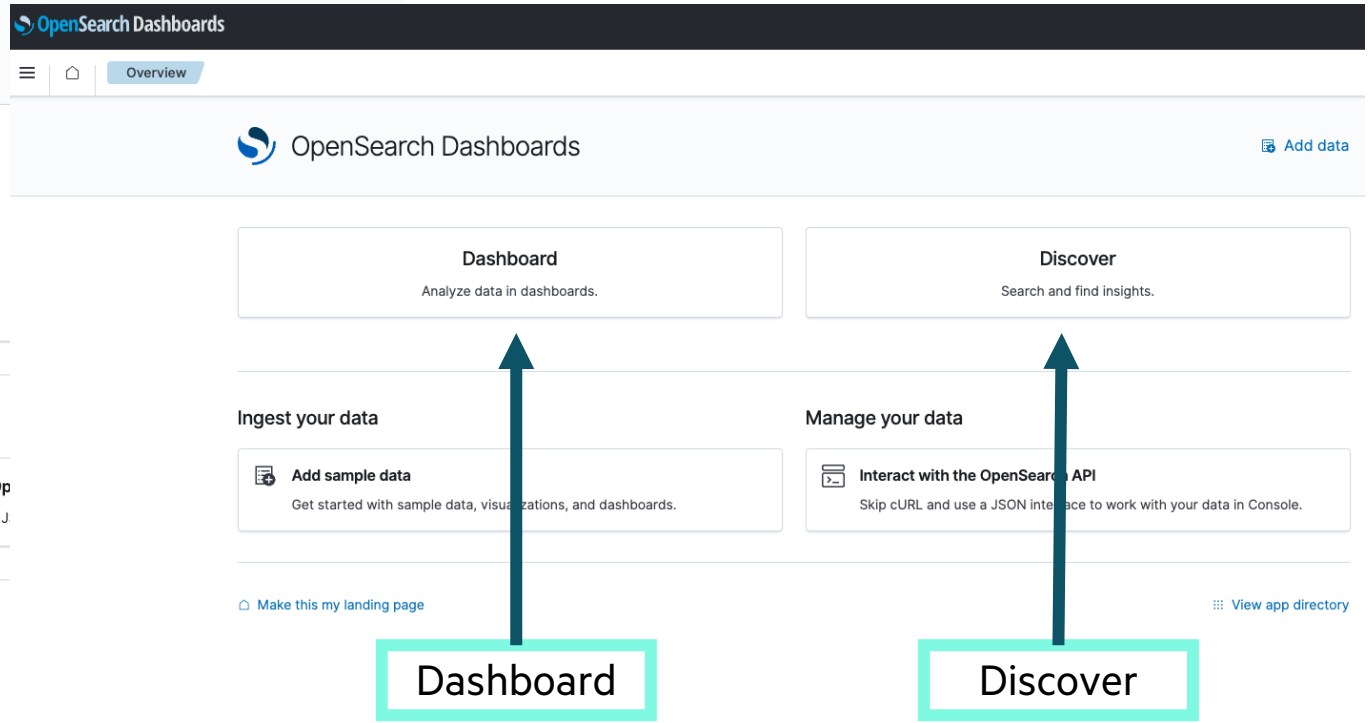
  2. Navigate to the following URL in a web browser:

     `https://sma-dashboards.cmn.SYSTEM_DOMAIN_NAME/`

  3. Login by entering a valid username and password
- See Opensearch documentation to further explore and analyze the system logs
  - https://opensearch.org/docs/latest/about/

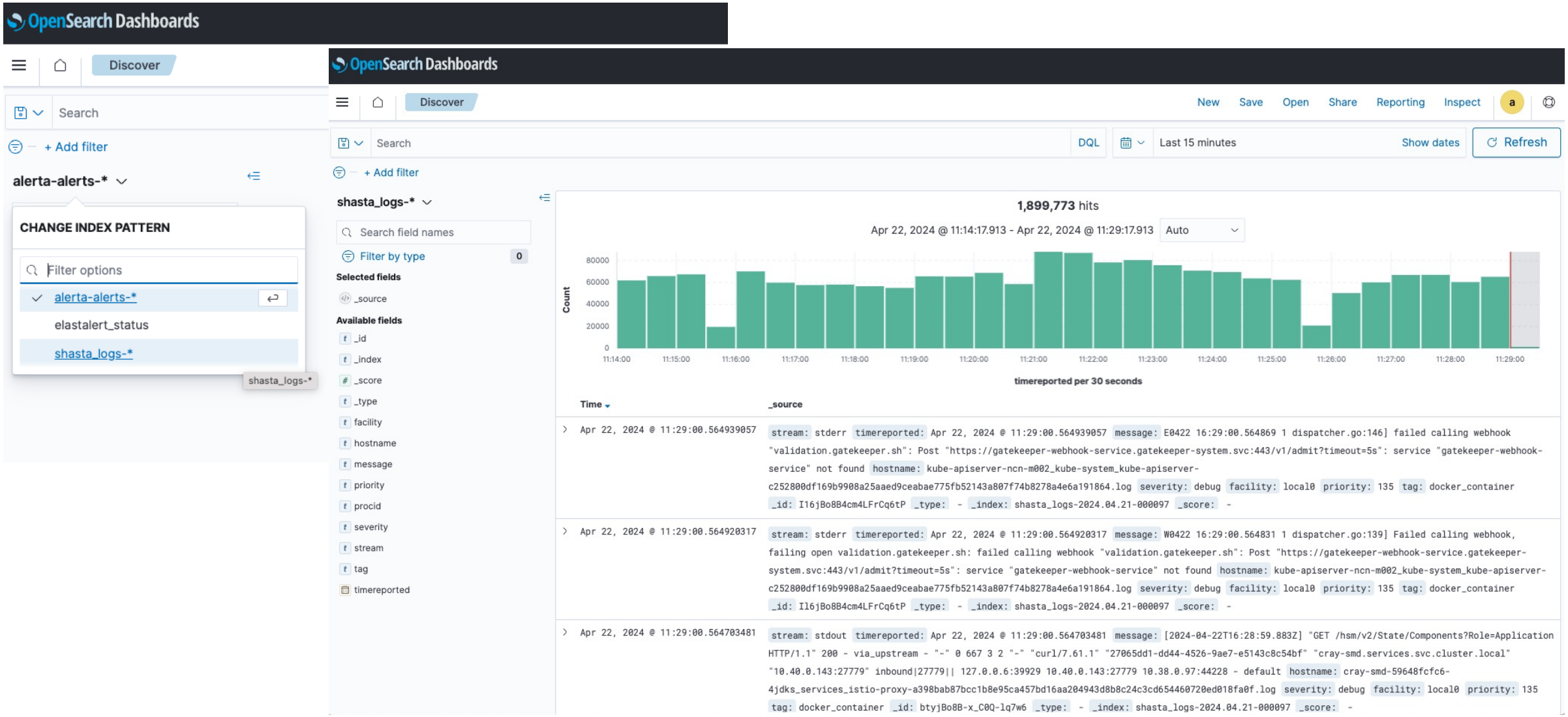
OpenSearch – sma-dashboards





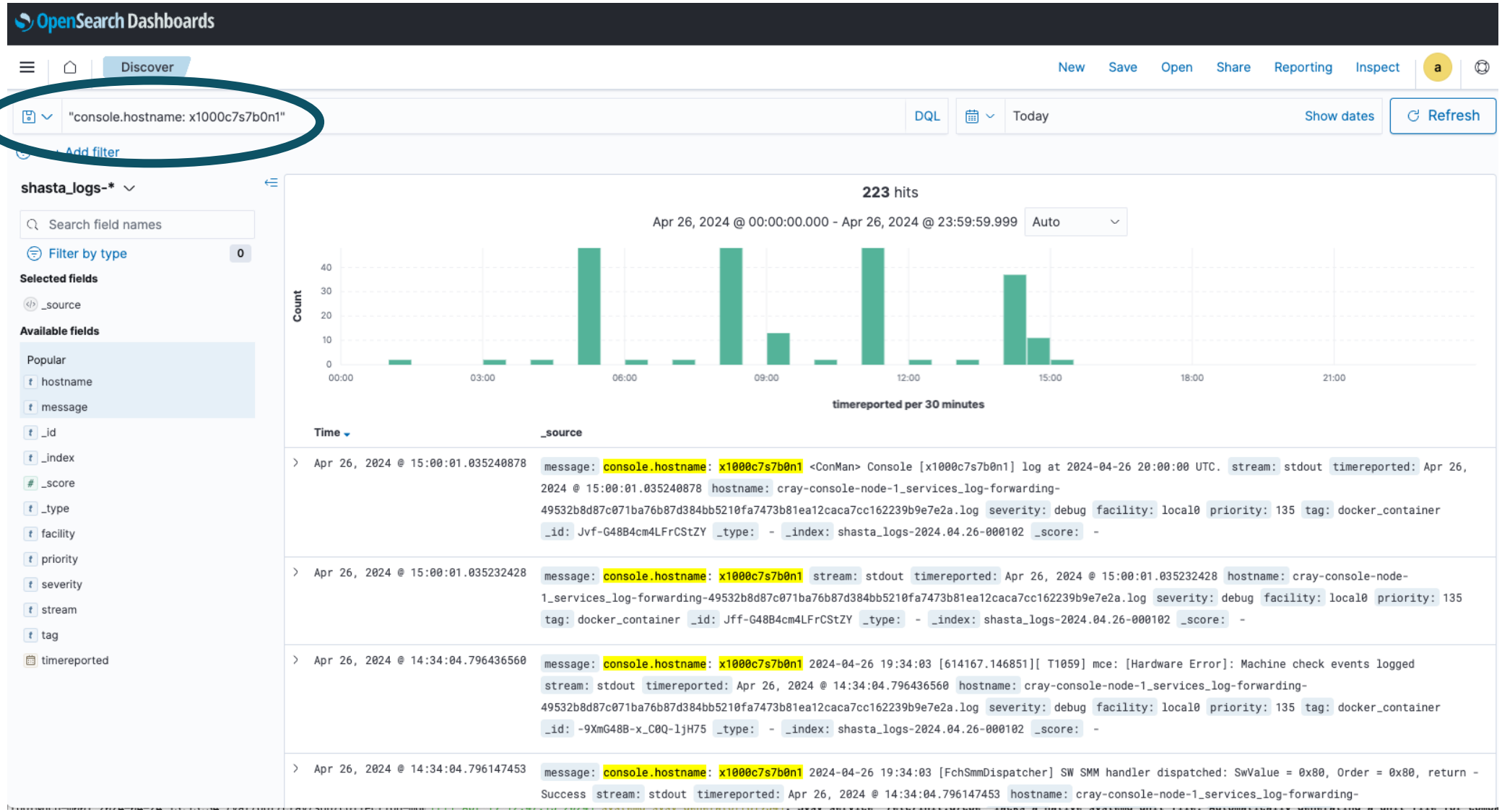
https://sma-dashboards.cmn.SYSTEM_DOMAIN_NAME

OpenSearch - Discover



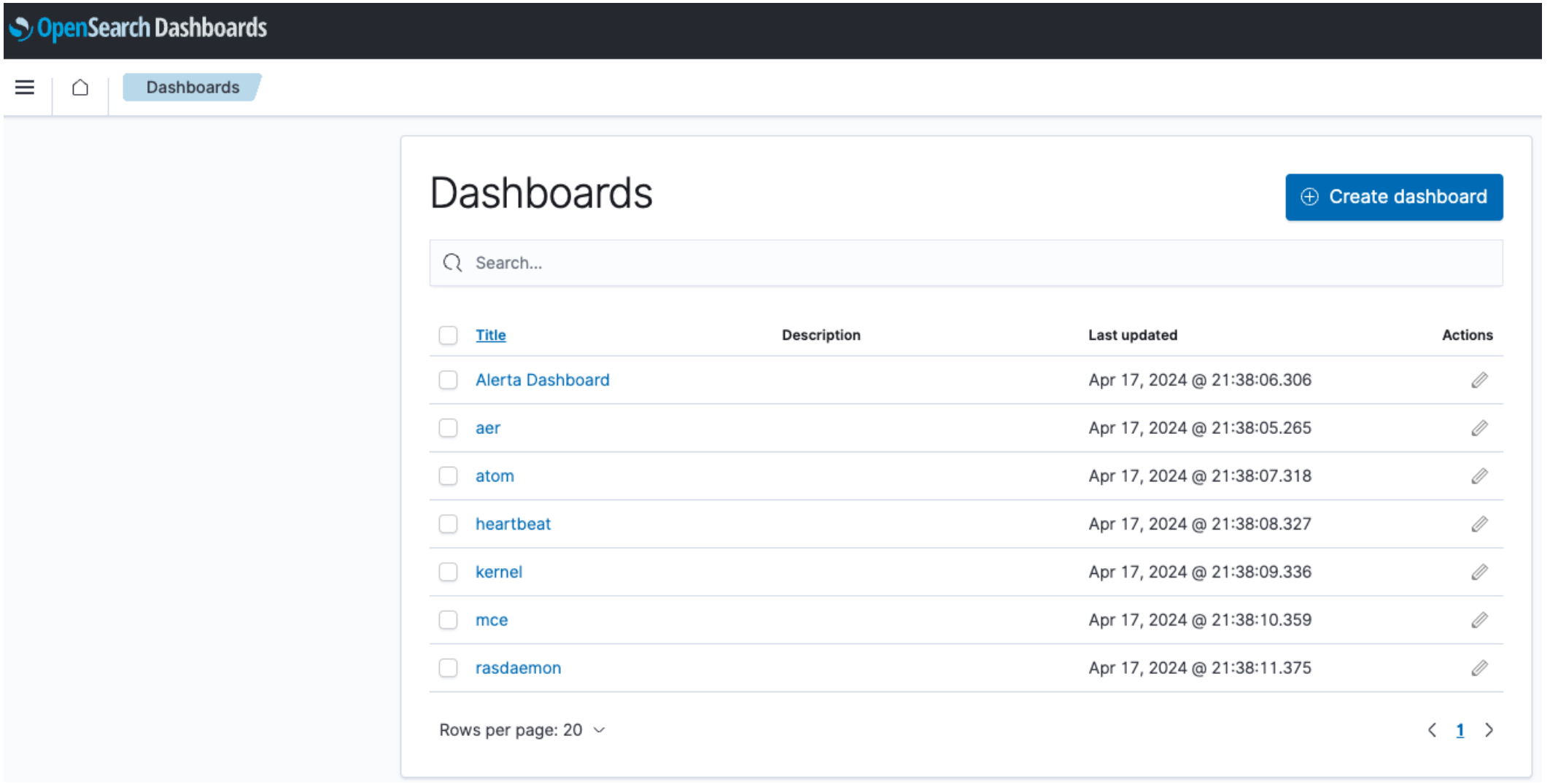
https://sma-dashboards.cmn.SYSTEM_DOMAIN_NAME

OpenSearch – Discover – Console Logs



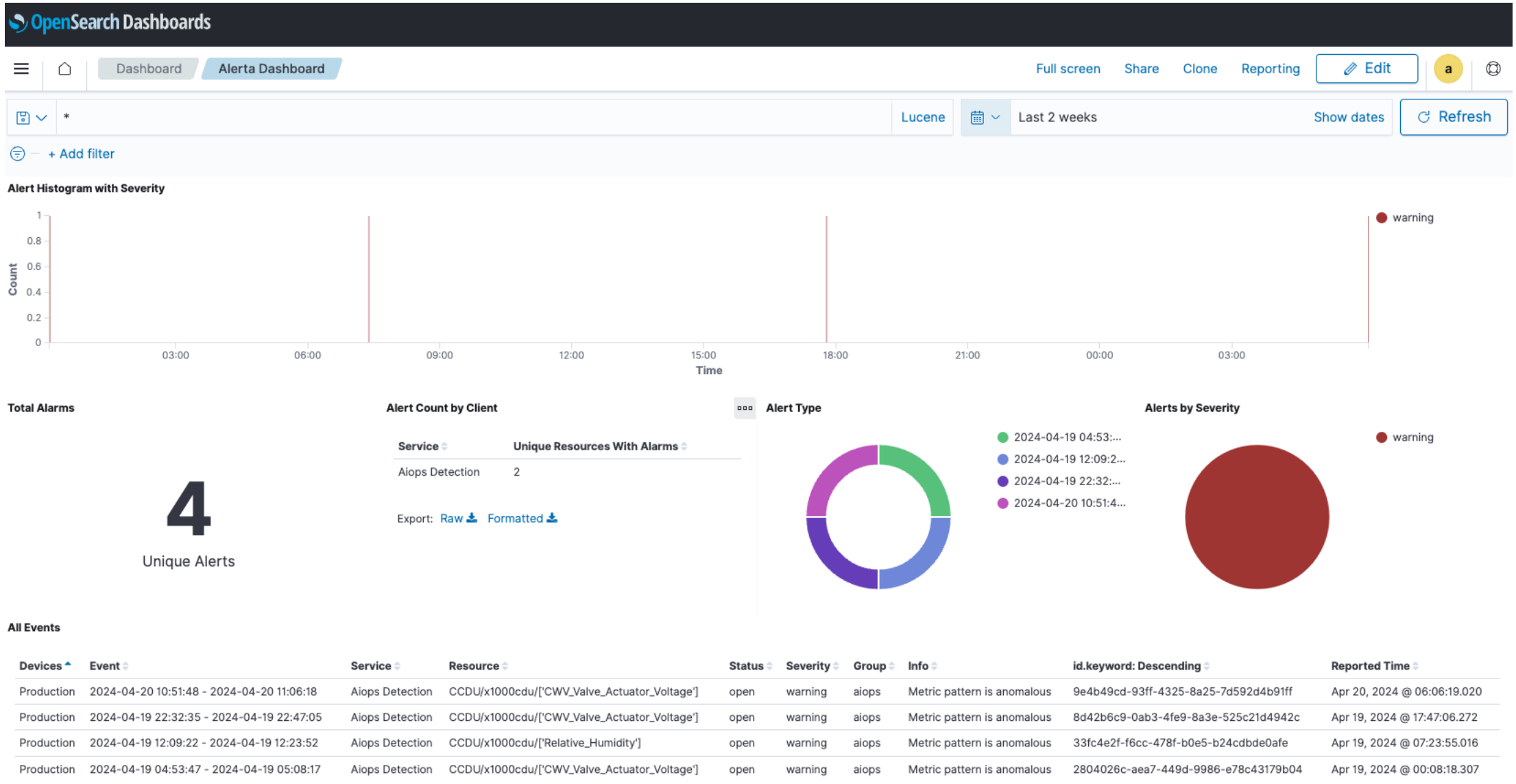
https://sma-dashboards.cmn.SYSTEM_DOMAIN_NAME

OpenSearch Dashboards



https://sma-dashboards.cmn.SYSTEM_DOMAIN_NAME

OpenSearch Dashboards - Alerta dashboard



https://sma-dashboards.cmn.SYSTEM_DOMAIN_NAME

OpenSearch Dashboards – AER, ATOM, heartbeat

Dashboard	Short description	Long description	Visualization and Search name
aer	AER corrected	Corrected Advanced Error Reporting messages from PCI Express devices on each node	aer-corrected
aer	AER fatal	Fatal Advanced Error Reporting messages from PCI Express devices on each node	aer-fatal
atom	ATOM failures	Application Task Orchestration and Management tests are run on a node when a job finishes. Test failures are logged	atom-failed
atom	ATOM admindown	Application Task Orchestration and Management test failures can result in nodes being marked admindown. An admindown node is not available for job launch	atom-admindown
heartbeat	Heartbeat loss events	Heartbeat loss event messages reported by the hbtd pods that monitor for heartbeats across nodes in the system	heartbeat

https://sma-dashboards.cmn.SYSTEM_DOMAIN_NAME

OpenSearch Dashboards - kernel

Dashboard	Short description	Long description	Visualization and Search name
kernel	Kernel assertions	The kernel software performs a failed assertion when some condition represents a serious fault so the node goes down	kassertions
kernel	Kernel panics	The kernel panics when something is seriously wrong so the node goes down	kernel-panic
kernel	Lustre bugs (LBUGs)	The Lustre software in the kernel stack performs a failed assertion when some condition related to file system logic represents a serious fault so the node goes down	lbug
kernel	CPU stalls	CPU stalls (Read-Copy-Update stalls where software in the kernel stack holds onto memory for too long) are serous conditions that can reduce node performance, and sometimes cause a node to go down. Read-Copy-Update is a vital aspect of kernel performance and rather esoteric	cpu-stall
kernel	Out of memory	An Out Of Memory (OOM) condition has occurred so the kernel must select an expendable process to kill to continue or if there is no expendable process the node usually goes down in some manner	oom

https://sma-dashboards.cmn.SYSTEM_DOMAIN_NAME

OpenSearch Dashboards – MCE and rasdaemon

Dashboard	Short description	Long description	Visualization and Search name
mce	MCE	Machine Check Exceptions (MCE) are errors detected at the processor level	mce
rasdaemon	rasdaemon errors	Errors from the rasdaemon service on nodes. The rasdaemon service is the Reliability, Availability, and Serviceability Daemon, and it is intended to collect all hardware error events reported by the Linux kernel, including PCI and MCE errors. This may include certain HSN errors in the future	rasdaemon-error
rasdaemon	rasdaemon messages	All messages from the rasdaemon service on nodes	rasdaemon

https://sma-dashboards.cmn.SYSTEM_DOMAIN_NAME

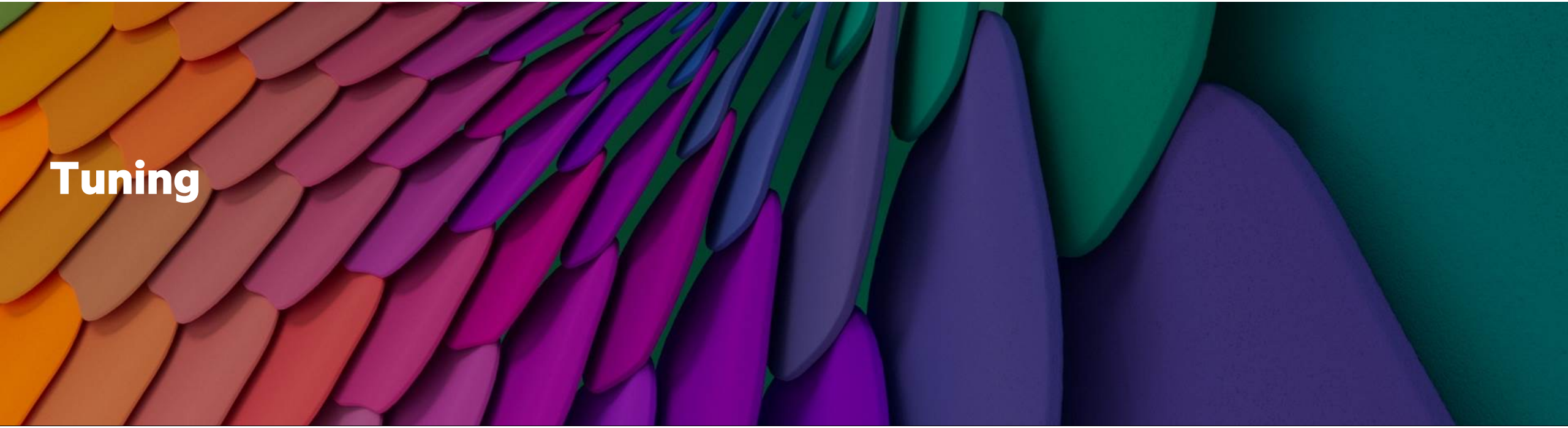
Ensuring continued system health

Ensuring continued system health

- Nexus
  - Export Nexus data to have a backup
  - Prune old data no longer needed (old software releases)
    - Open HPE ticket to get advice about what can be safely pruned
- BOS
  - Prune old BOS sessions (no TTL with BOSv1)
- CFS
  - Prune old CFS sessions (especially if all recent ones are showing failed)
    - or adjust CFS batcher backoff settings
- Check for active (unfinished) FAS sessions
- IMS review number of images and whether any are stale enough to be removed
- UAS/UAI review set of images available for containerized login
- Acknowledge and clear old alerts from various alerting tools
- Report on Kubernetes replicasets that have co-located replicas
  - This may reduce resiliency if a node with co-located replicas fails

    ```
    ncn# sat k8s --co-located-replicas
    ```
- Check that disk space on master and worker nodes has less than 85% utilization in / filesystem
  - Evicted pods indicate resource shortage

Tuning

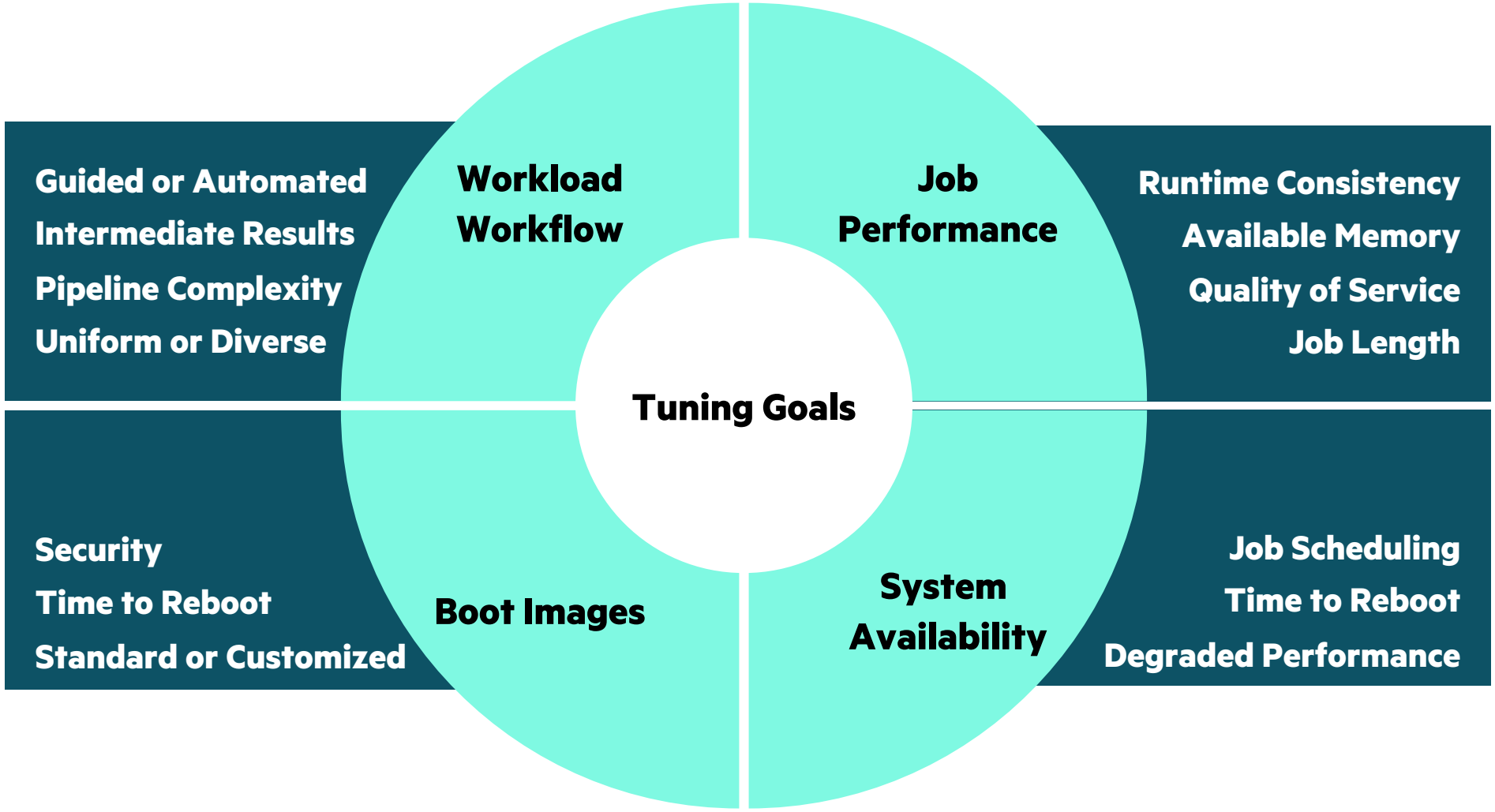


Tuning Agenda

- **Determine tuning goals**
- **Management infrastructure tuning**
- **Network tuning**
- **Compute node tuning**
- **Workload manager tuning**
- **Find underperforming nodes**

Determine tuning goals

Determine Tuning Goals

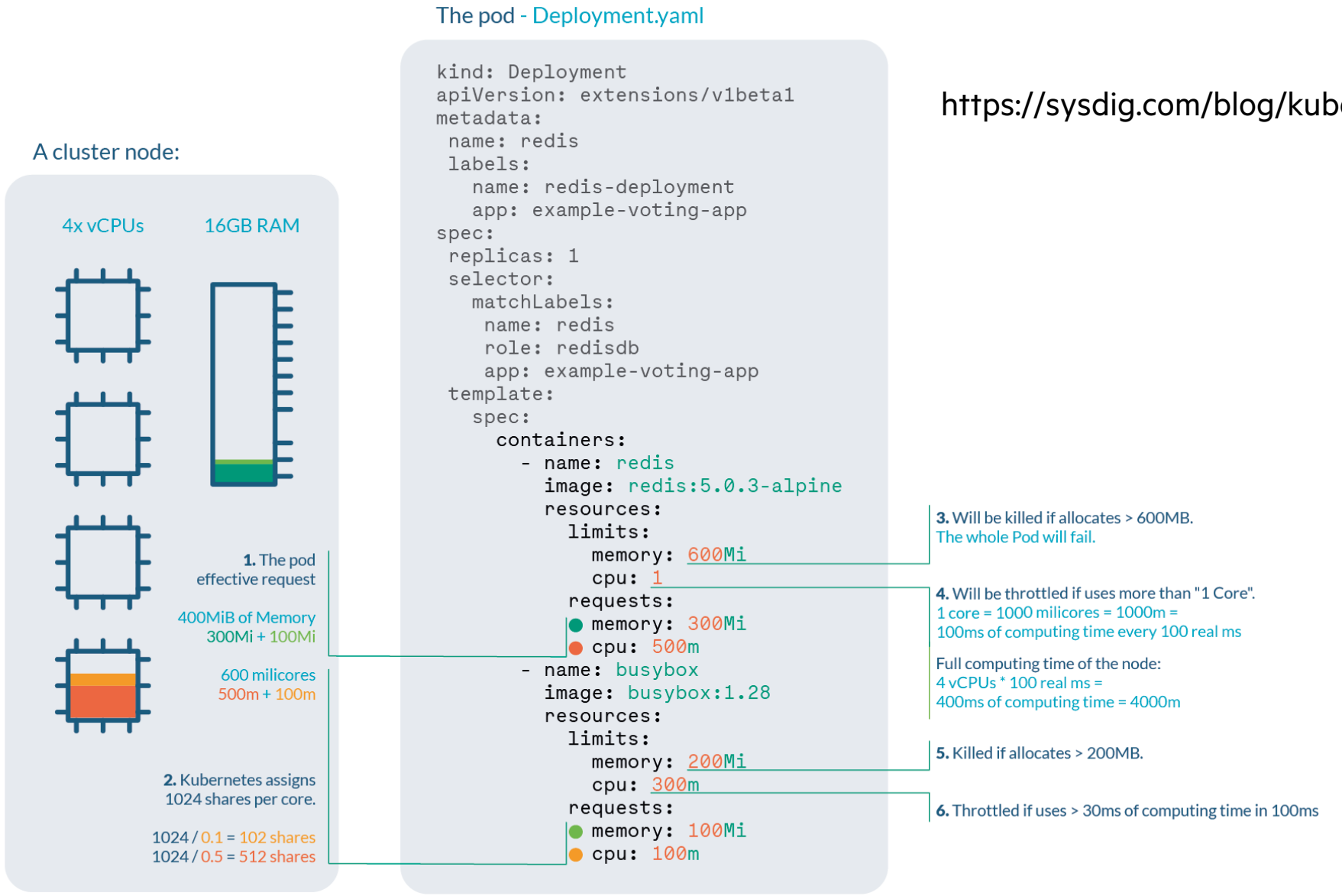


Management infrastructure tuning

- Kubernetes limits and requests for CPU and memory
- Tuning SMA components

Kubernetes Limits and Exceptions

https://sysdig.com/blog/kubernetes-limits-requests/



CPU and Memory Limits

```
ncn# kubectl get LimitRange --all-namespaces
NAMESPACE         NAME                              CREATED AT
backups           cpu-mem-limit-range               2023-01-16T18:03:02Z
ceph-cephfs       cpu-mem-limit-range               2023-01-16T18:03:01Z
ceph-rbd          cpu-mem-limit-range               2023-01-16T18:03:01Z
default           cpu-mem-limit-range-requests      2023-01-16T18:03:03Z
hnc-system        cpu-mem-limit-range               2023-01-16T18:40:51Z
ims               cpu-mem-limit-range               2023-01-16T18:03:02Z
istio-system      cpu-mem-limit-range               2023-01-16T18:03:02Z
kyverno           cpu-mem-limit-range               2023-01-16T18:40:51Z
loftsman          cpu-mem-limit-range               2023-01-16T18:03:01Z
metallb-system    cpu-mem-limit-range               2023-01-16T18:03:01Z
operators         cpu-mem-limit-range               2023-01-16T18:40:51Z
pki-operator      cpu-mem-limit-range               2023-01-16T18:40:51Z
services          cpu-mem-limit-range               2023-01-16T18:40:51Z
sma               cpu-mem-limit-range               2023-01-16T18:03:02Z
sysmgmt-health    cpu-mem-limit-range               2023-01-16T18:03:02Z
uas               cpu-mem-limit-range               2023-01-16T19:00:32Z
user              cpu-mem-limit-range               2023-01-16T19:00:32Z
vault             cpu-mem-limit-range               2023-01-16T18:40:51Z
velero            cpu-mem-limit-range               2023-01-16T18:03:03Z
```

Pod Memory usage

```
ncn# kubectl top pod --all-namespaces --sort-by=memory
```

NAMESPACE	NAME	CPU(cores)	MEMORY(bytes)
sma	opensearch-masters-1	1674m	35868Mi
sma	opensearch-masters-0	2540m	35497Mi
sma	opensearch-masters-2	1873m	35416Mi
sma	elasticsearch-master-0	22m	32969Mi
sma	elasticsearch-master-2	21m	32782Mi
sma	elasticsearch-master-1	21m	32766Mi
sma	cluster-kafka-2	1022m	20816Mi
sma	cluster-kafka-1	891m	13585Mi
sma	cluster-kafka-0	550m	11155Mi
sysmgmt-health	prometheus-cray-sysmgmt-health-kube-p-prom-0	1992m	10873Mi
sysmgmt-health	prometheus-cray-sysmgmt-health-kube-p-prom-1	778m	10503Mi
sysmgmt-health	prometheus-cray-sysmgmt-health-kube-p-prom-shard-1-1	1126m	10112Mi
sysmgmt-health	prometheus-cray-sysmgmt-health-kube-p-prom-shard-1-0	1721m	9262Mi
sma	sma-postgres-cluster-1	183m	8235Mi
sma	sma-postgres-cluster-0	121m	5362Mi
nexus	nexus-54d5bb4b95-92ctf	18m	4658Mi

Pod CPU Usage

```
ncn# kubectl top pod --all-namespaces --sort-by=cpu
NAMESPACE        NAME                                                  CPU(cores)   MEMORY(bytes)
sma              prometheus-kafka-adapter-b4bcb4947-5t7q5              5996m        1047Mi
kyverno          cray-kyverno-656d495b6-gspwn                          3551m        344Mi
sysmgmt-health   prometheus-cray-sysmgmt-health-kube-p-prom-shard-1-1  2018m        9695Mi
sysmgmt-health   prometheus-cray-sysmgmt-health-kube-p-prom-1          1818m        10458Mi
sma              opensearch-masters-0                                  1747m        35209Mi
sysmgmt-health   prometheus-cray-sysmgmt-health-kube-p-prom-shard-1-0  1678m        9277Mi
kube-system      kube-apiserver-ncn-m002                               1625m        2814Mi
sma              opensearch-masters-2                                  1475m        35618Mi
sysmgmt-health   cray-sysmgmt-health-thanos-query-0                    1362m        457Mi
sma              opensearch-masters-1                                  1129m        35891Mi
services         cray-power-control-75486588d7-t7ljx                   1103m        1031Mi
sma              cluster-kafka-2                                       804m         20489Mi
sysmgmt-health   prometheus-cray-sysmgmt-health-kube-p-prom-0          803m         10399Mi
kube-system      kube-apiserver-ncn-m001                               750m         2635Mi
sma              cluster-kafka-1                                       634m         13491Mi
```

Are pods hitting CPU limits?

- Check all pods

```
ncn# /opt/cray/platform-utils/detect_cpu_throttling.sh
Checking cray-ceph-csi-cephfs-nodeplugin-zqqvv
Checking cray-ceph-csi-cephfs-provisioner-6458879894-cbhlx
Checking cray-ceph-csi-cephfs-provisioner-6458879894-wlg86
Checking cray-ceph-csi-cephfs-provisioner-6458879894-z85vj
Checking cray-ceph-csi-rbd-nodeplugin-62478
*** CPU throttling for containerid
0f52122395dcf209d851eed7a1125b5af8a7a6ea1d8500287cbddc0335e434a0: ***
nr_periods 14338
nr_throttled 2
throttled_time 590491866
```

How many full periods have been elapsed

The total time the tasks were not run because of being over quota

- Check a specific pod

```
ncn# /opt/cray/platform-utils/detect_cpu_throttling.sh externaldns
Checking cray-externaldns-external-dns-6988c5d5c5-795lb
*** CPU throttling for containerid
76f45c4c18bf8ee6d4f777a602430e021c2a0d0e024380d22341414ca25ccffd: ***
nr_periods 6066669
nr_throttled 23725
throttled_time 61981768066252
```

The number of times the full allowed bandwidth was exhausted

Are pods hitting memory limits?

- Was a pod killed or restarted because it reached its memory limit?

```
ncn# kubectl get events -A | grep -C3 OOM
default 54m Warning OOMKilling node/ncn-w003 Memory cgroup out of memory:
Kill process 1223856 (prometheus) score 1966 or sacrifice child
default 44m Warning OOMKilling node/ncn-w003 Memory cgroup out of memory:
Kill process 1372634 (prometheus) score 1966 or sacrifice child
```

- What pod was killed?
  - A pod on ncn-w003 had a prometheus process in the cgroup reported as OOMKilled

```
ncn# kubectl get pod -A | grep prometheus
```

- Increase pod resource limits (CPU or memory)
  - https://cray-hpe.github.io/docs-csm/en-15/operations/kubernetes/increase_pod_resource_limits/

CPUThrottlingHigh Alert from Prometheus

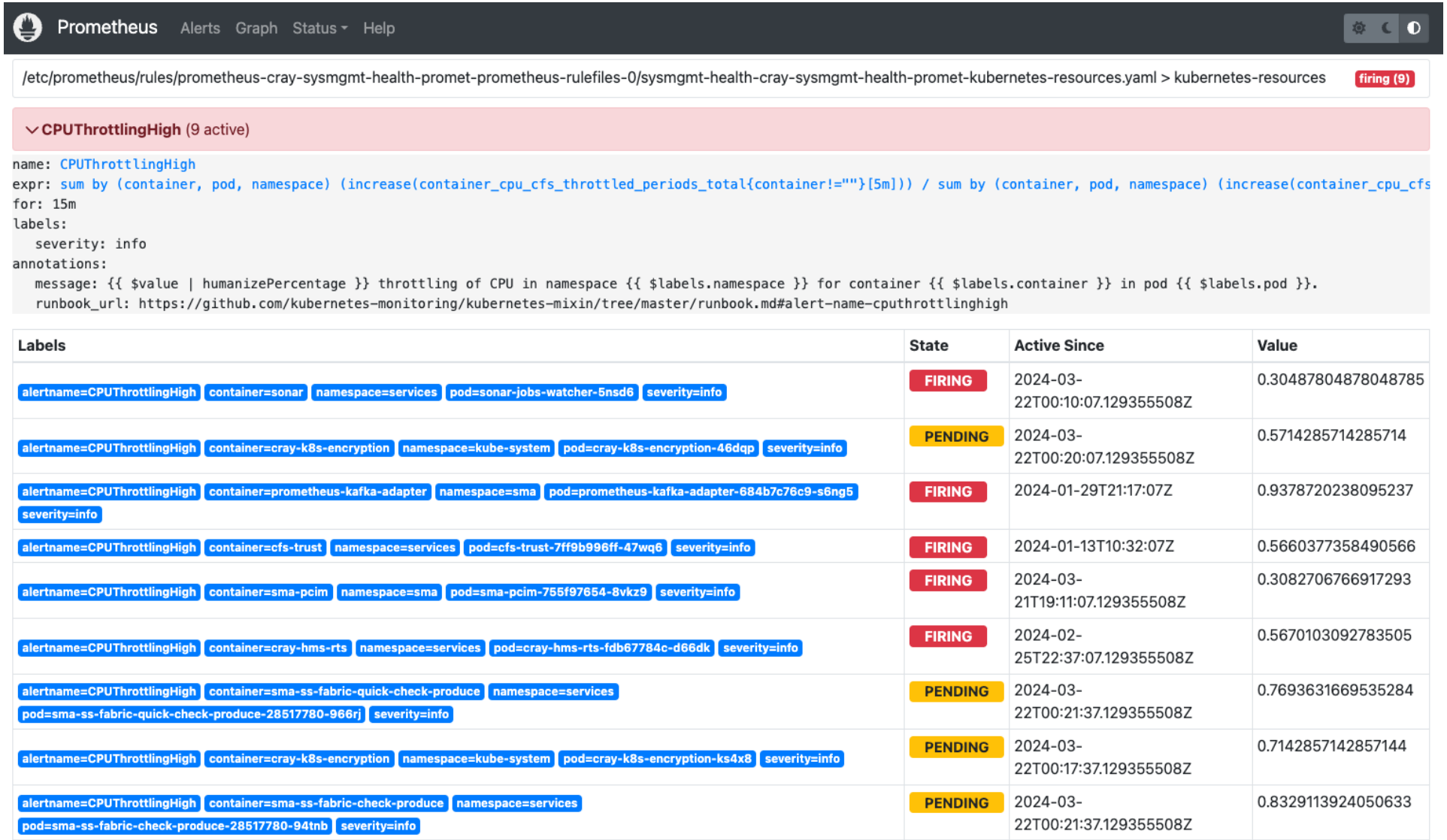
- CSM 1.4 and earlier

```
ncn# kubectl -n sysmgmt-health get svc cray-sysmgmt-health-promet-prometheus
```

- CSM 1.5

```
ncn# kubectl -n sysmgmt-health get svc cray-sysmgmt-health-kube-p-prometheus
NAME                                        TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)    AGE
cray-sysmgmt-health-kube-p-prometheus   ClusterIP   10.21.141.187   <none>        9090/TCP   34d
ncn# curl -s http://10.21.141.187:9090/api/v1/alerts |jq -j '.data.alerts \
| map(select(.labels.alertname == "CPUThrottlingHigh")) | max_by(.activeAt)'
{
  "labels": {
    "alertname": "CPUThrottlingHigh",
    "container": "manager",
    "namespace": "gatekeeper-system",
    "pod": "gatekeeper-controller-manager-588d6476db-d5g8v",
    "severity": "info"
  },
  "annotations": {
    "message": "28.03% throttling of CPU in namespace gatekeeper-system for container manager in
pod gatekeeper-controller-manager-588d6476db-d5g8v.",
    "runbook_url": "https://github.com/kubernetes-monitoring/kubernetes-
mixin/tree/master/runbook.md#alert-name-cputhrottlinghigh"
  },
  "state": "pending",
  "activeAt": "2022-04-27T16:11:07.129355508Z",
  "value": "2.803060813532017 3e-01"
}
```

Prometheus – alerts (CPUThrottlingHigh)



Labels	State	Active Since	Value
alertname=CPUThrottlingHigh container=sonar namespace=services pod=sonar-jobs-watcher-5nsd6 severity=info	FIRING	2024-03-22T00:10:07.129355508Z	0.30487804878048785
alertname=CPUThrottlingHigh container=cray-k8s-encryption namespace=kube-system pod=cray-k8s-encryption-46dqp severity=info	PENDING	2024-03-22T00:20:07.129355508Z	0.5714285714285714
alertname=CPUThrottlingHigh container=prometheus-kafka-adapter namespace=sma pod=prometheus-kafka-adapter-684b7c76c9-s6ng5 severity=info	FIRING	2024-01-29T21:17:07Z	0.9378720238095237
alertname=CPUThrottlingHigh container=cfs-trust namespace=services pod=cfs-trust-7ff9b996ff-47wq6 severity=info	FIRING	2024-01-13T10:32:07Z	0.5660377358490566
alertname=CPUThrottlingHigh container=sma-pcim namespace=sma pod=sma-pcim-755f97654-8vkz9 severity=info	FIRING	2024-03-21T19:11:07.129355508Z	0.3082706766917293
alertname=CPUThrottlingHigh container=cray-hms-rts namespace=services pod=cray-hms-rts-fdb67784c-d66dk severity=info	FIRING	2024-02-25T22:37:07.129355508Z	0.5670103092783505
alertname=CPUThrottlingHigh container=sma-ss-fabric-quick-check-produce namespace=services pod=sma-ss-fabric-quick-check-produce-28517780-966rj severity=info	PENDING	2024-03-22T00:21:37.129355508Z	0.7693631669535284
alertname=CPUThrottlingHigh container=cray-k8s-encryption namespace=kube-system pod=cray-k8s-encryption-ks4x8 severity=info	PENDING	2024-03-22T00:17:37.129355508Z	0.7142857142857144
alertname=CPUThrottlingHigh container=sma-ss-fabric-check-produce namespace=services pod=sma-ss-fabric-check-produce-28517780-94tnb severity=info	PENDING	2024-03-22T00:21:37.129355508Z	0.8329113924050633

https://prometheus.cmn.SYSTEM_DOMAIN_NAME

Tuning SMA Postgres

- Is SMA Postgres storage nearly exhausted?
  - When storage is completely exhausted, sma-postgres becomes inaccessible, requiring a manual recovery

```
ncn# kubectl exec -it -n sma sma-postgres-cluster-0 -c
postgres -- df -h | egrep "Used|pgdata"
Filesystem            Size  Used Avail Use% Mounted on
/dev/rbd28            5.7T  5.4T  291G  95%
/home/postgres/pgdata
ncn# kubectl exec -it -n sma sma-postgres-cluster-1 -c
postgres -- df -h | egrep "Used|pgdata"
Filesystem            Size  Used Avail Use% Mounted on
/dev/rbd47            5.7T  5.3T  525G  91%
/home/postgres/pgdata
```

- sma-pgdb-cron Kubernetes cronjob
  - Runs once daily to delete data from the SMA database which is older than a user-configurable retention period
  - Provides the first level of data pruning to prevent Postgres from exhausting its Persistent Volume Claim (PVC)
- sma-pgdb-prune Kubernetes cronjob
  - Runs hourly to check the size of SMA database components
    - sma database, pmdb database, write ahead log (WAL) records in the WAL database, other overhead databases
  - Deletes old data from the sma and pmdb databases if it grows beyond a given percentage of its allocated space
  - Provides a second level of data pruning and only acts when sma-pgdb-cron is unable to restrict the database from growing too fast and exceeding the space allotment
- Change the SMA Postgres Retention Policy
  - Configmap postgres-config
    - Default retention period is 7 days, but it may need to be lengthened or shortened
    - Run sma-pgdb-cron-test cronjob (only once) after changing retention period
    - Delete this cronjob once it is done so normal cronjob can handle pruning daily
  - Configmap hms-postgresql-pruner
    - Set retention_time to same value as postgres-config configmap
    - Run hms-postgresql-pruner-test cronjob (only once) after changing retention period
    - Delete this cronjob once it is done so normal cronjob can handle pruning hourly

Tuning SMA OpenSearch

- Change the Number of OpenSearch Pods
  - Likely the same as the number of worker nodes
- Change the Number of OpenSearch Shards for Each Index
  - Same as number of opensearch nodes
  - Number of replicas: Commonly 1, but might need to be increased
- Change the OpenSearch Log Volume Size
  - Initial size set based on best practices and number of nodes at installation
    - Increase when
      - Number of nodes changes
      - Log volume is unexpectedly high
      - Desired retention period is longer than is allowed by the available storage
- Identify and Remedy Unassigned OpenSearch Shards
  - Opensearch may exceed the java heap space available to the pod
    - If this happens data written by rsyslog will be retried
    - But index replicas shard allocation is not automatically healed and must be reallocated manually
- Set Disk Usage Limits for OpenSearch
  - Configure how OpenSearch logs will be pruned
    - This can help prevent OpenSearch from consuming all the storage in its PVC when incoming log data volume spikes and when the default disk usage limits are insufficient to delete old data fast enough to prevent OpenSearch storage from completely filling up
- Change the OpenSearch Index Retention Policy
  - Adjust the min_index_age to retain data for longer

Tuning SMA Kafka

- Resize Kafka PVC for each of three pods

```
ncn# kubectl -n sma edit pvc data-cluster-kafka-0
ncn# kubectl -n sma edit pvc data-cluster-kafka-1
ncn# kubectl -n sma edit pvc data-cluster-kafka-2
```

- Adjust Kafka pruning parameters
  - Disable purging of Kafka data due to size limits when there is ample storage for a 4 hour retention time
  - Reduce the amount of data stored for a specific kafka topic if storage is insufficient for that data
  - Both disk usage limit and retention policy data pruning are enabled by default

Tuning SMA Monasca

- Monasca service
  - Default memory 1216MB
  - Java heap size 870MB
- Under heavy load, OOM may happen

```
ncn# kubectl describe –n sma-monasca-thresh-dmtf
o.a.s.d.worker [ERROR] Error when processing event java.lang.OutOfMemoryError: Java heap space
```

- Change the configuration values in the sma-monasca section of customizations.yaml for permanent change

```
sma-monasca:
  thresh:
    maxHeapMB: ”990"
    resources:
      limits:
        memory: ”1600"Mi
```

- Change on running system

```
ncn# kubectl -n sma edit deployment sma-monasca-thresh-dmtf
- env:
  - name: MAX_HEAP_MB
   value: ”990"
...
resources:
  limits:
    memory: ”1600"Mi
```

Tuning SMA LDMS

- LDMS data can be collected from all compute nodes or a subset
  - Ensure that LDMS compute node aggregation service is running on worker nodes
- LDSM data can be collected from NCNs
  - Ensure that LDMS NCN aggregation service is running on worker nodes
- Aggregation pods

```
ncn# kubectl get pods -o wide -A | grep agg
sma sma-ldms-aggr-compute-0 2/2 Running 0  4d10h   10.38.0.76    ncn-w001
sma sma-ldms-aggr-ncn-0     2/2 Running 0  4d9h    10.38.0.97    ncn-w001
```

- Can be extended with additional LDMS samplers
  - SMA Admin guide
    - Add Customer Provided Samplers to LDMS v4 Configuration

Network tuning

- Tuning ARP cache
- Tuning HSN

Tuning ARP cache

- Reduce the frequency of ARP cache misses during connection establishment
  - MPI jobs depend on ARP table for establishing TCP/IP connection for setup and tear down information
  - Needed for booting nodes over NMN
- See guidance and procedures in Slingshot Operations Guide and CSM documentation
- Create static ARP entries for the HSN NICs on all compute nodes during their boot sequence to avoid invalidating the ARP cache (removing them)
- Sysctl settings to adjust
  - `gc_thresh1` is the minimum number of entries to keep
    - The garbage collector will not purge entries if there are fewer than this number in the ARP cache
    - Recommended: 4096, default: 128
      - Number of nodes on HSN multiplied by (square of number of NICs per node)
  - `gc_thresh2` is the threshold where the garbage collector becomes more aggressive about purging entries
    - Entries older than 5 seconds will be cleared when greater than this number
    - Recommended: 4096, default 512
      - `gc_thresh1` multiplied by 1.5
  - `gc_thresh3` is the maximum number of non-PERMANENT neighbor entries allowed
    - Increase this when using large numbers of interfaces and when communicating with large numbers of directly-connected peers
    - Recommended: 8192, default 1024
      - `gc_thresh2` multiplied by 2
  - Set `gc_stale_time` to 4 minutes to reduce the frequency of ARP broadcasts on the network
    - Recommended: 240, default: 30
  - Setting `base_reachable_time_ms` to a very high value avoids ARP thrash
    - Recommended: 1500000, default: 30000

Tuning HSN

- To achieve high TCP performance
  - TCP window size
    - TCP window size is the amount of received data that can be buffered during a connection
    - The size is calculated based on the latency bandwidth of the link speed and Round-Trip Time (RTT) using the following formula:
      - TCP window size = 2 (Throughput* RTT)
      - Where, expected throughput is in bits/sec and RTT in ms
    - Note: RTT is system dependent and measures the duration of end-to-end communication of data packet, include the latency introduced by a gateway when connecting to another network
  - Enable fair queuing
    - Use one queue per packet flow and service them in rotation, this ensures that each flow receives an equal fraction of the resources
  - Pace and shape the bandwidth
    - Queuing theory explains that bursty traffic produces higher queueing delays, more packet losses, and lower throughput
    - TCP congestion control mechanisms can create burst traffic flows on high bandwidth and multiplexed networks
    - Therefore, smoothing the behavior of TCP traffic by evenly spacing data transmissions across a round-trip time increases the performance
    - A flow limit of 10,000 is chosen to gain a couple of Gbps in comparison to a flow limit of 1000
  - Set Tx and Rx queue sizes based on the traffic load
    - The default Rx and Tx queues are set to 8
      - If a high traffic load is given, you can set Rx and Tx queues to 16
  - Set Linux Network Tx queue size based on the traffic load
    - The default Network Tx queue are set to 1000
    - If a high traffic load of MTU 1500 is given, you must set the Network Tx queue to 10000
  - Set CPU governor to Performance to maintain a higher clock speed limit, thus increasing the performance

Tuning HSN - Current recommendations for TCP settings

- The recommended settings are based on the TCP window size formula and RTT of 0.13ms which may differ for each system
  - To use 8 MB buffers with RTT 0.13 ms, add the following to /etc/sysctl.conf file:
    ```
    sysctl -w net.core.rmem_max=8000000
    sysctl -w net.core.wmem_max=8000000
    ```
  - To enable Fair Queuing, use the following command (for each HSN NIC):
    ```
    tc qdisc replace dev hsn0 root mq
    ids+=( $(tc qdisc show dev hsn0 | awk '{print $5} ' | sed s'/.*:// ') )
    for i in ${ids[@]}; do tc qdisc replace dev hsn0 parent "8001:$i" fq; done
    tc qdisc show dev hsn0
    ```
  - To pace and shape the bandwidth, set the TCP window size to 8 MB buffers:
    ```
    sysctl -w net.ipv4.tcp_rmem ="4096 87380 8000000"
    sysctl -w net.ipv4.tcp_wmem ="4096 65536 8000000"
    ```
  - To enable 16 Rx 16 Tx queue, use the following command (for each HSN NIC): :
    ```
    ethtool -L hsn0 rx 16
    ethtool -L hsn0 tx 16
    ```
  - To set the Network Tx queue length to 10000, use the following command (for each HSN NIC): :
    ```
    ip link set dev hsn0 txqueuelen 10000
    ```
  - To further pace and shape the bandwidth by increasing the flow limit to 10,000 flows, use the following (for each HSN NIC): :
    ```
    for i in {1..10}; do tc qdisc replace dev hsn0 parent "8001:$i" fq flow_limit 10000 maxrate 34gbit; done
    for i in {a..f}; do tc qdisc replace dev hsn0 parent "8001:$i" fq flow_limit 10000 maxrate 34gbit; done
    ```
  - To set the CPU Frequency governor, use the following command:
    ```
    cpupower frequency-set -g performance
    ```

Compute node tuning

- Low Noise Mode (LNM)
- DVS
- CPS tuning
- Overlay preload for DVS

Low Noise mode (LNM)

- Configures Linux kernel so OS tasks are migrated to one or more CPUs (on each node) which are excluded from application use
  - Full LNM
    - Kernel parameters at boot will reduce noise by moving system activities to CPU 0 (and potentially additional CPUs as well) and user space is configured to move any overhead processes to CPU 0
      - Will probably want more than one CPU used for system activities
  - Lightweight mode
    - The kernel parameters are not used, but the user space configuration is still done
  - Must coordinate COS kernel parameters and WLM settings for LNM
- IRQs can be listed by name instead of by number, and there are new options for selecting to what CPUs the IRQs are directed
- At boot, systemd runs `lnmctl` to configure the user space environment for low noise
  - `lnmctl` reads default configuration from `/etc/lnm-default.json` and site configuration from `/etc/lnm.json`

LNM kernel parameters

- In full LNM mode, the Linux kernel must be booted with parameters to direct noise overhead to CPU 0
  - The CPU range has to be specified for the number of CPUs on the node

```
nohz_full=1-255
rcu_nocbs=1-255
rcu_nocb_poll
```

- In addition, two parameters are specified on the kernel command line to guide the behavior of the user space configuration
  - The lnm parameter must be set to either full or lightweight.

```
lnm={ full, lightweight }
```

  - The lnm.cpu parameter is optional and is set to what CPU(s) to use to handle system overhead

```
[lnm.cpu=0]
```

- These boot parameters are set in a BOS session template
- Check the status of Low Noise on a node when invoked as shown below:

```
/usr/sbin/lnmctl --status
Mode: full
CPU: [0, 63]
```

  - This indicates that Low Noise Mode is active in its “full” state and CPUs 0 and 63 are the “system” CPUs

LNM configuration file

- Tunables:
  - Sysctl: Settings to be set with the sysctl utility, and their values
  - Files: files to change (typically under /sys) and the values
  - Systemd: lists systemd services to start or stop, and the action
- Processes:
  - By default, all processes are migrated, with the exception of processes such as per-CPU threads that cannot be migrated
  - The default LNM configuration file excludes IRQ processes, Lustre processes, and Spectrum Scale processes from migration to the system CPU(s)
- IRQs:
  - IRQ 0 to CPU 0
  - IRQ 1 to the highest numbered CPU
  - IRQ 2 to a list of CPUs
  - Interrupts with gpu in the name to the closest CPU
  - Interrupts with ‘hsn0’ in the name are spread across the system CPUs in round-robin order
  - Interrupts with ‘net1’ in the name are spread across the list of CPUs in round-robin order

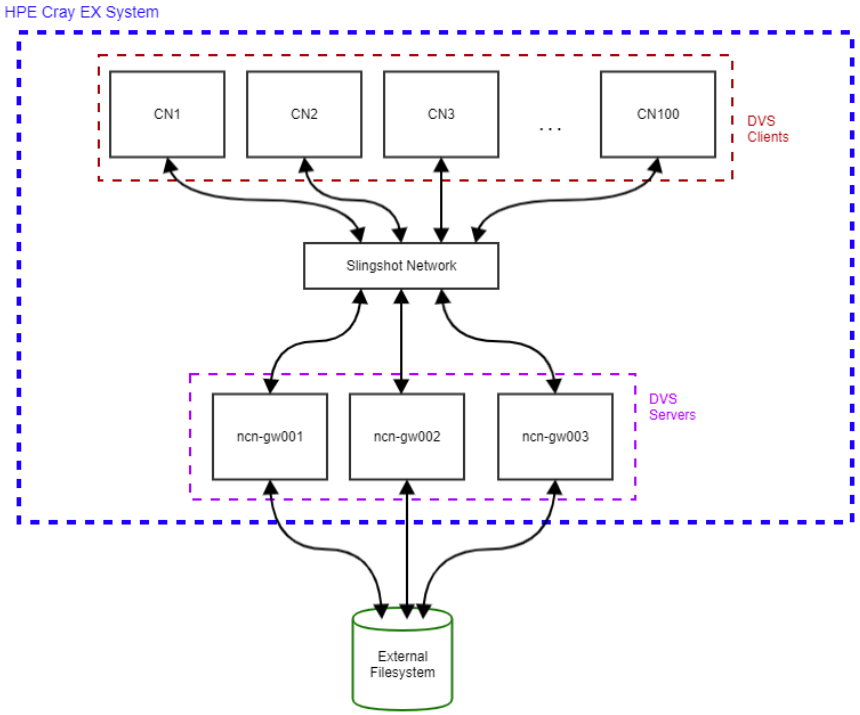
```
{
  "Tunables": {
    "sysctl": {
      "vm.stat_interval": 120
    },
    "files": {
      "/sys/bus/workqueue/devices/writeback/cpumask": “CPUMASK”,
      "/sys/kernel/mm/transparent_hugepage/enabled": "never"
    },
    "systemd": {
      "irqbalance.service": "stop"
    }
  },
  "Processes": [".*watchdog.*" ,"DVS-IPC_msg" ,"acceptor_[0-9]*",
     "ldlm_bl_.*" ,"ldlm_cb.*" ,"ll_cfg_requeue" ,"lst_s_.*" ,
     "lst_t_.*" ,"ptlrpc.*" ,"socknal_.*"],
  "IRQs": {
    "0": "cpu_0",
    "1": "cpu_last",
    "2": "cpu_all:1,2,4,8",
    ".*gpu": "cpu_closest:1,2,4,8”,
    ".*hsn0": "round-robin"
    ".*net1": "round-robin:1,2,4,8"
  },
  "CPU": "0,128”
}
```

Low noise mode Slurm

- Avoid placing applications on CPU 0 if some or all system compute nodes are configured with the Low Noise Mode feature
- Slurm.conf
  - SchedulerParameters=spec_cores_first
    - Use core 0 instead of last core
  - AllowSpecResourcesUsage=YES
    - (Optional) allows users to override the specialized cores with `srun -S`
  - NodeName=nid000010 Sockets=2 CoresPerSocket=16 ThreadsPerCore=2 RealMemory=55296 Feature=Intel_Xeon_Gold_6130 CoreSpecCount=1
    - On each node configured with LNM, avoid one core by default

DVS

- Data Virtualization Service (DVS)
  - Distributed network service projects file systems mounted on NCNs to other nodes within the system
  - Projecting makes a file system available on nodes where it does not physically reside
  - DVS-specific configuration settings enable clients to access a file system projected by DVS servers
  - Represents a software layer that provides scalable transport for file system services
  - Uses Lustre Networking (LNet) to communicate over the network
    - LNet configuration is done by the code that configures DVS
- Works with CPS to project internal file systems to nodes
- Projecting external file systems from gateway nodes
  - DVS provides I/O performance and scalability to many nodes
    - Far beyond the number of clients supported by a single NFS server
  - HPE DVS configuration minimizes
    - Operating system noise
    - Impact on compute node memory resources
  - DVS
    - Uses Linux virtual file system (VFS) interface to process file system operations
    - Can project Any POSIX-compliant file system
      - such as Spectrum Scale (GPFS), NFS, and Lustre
  - Gateway nodes need custom OS images built to support Spectrum Scale

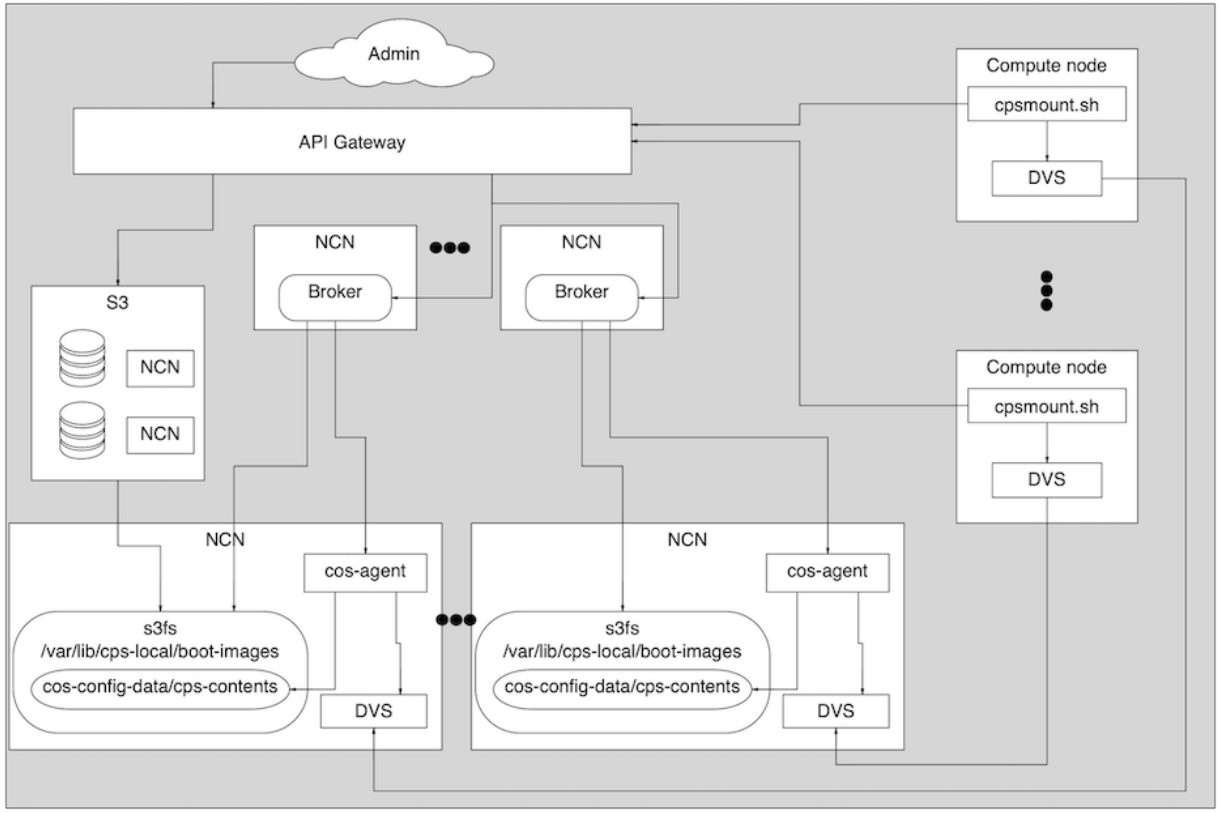


Compute node Root File System Mounts

- All files in the compute node root file system (rootfs) are provided from a squashFS image stored in S3 (Ceph)
- Compute node rootfs images are projected by CPS pods and mounted via DVS
- Rootfs images are mounted on compute nodes with `/opt/cray/cps-utils/bin/cpsmount.sh` and are mounted read-only
  - A compute node local overlay file system is configured to enable writes "on top of" the `rootfs` to an ephemeral in-memory file system
- DVS mount content is accessed over the network on demand
  - When a block is first referenced, DVS caches the content in the node-local Linux page cache so future references to that data will not involve the network
    - If available memory gets too low, Linux can evict these pages, and thus the data will be accessed over the network again (and cached again) if/when they are referenced again
    - Overlay Preload can permanently "pin" files in memory on the compute node at boot time so they can never be evicted
- DVS can also project other filesystems unrelated to CPS
  - Projections of user file systems using DVS can be configured as read-write or read-only

What Is the Content Projection Service (CPS)

- The Content Projection Service (CPS) is a container-based microservice managed by Kubernetes
  - The main components of CPS are
    - Brokers provide the API service
    - cos-agent takes requests from CPS to maintain CPS contents data, DVS server list, and DVS exports
    - s3fs mounted contents
    - CPS internal data
- At node boot Boot Script Service (BSS) provides
  - The Linux kernel
  - `initrd`
  - Boot parameter data
- CPS provides
  - Node's root file system image (operating system image)
  - HPE Cray Programming Environment (CPE) images



`cray cps contents` provides a list of images being managed by the content manager

`cray cps deployment` provides a list of CPS pods and their statuses

`cray cps transports` provides a list of images currently being exported (served) to nodes

CPS tuning

- PodAntiAffinity ensures that there will be no more than one instance of cray-cps pods per worker node
- Scaling the number of cray-cps pods is helpful for maintaining resiliency and load-balancing
  - Default: 2 pods

```
ncn# kubectl -n services scale --current-replicas=2 --replicas=3 deployment/cray-cps
```

- CPS will not deploy any cm-pm pods with CSM 1.5/COS Base 3.0.0/USS 1.0.0

Cray Overlay Preload

- Compute node root filesystem utilizes the Linux overlayfs architecture
  - Read-only lower layer that uses the Data Virtualization Service (DVS)
  - Read-write, RAM based upper layer
  - This supports copying files from the lower layer to the upper layer to increase performance and support writes
- The Overlay Preload feature uses this copy operation to increase performance on frequently accessed files
  - A list of files is provided at boot, and they are all copied into local memory
  - All future references to those files are serviced by the local file system, rather than requiring remote data and/or metadata DVS operations
  - This improves system and application performance
    - However, the amount of memory available on the node is reduced by the cumulative size of all files copied into its memory
- The total amount of memory used by Overlay Preload can be configured by the system administrator to balance the performance and memory requirements of the system
- The system is shipped with a default list of files to be preloaded
  - This list is specific to the operating system release provided and the IO access recorded during system boot
  - The administrator can modify this list if desired
    - Sites may define their own file lists to optimize work for specific workloads
    - The Overlay Preload package ships with a script to aid in determining which files are accessed at boot time
      - It will analyze boot behavior and produce a list of files accessed

Configure Cray Overlay Preload

- Configuration Settings
  - Overlay Preload configuration is managed using the CFS overlay-preload Ansible role
    - overlay-preload-size-limit
      - The size, in MB, that limits the amount of file data that is promoted to the overlay cache
        - A value of 0 indicates ‘unlimited’ file data
- File lists
  - The list of files to be preloaded at boot are located in the file /opt/cray/overlay-preload/config/dist/overlay-preload.filelist
    - The file format is a list of file paths, one per line, with support for wildcard values
  - The file list in the default boot image may be modified
  - The file is read early in the boot process, and files will be processed in order
    - If there are constraints placed on total preload size, processing will stop once the limit is reached
      - In this case, files that are critical for preloading should be placed first
- The Overlay Preload Log File and Symlinks
  - Overlay Preload creates a log file on affected nodes at /var/log/cray/overlay-preload.log
    - The log file contains warnings for files that were not found, as well as the number and size of the files preloaded on the node
  - Any symlinks included in a file list may not be copied from the lower layer to the node-local RAM file system, which might look confusing
    - For example, if a site’s content list contains /etc/alternatives/unzip, which is a symlink to /usr/bin/unzip-plain
    - In this case, both the link and its target are present in lower layer, but neither of them appear in the node-local file system
    - This is expected and correct behavior
    - A site that is concerned about possible confusion for administrators can decide to exclude symlinks from file lists, or simply list the target of the symlink in a file list to ensure that it is present in the node-local file system

Custom cray Overlay Preload

- Create Custom Loads for Specific Workloads
  - Sites may define their own file lists to optimize work for specific workloads
    - Either create an Ansible play for CFS to run pre-boot for image customization or use the IMS method to jump into the image customization process via ssh to run a command
  - The following is a general workflow for this process:
    - Enable the cray-preload-strace service in the image that will be booted

      ```
      image# systemctl enable cray-preload-strace
      ```
    - Boot a compute node with the new filesystem image
    - Log into the compute node as root and kill any strace process
      - The strace log can be found at /cray-preload-strace.log.
    - Run the `preload-strace-analyze.sh` script with the strace log as input

      ```
      compute# preload-strace-analyze.sh /cray-preload-strace.log
      ```
      - The output will be a list of files, access counts, and sizes
      - The sum is included at the bottom
      - This can be used as the basis for creating or modifying an overlay file list
    - Disable the cray-preload-strace service

      ```
      compute# systemctl disable cray-preload-strace
      ```

Workload management tuning

- Slurm and PBS Pro
- Slurm config for HPE 200GB (Cassini) NICs
- Application Task Orchestration and Management (ATOM)

Workload Management (WLM)

Slurm and PBS Pro

- Actively working with SchedMD and Altair on HPE Cray Ex system
- Both WLMs supported

HPE Cray WLM services

- PALS – Parallel Application Launch Service
  - libpals is used for both PBS Pro and Slurm
  - Launcher part of PALS (mpiexec, aprun, palsd) is only used for PBS Pro
- Application Task Orchestration and Management (ATOM)
  - application and job prologue and epilogue task runner
    - compute node cleanup
    - node health checking
    - energy usage reporting

Slurm config for HPE 200GB Cassini NICs

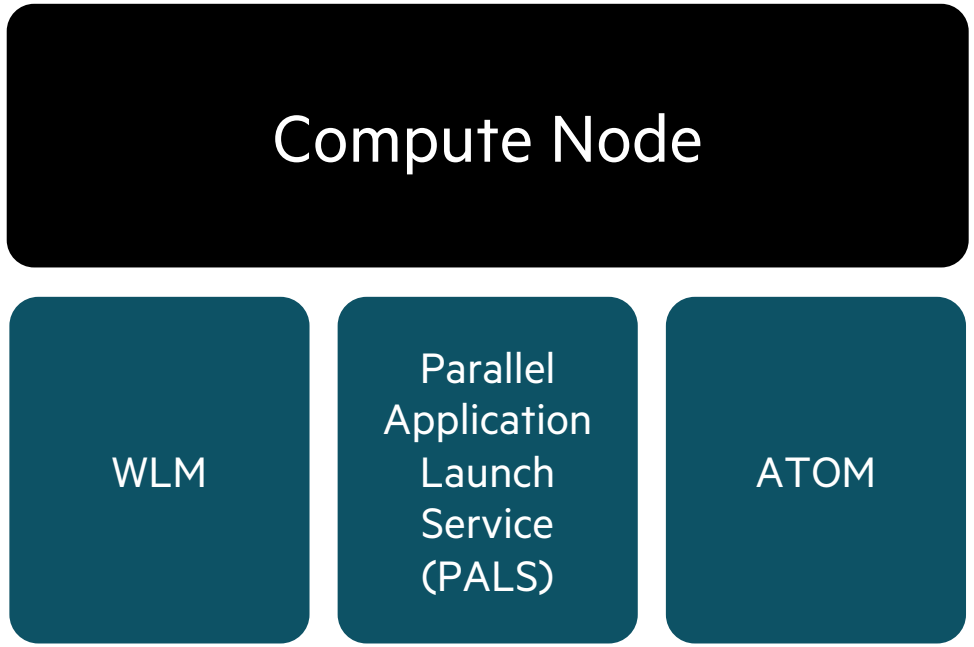
- Set SwitchType=switch/hpe_slingshot in slurm.conf
- SwitchParameters determine behavior
  - vnis=<min>-<max> - Range of VNIs to allocate for jobs and applications
    - Default is 32768-65535.
  - tcs=<class1>[:<class2>]... - Set of traffic classes to configure for applications
    - Supported traffic classes are [DEDICATED_ACCESS], [LOW_LATENCY], [BULK_DATA] and [BEST_EFFORT]
  - single_node_vni=<all|user|none> - Allocates single node VNI as follows:
    - Not set - Does not allocate VNI for single-node job steps.
    - single_node_vni (no value) - Allocates a VNI for all job steps
    - single_node_vni=all - Allocates a VNI for all job steps.
    - single_node_vni=user - Allocates a VNI for single-node job steps using the srun --network=single_node_vni option or SLURM_NETWORK=single_node_vni environment variable
    - single_node_vni=none - Does not allocate VNI for single-node job steps
  - job_vni=<all|user|none> - Allocates job VNI as follows:
    - Not set - Does not allocate additional VNI for jobs
    - job_vni (no value) - Allocates an additional VNI for jobs, shared among all job steps
    - job_vni=all - Allocates an additional VNI for jobs, shared among all job steps
    - job_vni=user - Allocates an additional VNI for any job either using the srun --network=job_vni option or SLURM_NETWORK=job_vni environment variable
    - job_vni=none - Does not allocate additional VNI for jobs
  - adjust_limits
    - If set, slurmd sets an upper bound on network resource reservations by taking the per-NIC maximum resource quantity and subtracting the reserved or used values (whichever is higher) for any system network services
      - This is the default
  - no_adjust_limits
    - If set, slurmd calculates network resource reservations based only upon the per-resource configuration default and number of tasks in the application
      - It does not set an upper bound based on resource usage of already-existing system network services
      - Setting no_adjust_limits can result in more application launch failures due to network resource exhaustion, but if an application requires a certain amount of resources, this option ensures it

More Slurm SwitchParameters

- Set SwitchType=switch/hpe_slingshot in slurm.conf
- SwitchParameters determine behavior
  - jlope_url=<url> - If set, slurmctld uses the configured URL to request Instant On NIC information, from the HPE jackalope daemon REST API, for each node in a job step
  - jlope_auth=<BASIC|OAUTH> - HPE jackalope daemon REST API authentication type, default is OAUTH
  - jlope_authdir=<directory> - Directory containing authentication information files. Default is /etc/jackaloped for BASIC authentication and /etc/wlm-client-auth for OAUTH authentication.
  - def_<rsrc>=<val> - Per-CPU reserved allocation for this resource.
  - res_<rsrc>=<val> - Per-node reserved allocation for this resource. If set, overrides the per-CPU allocation.
    - max_<rsrc>=<val> - Maximum per-node application for this resource.
- Resources are:
  - txqs - Transmit command queues. The default is 2 per-CPU, maximum 1024 per-node.
  - tgqs - Target command queues. The default is 1 per-CPU, maximum 512 per-node.
  - eqs - Event queues. The default is 2 per-CPU, maximum 1023 per-node.
  - cts - Counters. The default is 1 per-CPU, maximum 1023 per-node.
  - tles - Trigger list entries. The default is 1 per-CPU, maximum 2048 per-node.
  - ptes - Portable table entries. The default is 6 per-CPU, maximum 2048 per-node.
  - les - List entries. The default is 16 per-CPU, maximum 16384 per-node.
  - acs - Addressing contexts. The default is 4 per-CPU, maximum 1022 per-node.

Application Task Orchestration and Management (ATOM)

- General purpose job and application prologue and epilogue task runner
  - Configuration
  - Compute node cleanup
  - Node health testing
- ATOM is only called by PALS and WLMs
- ATOM REST API is not exposed on the network
  - Users cannot call ATOM APIs directly
- Extensible and configurable by the customer
  - New tasks added by dropping in a new task configuration file
  - Runs tasks in lexical order, so sites can choose ordering
- Tasks can be disabled or enabled by site administrator or user
  - Site administrator can force some tasks to run or not permit others to be enabled
- In compute node image, `/etc/sysconfig/atomd` contains configurable variables which control file locations and settings for ATOM daemon



What Is a Task?

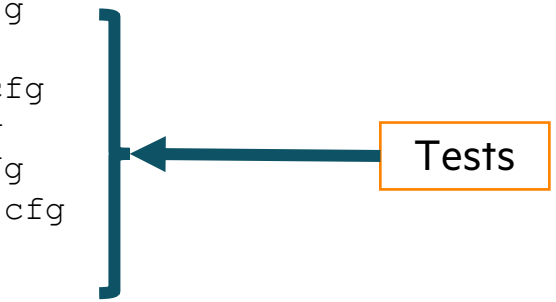
- ATOM daemon startup
  - Initialize Boot FreeMem
- Compute node cleanup
  - Clear VM/Lustre cache
  - Compact memory
- Node health
  - Free memory check
- Reporting
  - Task stats

- Any executable action that is run at a specified time
  - “On this event, run this script and if it fails, do this”
  - “On this event, run this script and if it succeeds, do this”
- Command can be inline commands or executed (Python/shell/binaries)
- Executed in filename lexical order

```
010_bootfreemem_init
{
    "name": “bootfreemem_init",
    "description": "Initialize /proc/boot_freemem",
    "onSuccess": [],
    "onFailure": [],
    "events": ["startup"],
    "timeout": 2,
    "command": ["/bin/sh", "-c", "echo 1 >/proc/boot_freemem"],
    "enabled": true,
    "userControl": false
}
```

ATOM Task configuration files

```
nid001000# ls -1 /etc/atom.d
010_bootfreemem_init.cfg
015_make_acct_dir.cfg
020_clear_lustre_caches.cfg
020_clear_lustre_caches_job.cfg
025_clean_tmpdirs.cfg
030_clear_vm_cache.cfg
040_compact_memory.cfg
040_compact_memory_job.cfg
080_memerror_test.cfg
085_pcie_test.cfg
090_hugepages_test.cfg
100_freemem_test.cfg
110_zeropage_test.cfg
150_filesystem_test.cfg
160_gpu_test.cfg
170_mce_test.cfg
180_pcie_devices_list_init.cfg
190_clean_cxiservices.cfg
200_energy_end.cfg
200_energy_start.cfg
800_admindown.cfg
850_reboot.cfg
900_panic.cfg
999_hello_atom.cfg
```



ATOM Task Execution Files

- Execution files are in /opt/cray/atom/sbin and are referenced in the “command” field
  - Test for zero page memory corruption at job end

```
nid001000# cat /etc/atom.d/110_zeropage_test.cfg
{
    "name": "zeropage_test",
    "description": "Check for zero page memory
corruption",
    "onSuccess": [],
    "onFailure": ["admindown"],
    "events": ["jobEnd"],
    "timeout": 5,
    "command": ["/opt/cray/atom/sbin/zeropage"],
    "enabled": true,
    "userControl": false,
    "exclusive": false
}
```

- Compact fragmented memory at end of every application and job so hugepage allocations remain efficient

```
nid000001# cat
/etc/atom.d/040_compact_memory.cfg
{
    "name": "compact_memory",
    "description": "Compact fragmented memory to
allow better hugepages allocation",
    "onSuccess": [],
    "onFailure": [],
    "events": ["appEnd", "jobEnd"],
    "timeout": 30,
    "command":
["/opt/cray/atom/sbin/compact_memory.py"],
    "enabled": true,
    "userControl": true
}
```

CFS configuration for ATOM

- ATOM configuration is done by CFS, so add or change data in VCS (git)
  - Configuration settings can be used to specify directory paths
    - atom_filesystems
      - list of directory paths mounted on all compute nodes to check at application and job end time
    - atom_tmpdirs
      - list of directory paths to be cleaned up at job end time
  - Create the `group_vars/all/atom.yml` file in the pbs-config-management or slurm-config-management git repository
  - Edit and populate it with the desired settings. For example:

```
atom_filesystems:
  - "/scratch"

atom_tmpdirs:
  - "/tmp"
  - "/var/tmp"
  - "/dev/shm"
```

  - Can override or add new ATOM configuration files or tasks

`roles/atom/files/config/`
`roles/atom/files/tasks/`

Find underperforming nodes

- SAT hwmatch
- SAT slscheck
- CSM Diags
- Slingshot diagnostics
- Scan console logs

SAT hwmatch

- Report hardware mismatches for processors and memory
  - Checks that certain values within processors and memory modules match
    - at the level of the node
      - which has multiple processors and memory modules
    - at the level of the node card
    - at the level of the blade slot
  - Processor values that must match are model, core count, and speed
  - Memory module values that must match are memory type, memory device type, and size
  - The number of memory modules must also match
  - The memory manufacturer must match for just the node level, not card or slot levels

```
ncn# sat hwmatch
+--------------+-------+---------------+---------------------+----------------------+
| xname        | Level | Category      | Field               | Values               |
+--------------+-------+---------------+---------------------+----------------------+
| x1011c1s5b0 | card  | memory module | Device Type         | Other (24), DDR5 (72) |
| x1107c2s2b0 | card  | node          | Memory Module Count | 24 (3), 23 (1)        |
+--------------+-------+---------------+---------------------+----------------------+
```

SAT slscheck

- Perform a cross-check between SLS and HSM

```
ncn-m002:~ # sat slscheck
```

xname	SLS Type	SLS Class	SLS Role	SLS Subrole	Comparison Result
x1000c0s0b0	NodeBMC	Mountain	MISSING	MISSING	SLS component missing in HSM Components
x1000c0s0b0	NodeBMC	Mountain	MISSING	MISSING	SLS component missing in HSM Redfish Endpoints
x1000c0s0b0n0	Node	Mountain	Compute	MISSING	SLS component missing in HSM Components
x1000c0s0b0n1	Node	Mountain	Compute	MISSING	SLS component missing in HSM Components
x1002c4s7b1	NodeBMC	Mountain	MISSING	MISSING	SLS component missing in HSM Components
x1002c4s7b1	NodeBMC	Mountain	MISSING	MISSING	SLS component missing in HSM Redfish Endpoints
x1002c4s7b1n0	Node	Mountain	Compute	MISSING	SLS component missing in HSM Components
x1002c4s7b1n1	Node	Mountain	Compute	MISSING	SLS component missing in HSM Components
x3000c0s4b0	NodeBMC	River	MISSING	MISSING	Role mismatch: SLS:MISSING,HSM:Management
x3000c0s6b0	NodeBMC	River	MISSING	MISSING	Role mismatch: SLS:MISSING,HSM:Management
x3000c0s16b0n0	Node	River	Application	Lnet	SubRole mismatch: SLS:Lnet,HSM:LNETRouter
x3000c0s18b0n0	Node	River	Application	Lnet	SubRole mismatch: SLS:Lnet,HSM:LNETRouter
x3000m0	CabinetPDUController	River	MISSING	MISSING	Class mismatch: SLS:River,HSM:MISSING

Nid defragmentation procedure left artifacts in SLS for this blade type with b1 and b1n0,b1n1 components which are not in HSM

Installed as SubRole=Lnet, but later changed to LNETRouter

CSM Diags

- Can CSM Diags find underperforming nodes?
  - Are nodes uniform?
  - Is a group of nodes of the same type uniform?
- Consistency checks across all nodes
  - Change the nodes file for the test runs if you have multiple types of compute nodes in the system
- System level diagnostics
  - Run per node (olconf, linpack)
  - Run system wide (cwolconf, cwlinpack)
- GPU diagnostics
  - Per node for either AMD or Nvidia
- Fabric diagnostics
  - Run across set of nodes
  - Run system wide

CSM Diags
Consistency checks

- cpuchk
  - Checks for the uniformity in the CPU configuration on selected nodes by verifying CPU model, vendor, architecture, number of online or offline cores, number of threads and sockets on each node, and reports inconsistencies
  - Also checks the CPU operating frequency on the nodes and reports outlier nodes based on CPU operating frequency
- memchk
  - Verifies the consistency of memory configuration across selected nodes by examining the size of physical memory, swap space, DIMM size, speed, number of DIMMs per socket and balanced memory configuration on the nodes, and identifies any discrepancies
- fabricchk
  - Checks for uniformity in the slingshot fabric configuration on selected nodes by verifying HSN device states, HSN device checks, PCI link status and speed, NUMA, NIC version, and reports inconsistencies
  - Checks for any errors in IP and MAC address configuration, DNS configuration, PCIe errors and lmon errors, and reports inconsistencies
  - Performs ping check to the specified node through the specified HSN device
- netchk
  - Checks for the uniformity in the ethernet configuration on selected nodes by verifying driver name, version, firmware version, ethernet PCI value, duplex value, advertised link speed and auto-negotiation value on the nodes, and reports inconsistencies
- fschk
  - Checks for the Lustre and NFS mount point on selected nodes. Also checks for file system size and usage, and reports if the file system usage exceeds a certain threshold
- mpichk
  - Runs the MPI “hello world!” to determine the MPI health of the system

CSM Diags
System Level Diagnostics

- linpack
  - Uses High Performance Linpack (HPL), an industry standard test to check CPU performance in terms of GFLOPS for each node in the cluster
- cwlinpack
  - MPI application that uses HPL to check CPU performance in terms of GFLOPS for a group of nodes or an entire Cluster
- stream
  - An industry standard memory bandwidth performance tool that measures sustained memory bandwidth (in MB/s)
- olcmt
  - Test memory and cache components. Several memory test algorithms are used to ensure a broad test platform
- olconf
  - Confidence test which runs on each single node to test the sanity of CPU and memory
- cwolconf
  - Cluster-wide confidence test which runs on each single node. This test will test the sanity of CPU, memory.
- rank
  - MPI application that runs on a clustered system which communicates with other ranks in the cluster to stress the interconnect
- pandora
  - Stress the entire system consisting of processors, memory, I/O and network
- cwhpcc
  - Uses HPC Challenge, which is a benchmark suite that combines several benchmarks to test a number of independent attributes of the performance of HPC systems

CSM Diags
Nvidia GPU Diagnostics

- gpu-burn
  - GPU CUDA test performs single or double precision matrix multiplications to stress the GPUs and report their respective health status
- xkbandwidth
  - Measures memory bandwidth between CPU and GPU, and within the GPU memory to memory
- xkcheck
  - Checks for the consistency of the GPU properties across a selected range of nodes
- xkdgemm
  - Runs dgemm performance measurements on the GPUs
- xkmemtest
  - Memory tests on the GPU memory
- xkstress
  - Performs performance measurements on the GPU and does memory transfers right after to stress both the GPU processor and the PCIe link

CSM Diags
AMD GPU Diagnostics

- amdgpubandwidth
  - Saturate the PCIe bus with DMA transfers between system memory and a target GPU card's memory with unidirectional or bidirectional transfers
- amdgpuproperties
  - Queries the configuration of a target device and returns the device's static characteristics
- amdgpuedpp
  - Characterize the peak power capabilities of a GPU to different levels of use
- amdgpumonitor
  - Monitor and characterize the response of a GPU to different levels of use
- amdgpup2pchk
  - Provide the list of all GPUs that support P2P and characterize the P2P links between peers
  - Performs a peer-to-peer throughput test between all unique P2P pairs for performance evaluation
- amdgpupciechk
  - Targets and qualifies the configuration of the platforms PCIe connections to the GPUs
- amdgpupciemonitor
  - Actively monitor the PCIe interconnect between the host platform and the GPU
- amdgpubioschk
  - Verify that a platform's SBIOS has satisfied the BAR mapping requirements for VDI and Radeon Instinct products for ROCm support
- amdgpustresstest
  - Bring the CPUs of the specified GPU(s) to a target performance level in gigaflops by performing large matrix multiplications using SGEMM or DGEMM (Single or Double-precision General Matrix Multiplication) available in a library like rocBlas

CSM Diags
Fabric Diagnostics

- check_excessive_pause
  - Checks all ports on the fabric that are reachable from the fabric manager
  - Checks for an error flag that indicates that there are excessive amounts of PAUSE frames on a port
- dgnettest
  - Slingshot network diagnostic to measure bandwidth and latency
  - Performs all-to-all and bisection bandwidth test and latency validation

CSM Diags
OSU (Ohio State University) MPI Benchmarks

- Startup test
  - osu_startup
- pt2pt tests
  - osu_bw_bibw
  - osu_single_multi_latency
  - osu_multiplebw_message_rate
  - osu_multithread_multiprocess_latency
- One-sided tests
  - osu_bw_latency_ops
  - osu_put_bibw
  - osu_get_acc_latency
- Collective tests
  - osu_collective_blocking_barrier
  - osu_collective_MPI_blocking_ops
  - osu_collective_MPI_non_blocking_ibarrier
  - osu_collective_MPI_non_blocking_ops

- See https://mvapich.cse.ohio-state.edu/benchmarks/

Slingshot diagnostics

- HPE Slingshot Troubleshooting Guide
  - dgnettest
    - Loopback bandwidth test, latency test, fabric bisection bandwidth test, fabric all-to-all
  - cxibwcheck.sh
    - Bi-directional loopback bandwidth for each Slingshot NIC on each node in a group of nodes
  - bwcheck.sh
    - Uni-directional loopback bandwidth for each Mellanox NIC on each node in a group of nodes
  - cxiberstat.sh
    - Measures NID link corrected and uncorrected bit error rates (BERs)
- cxi_healthcheck
  - PCIe speed and width
  - Presence of PCIe errors
  - Algorithmic MAC assignment (optional)
  - Link state and speed
  - Link-layer retry setting is enabled
  - Internal loopback mode is disabled
  - Priority Flow Control (PFC) is enabled
  - Acceptable number of link flaps in the past hour (< 5) and the past 10 hours (< 10)
  - Presence of common error messages related to HPE Slingshot 200GB NIC in the kernel log
  - Services (retry handler, etc.) are in a running state (optional)
  - Resource and retry handler leak detection
  - Successful ping from HPE Slingshot 200 GB NIC interface to an external host / interface (optional)
  - Good, corrected, and uncorrected codeword rate check
  - Firmware revision check (optional)

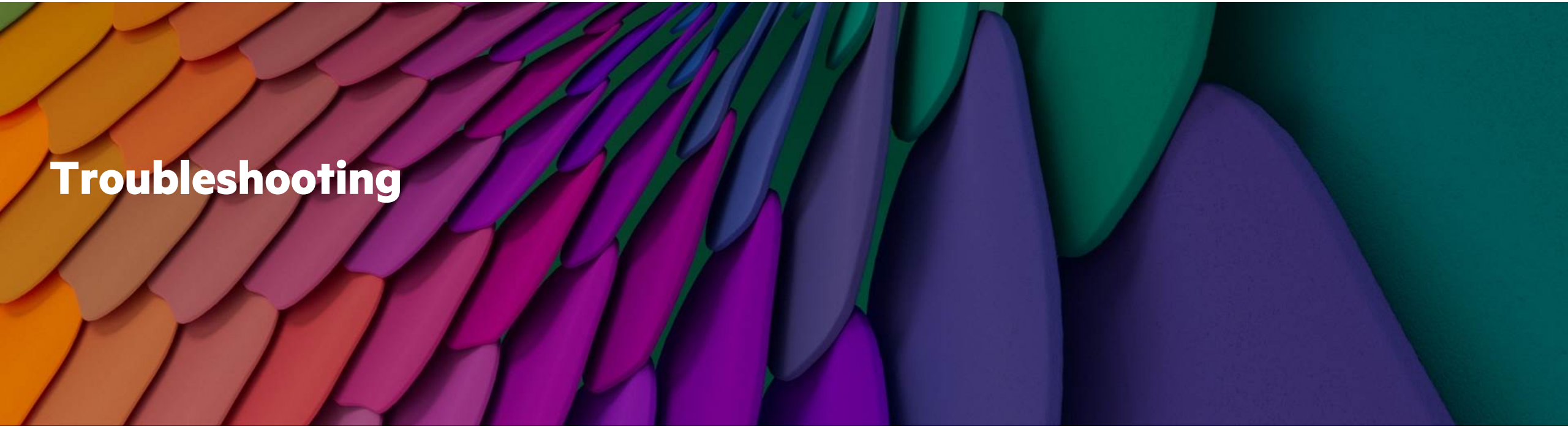
Scan console logs

- MCEs for memory DIMM errors
- HSN NIC lane degrade messages

- How do you find problems across many nodes?
  - From cray-console-node pod
    - kubectl exec into cray-console-node-0 pod
    - Find pattern from a bad node's console log in /var/log/conman/console.XNAME
    - Then grep for pattern in the rest of the console logs in /var/log/conman/console.*
  - From management node
    - ssh to BMC of liquid-cooled node
    - Find pattern in /var/log/n0/current (for node 0 on the BMC)
    - Then pdsh to group of BMCs for liquid-cooled nodes to grep for pattern in /var/log/n*/current

  - Run hwtriage tool from management node against each node considered bad
    - It will find MCEs and PCIe lane degrades (and other issues)
    - See CSM Diags Administration Guide or later in this presentation

Troubleshooting



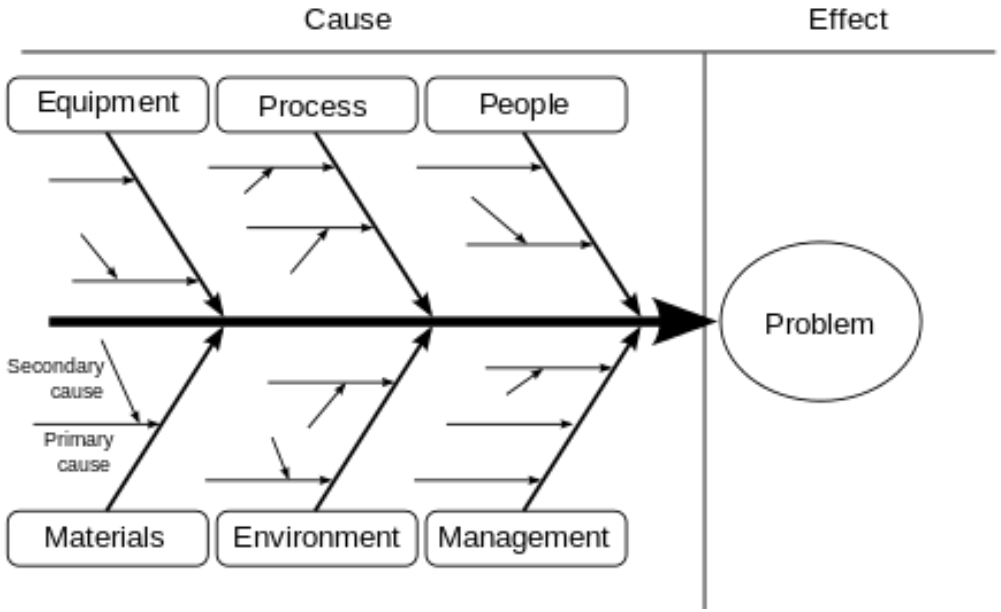
Troubleshooting Agenda

- Troubleshooting theories and methodologies
- Service dependencies
- Boot troubleshooting
- CFS troubleshooting
- Node hardware troubleshooting
- Slingshot troubleshooting
- Triage data collection for HPE Service

Troubleshooting theories and methodologies

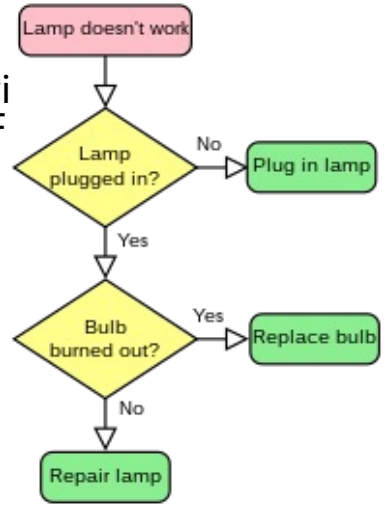
Troubleshooting theories and methodologies

- Ishikawa diagram (or cause-and-effect diagram)
  - https://upload.wikimedia.org/wikipedia/commons/5/52/Ishikawa_Fishbone_Diagram.svg
  - Identify potential factors causing an overall effect
  - Each cause or reason for imperfection is a source of variation
  - Causes are usually grouped into major categories to identify and classify these sources of variation



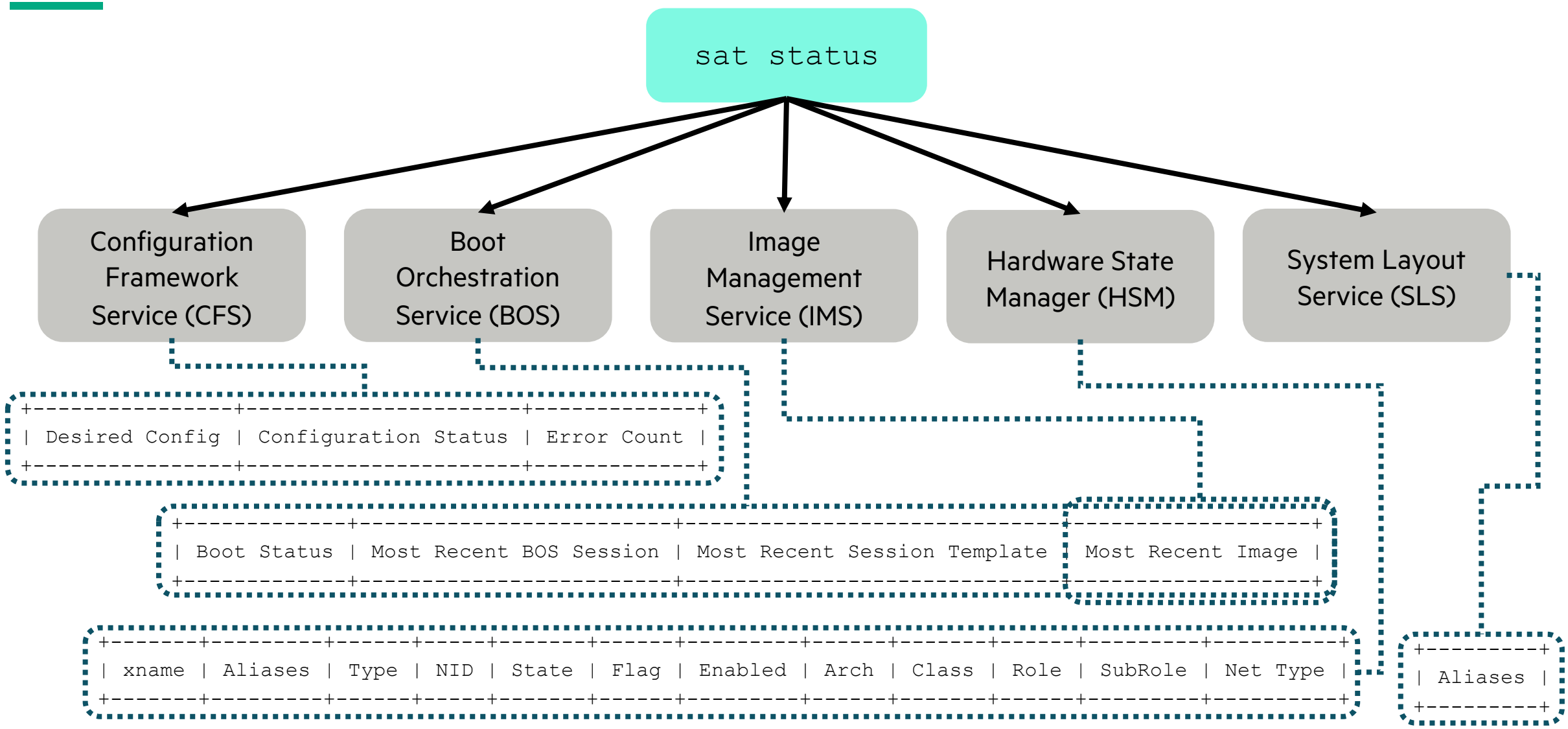
- Root cause analysis
  - Identify and describe the problem clearly
  - Establish a timeline from the normal situation until the problem occurs
  - Distinguish between the root cause and other causal factors (for example, using event correlation)
  - Establish a causal graph between the root cause and the problem

- Flow chart (process flow)
  - https://upload.wikimedia.org/wikipedia/commons/9/91/LampFlowchart.svg
  - Represents a workflow or process with arrows connecting boxes
    - Process step is a rectangular box
    - Decision is a diamond



Service dependencies

SAT status API Requests



Boot troubleshooting

- What a Normal Boot Looks like
- When Things Go Wrong

What a Normal Boot Looks like

Power action

- NodeController (BMC) receives request to power on the node from CAPMC or PCS
- Node starts displaying POST messages from BIOS

PXEboot

- DHCP
- TFTP
- BSS
- Ceph S3 storage (kernel and initrd)

Dracut scripts in single-user Linux

- DHCP (again)
- Spire
- DVS/LNET
- Pivot root to DVS-mounted squashfs OS rootfs

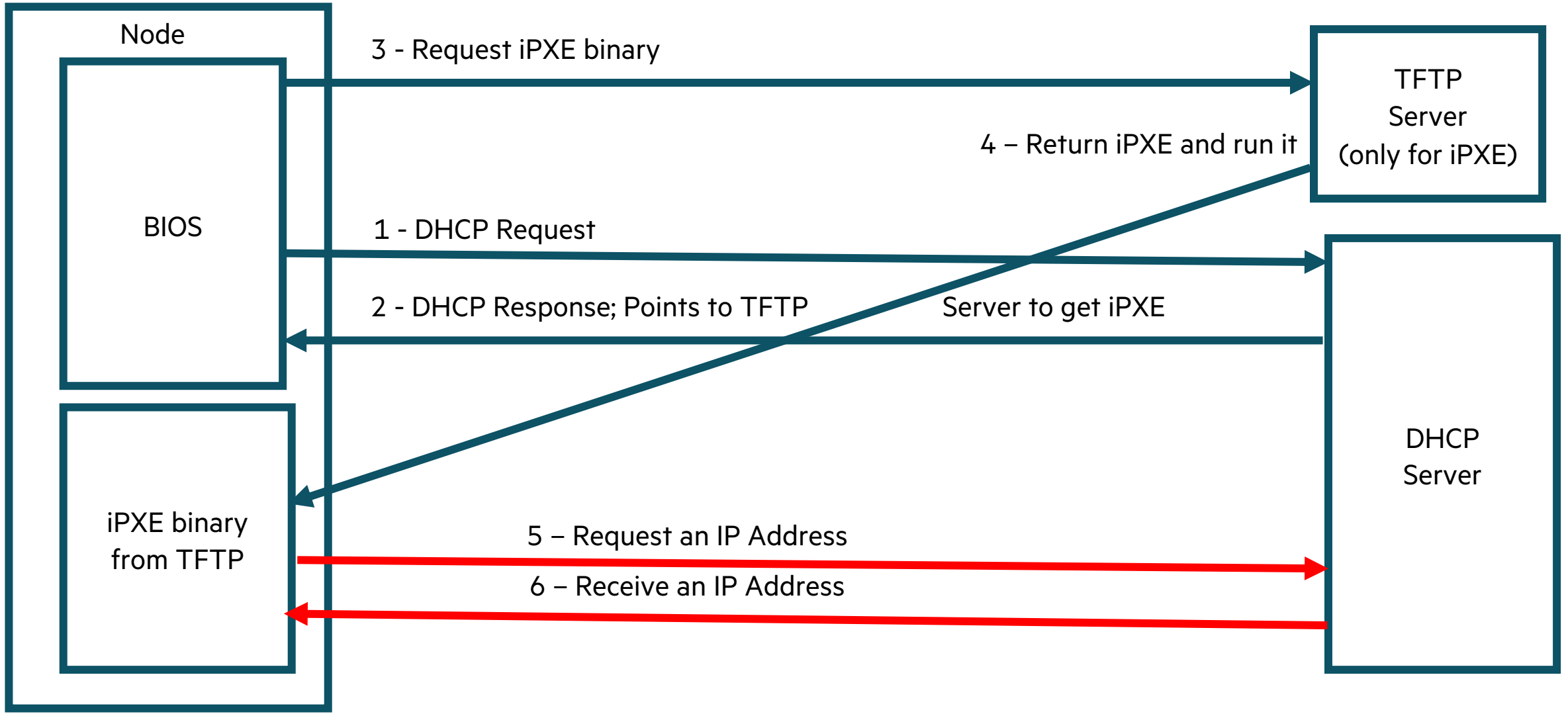
Multi-user Linux available

- BOS reporter
- CFS state reporter
- Node heartbeat

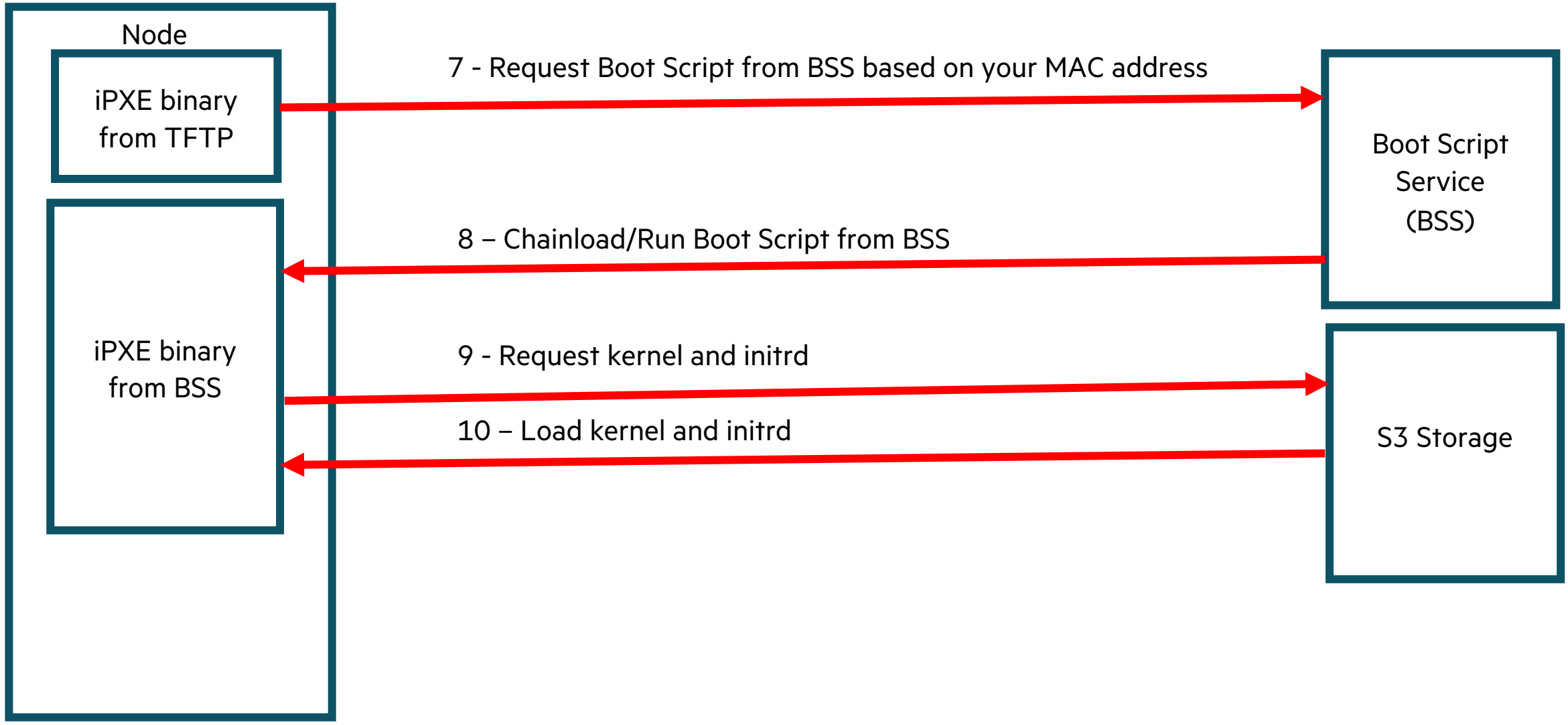
Post-boot CFS Configuration

- CFS runs layers of Ansible from desired configuration for node

Boot Diagram Steps 1 -6



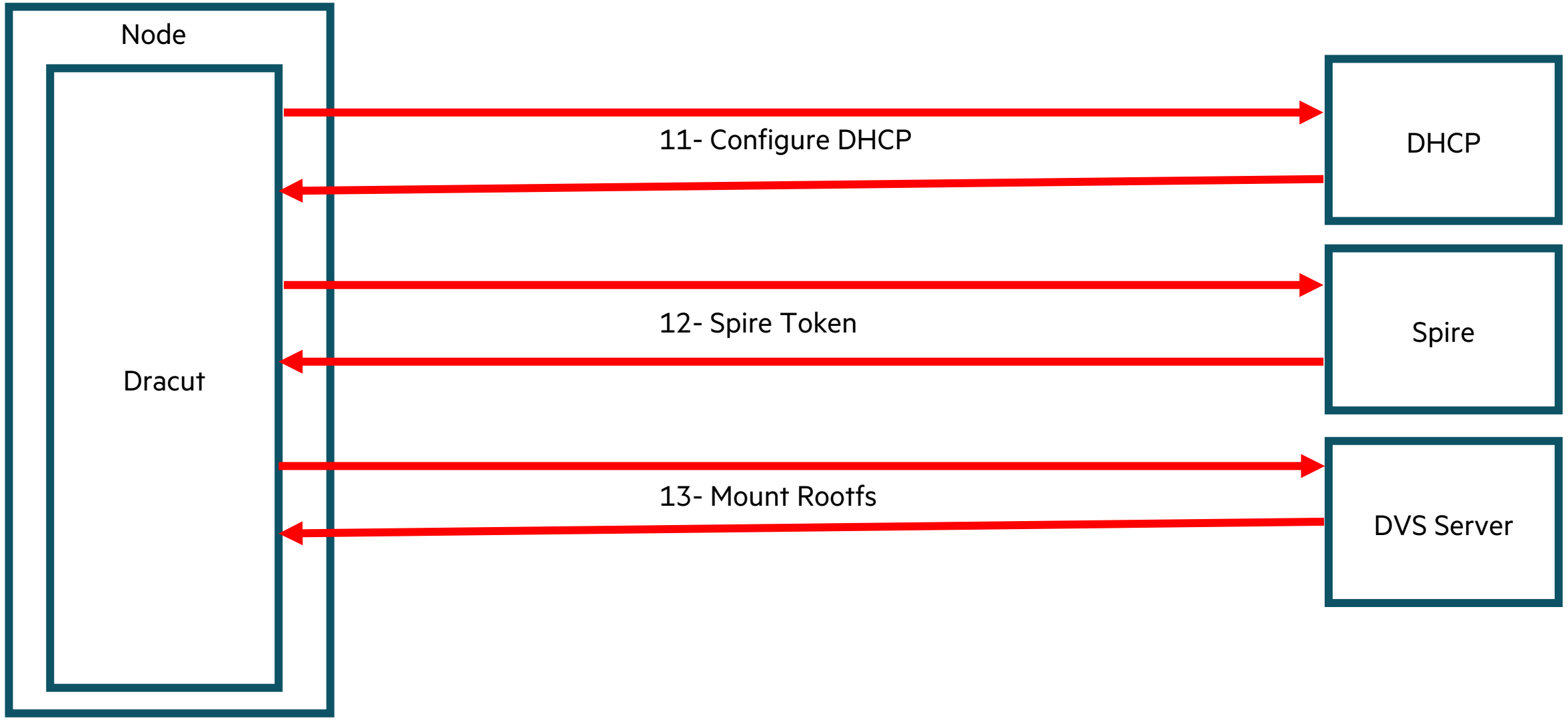
Boot Diagram Steps 7 - 10



Dracut – Whole bunch of Stuff happens in dracut

- Hop into Initrd – run Dracut scripts
- Dracut-initqueue configures NICs via DHCP (this is the third time we've configured DHCP, if you've been counting)
  - NMN NICs
  - HSN NICs
- Chronyd is started
- Spire-agent is started
- Configure CPS, DVS, and LNET
- Move to Dracut-pre-mount phase
- We do the switch root to the root file system

Dracut Diagram

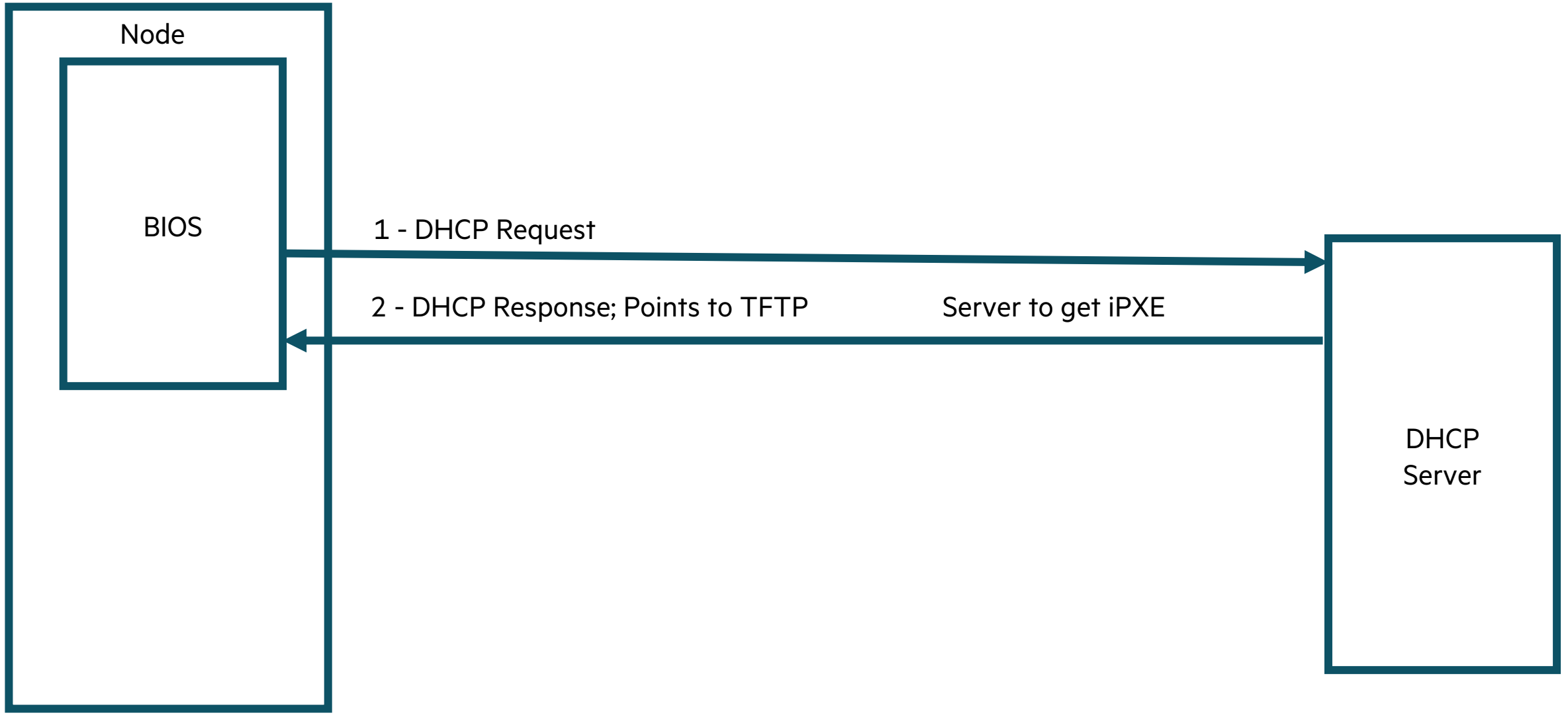


What does this look like in the console?

Node PXEboot – Console

- 2023-06-17 13:29:51 >>Start PXE over IPv4 on MAC: 00-40-A6-84-7B-B7InstallProtocolInterface: EfiPxeBaseCodeCallback 86293190

First DHPC Request



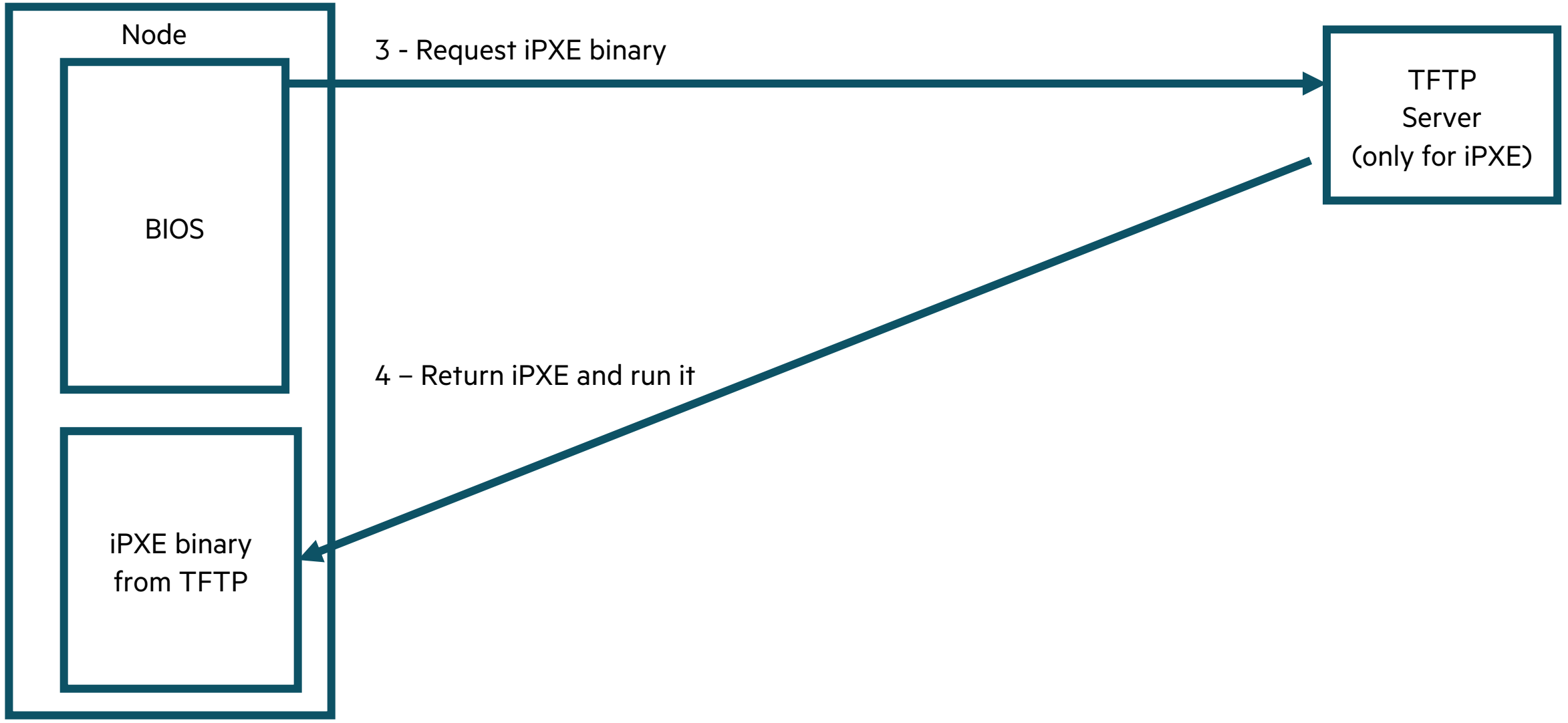
First DHCP Request – Console

2023-06-30 20:33:34 Station IP address is 10.100.0.40

IP address offered to the node: 10.100.0.40

Node only reports this on a successful DHCPACK

TFTP Request



TFTP Messages in the console

- TFTP Request

```
2023-06-30 20:33:34 Server IP address is 10.92.100.60
2023-06-30 20:33:34 NBP filename is ipxe.efi
2023-06-30 20:33:34 NBP filesize is 1004800 BytesInstallProtocolInterface:
B4D311E6-721D-4051-956D-2410C1D789AA 0
```

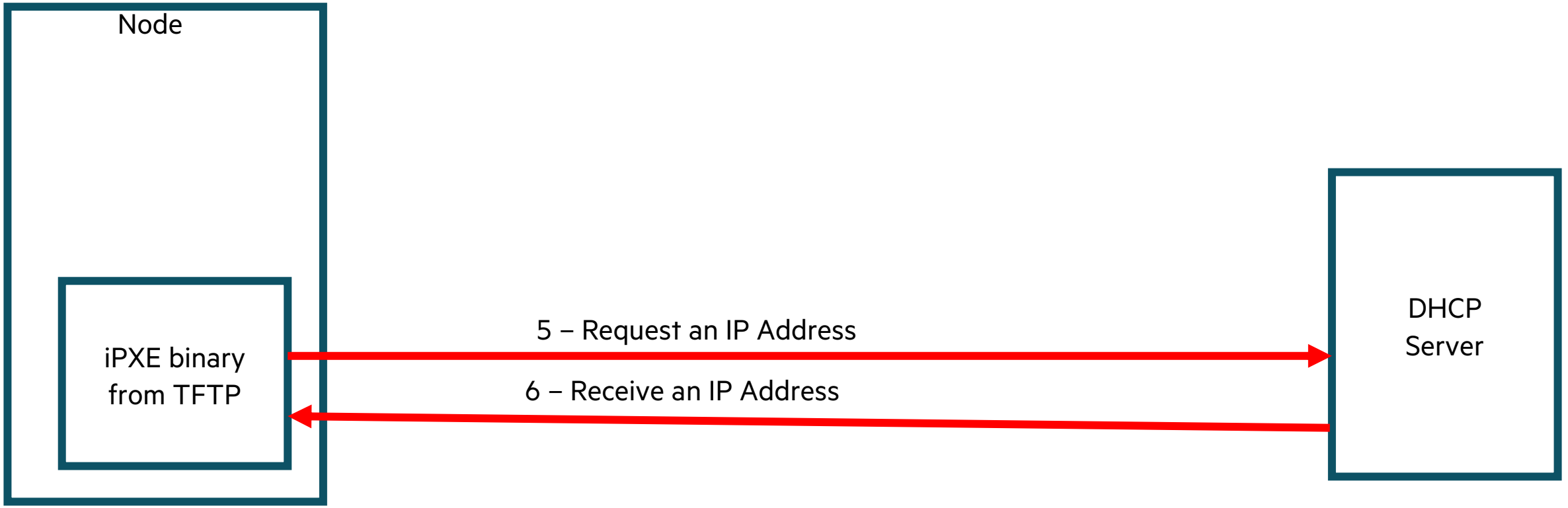
- Node Downloads iPXE binary from TFTP server

```
2023-07-06T09:39:53.95474 Downloading NBP file...
2023-07-06T09:39:53.95475
2023-07-06T09:39:53.95476 NBP file downloaded successfully.
```

- Node Executes iPXE binary

```
2023-07-06T09:39:53.95502 iPXE initialising devices...
2023-07-06T09:40:13.96315 iPXE 1.0.0+-- Open Source Network Boot Firmware --
http://ipxe.org
2023-07-06T09:40:13.96316 Features: DNS HTTP HTTPS iSCSI TFTP SRP AoE EFI Menu
```

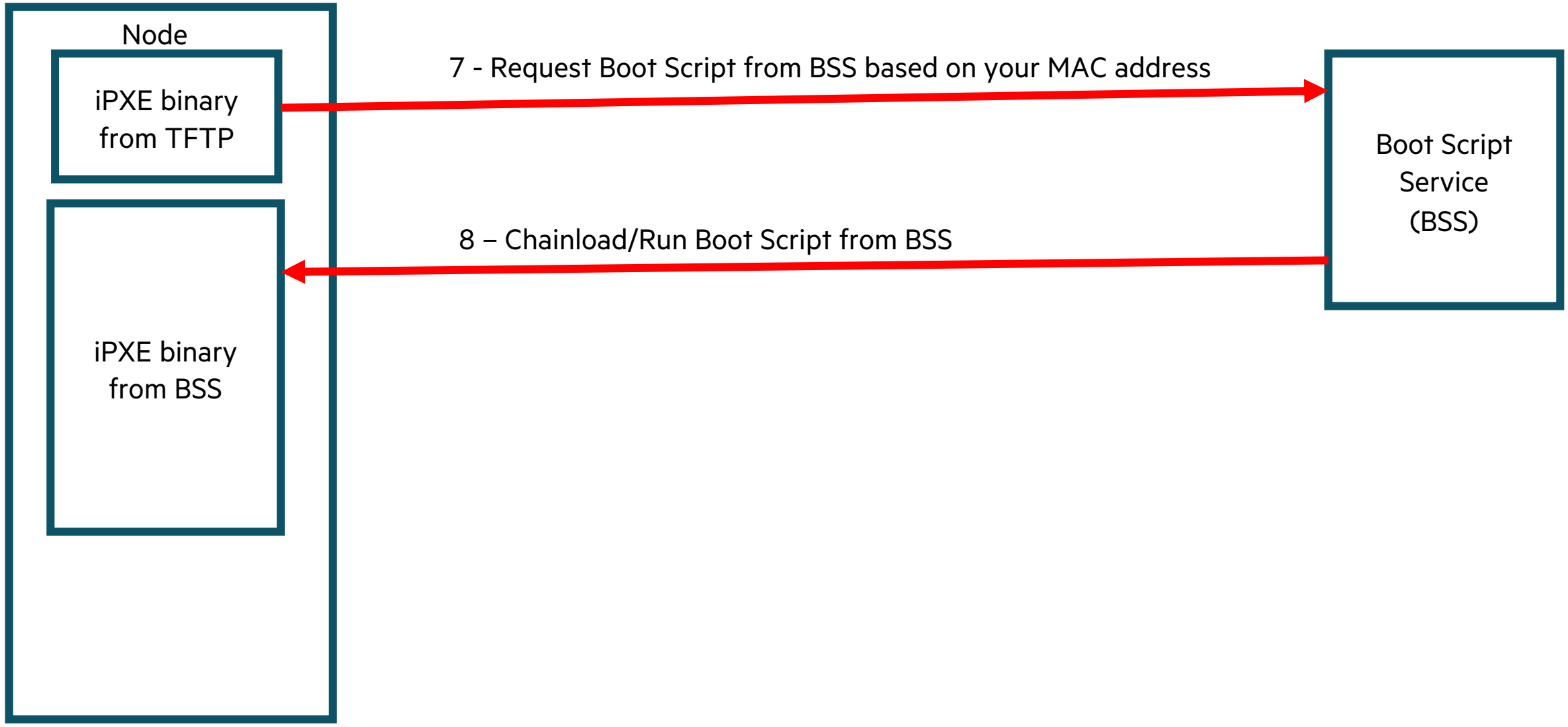
iPXE binary issues DHCP request



CONSOLE

```
2023-07-06T09:40:46.72510 Configuring (net1 00:40:a6:85:91:ed)..... ok
2023-07-06T09:40:46.72517 net1 IPv4 lease: 10.100.0.40 mac:
00:40:a6:85:91:ed
```

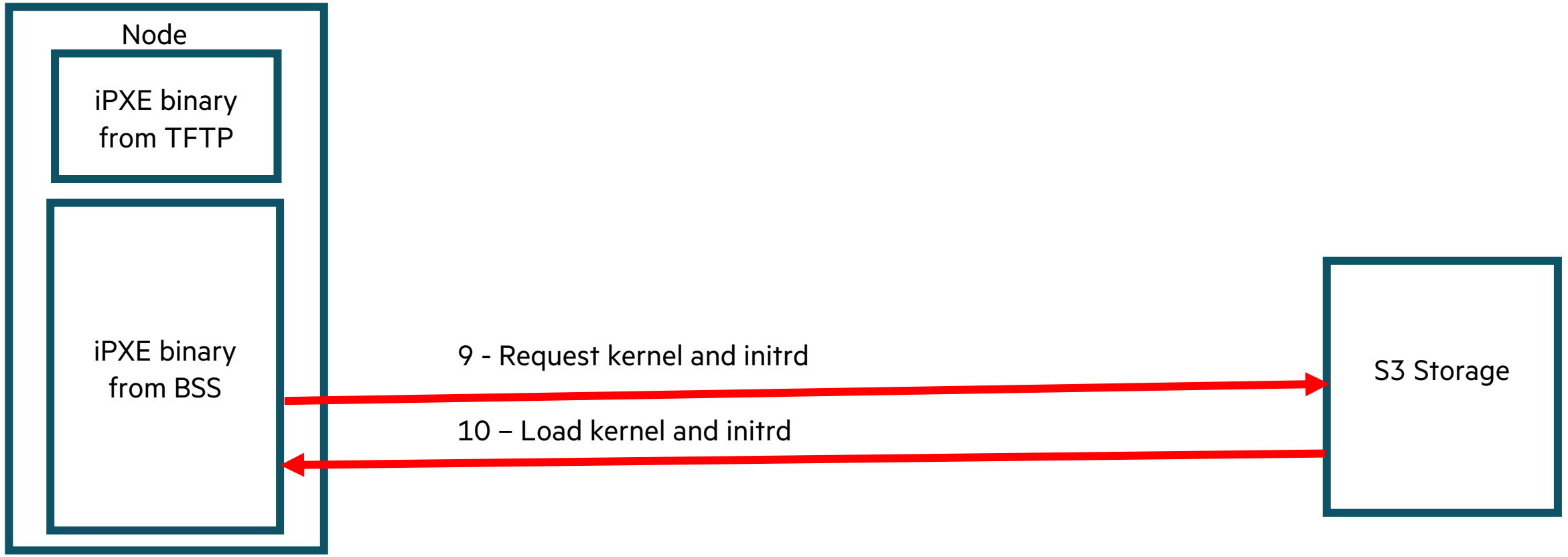
Chainload Boot Script



Chainload Boot Script – Console

```
2023-07-06T09:40:13.96317 Chaining to BSS; trying interfaces net2, net0, net1,
net3, net4, net5...
2023-07-06T09:40:13.96318 Chain attempt 1 of 1024

2023-07-06T09:40:46.72520 https://api-gw-service-
nmn.local/apis/bss/boot/v1/bootscript...X509 chain 0x90f96248 added X509 0x90f973e8
"system.domain.name"
2023-07-06T09:40:46.72521 X509 chain 0x90f96248 added X509 0x91013460 "Platform CA
- L1 (f1af0273-386e-46d9-a194-2c8ca37a87fb)"
2023-07-06T09:40:46.72523 X509 chain 0x90f96248 added X509 0x910135a0 "Platform CA
(f1af0273-386e-46d9-a194-2c8ca37a87fb)"
2023-07-06T09:40:46.72524 EFITIME is 2023-07-06 09:40:45
2023-07-06T09:40:46.72525 X509 0x91013460 "Platform CA - L1 (f1af0273-386e-46d9-
a194-2c8ca37a87fb)" is a root certificate
2023-07-06T09:40:46.72526 X509 0x90f973e8 "system.domain.name" successfully
validated using issuer 0x91013460 "Platform CA - L1 (f1af0273-386e-46d9-a194-
2c8ca37a87fb)"
```

Request and load Kernel and initrd



CONSOLE

```
2023-07-06T09:40:46.72562 http://rgw-vip.nmn/boot-images/4bceedc2-4671-448e-bc32-f784c66b3cfe/kernel... ok
2023-07-06T09:40:46.72573 http://rgw-vip.nmn/boot-images/4bceedc2-4671-448e-bc32-f784c66b3cfe/initrd... ok
```

Node Executes the Kernel – Console

```
2023-07-06T09:40:47.95290 Linux version 5.14.21-150400.24.11_12.0.57-
cray_shasta_c (abuild@cflobswrk02) (gcc (SUSE Linux) 7.5.0, GNU ld (GNU
Binutils; SUSE Linux Enterprise 15) 2.37.20211103-150100.7.37) #1 SMP Sun
Dec 11 15:40:04 UTC 2022 (4ac4a0d)
2023-07-06T09:40:47.95349 Command line: kernel initrd=initrd
console=ttyS0,115200 bad_page=panic crashkernel=512M hugepagelist=2m-2g
intel_iommu=off intel_pstate=disable iommu.passthrough=on
modprobe.blacklist=amdgpu numa_interleave_omit=headless oops=panic
pageblock_order=14 rd.neednet=1 rd.retry=10 rd.shell
systemd.unified_cgroup_hierarchy=1 cxi_core.enable_fgfc=1 ip=dhcp
clocksource=tsc tsc=reliable spire_join_token=2f97e84e-6ca2-4f7b-8bcf-
59e6483b2040 root=craycps-s3:s3://boot-images/4bceedc2-4671-448e-bc32-
f784c66b3cfe/rootfs:98830d5f7ecf60bea0fe5a84eb8be871-231:dvs:api-gw-
service-nmn.local:300:nmn0:0 nmd_data=url=s3://boot-images/4bceedc2-4671-
448e-bc32-f784c66b3cfe/rootfs,etag=98830d5f7ecf60bea0fe5a84eb8be871-231
bos_session_id=96942ef7-e061-433c-886d-4dab601c8fdd xname=x1001c5s0b0n1
nid=1417 bss_referral_token=051168d1-abda-431b-a135-adf1b4906a0c
ds=nocloud-net;s=http://10.92.100.81:8888/
```

Dracut – Whole bunch of Stuff happens in dracut

- Hop into Initrd – run Dracut scripts
- Dracut-initqueue configures NICs via DHCP (this is the third time we've configured DHCP, if you've been counting)
  - NMN NICs
  - HSN NICs
- Chronyd is started
- Spire-agent is started
- Configure CPS, DVS, and LNET
- Move to Dracut-pre-mount phase
- We do the switch root to the root file system

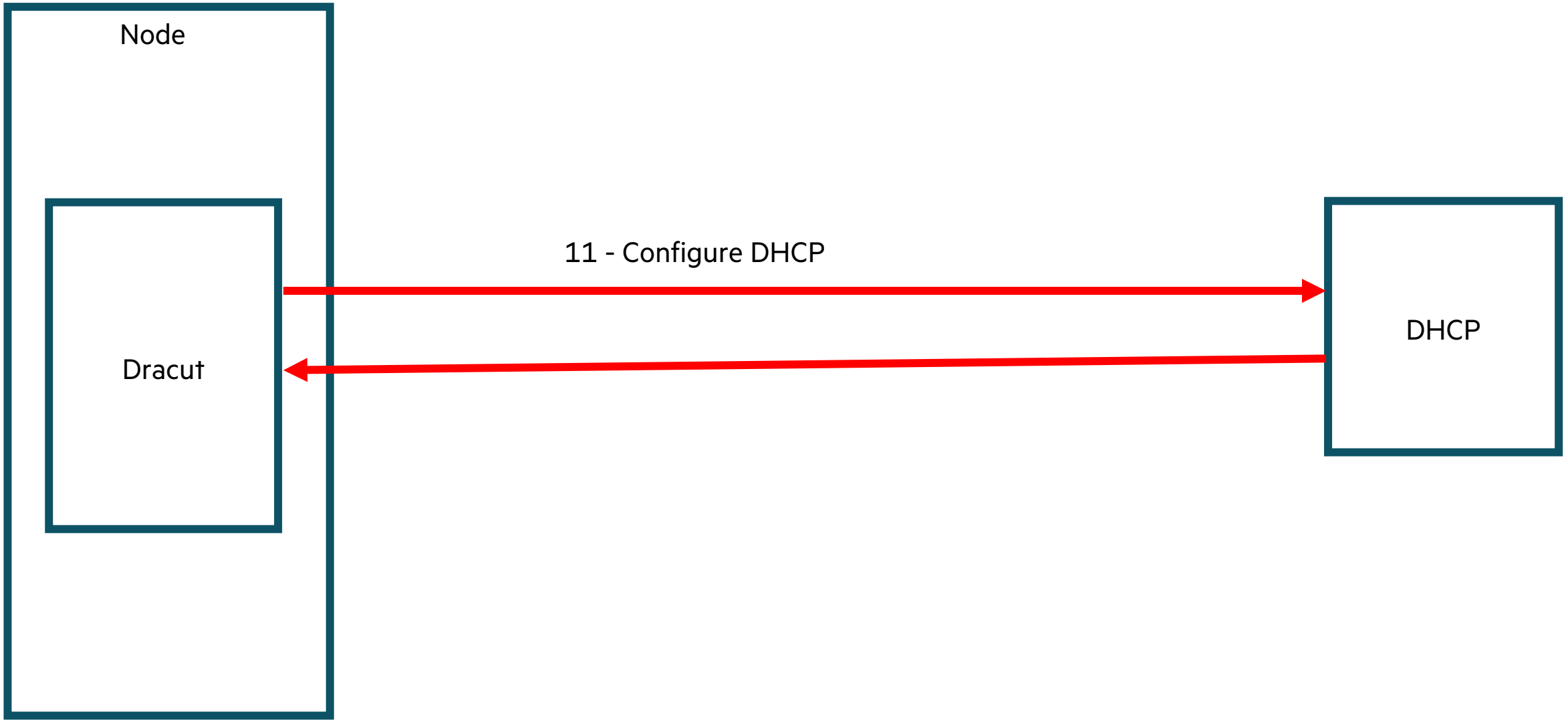
Node runs dracut scripts in initrd – Console

```
2023-07-06T09:41:03.84340 Run /init as init process
2023-07-06T09:41:03.84342 autofs4: module verification failed: signature
and/or required key missing - tainting kernel
2023-07-06T09:41:03.84343 systemd[1]: Inserted module 'autofs4'
2023-07-06T09:41:03.84344 systemd[1]: systemd 249.11+suse.129.g17d488c53a
running in system mode (+PAM +AUDIT +SELINUX +APPARMOR -IMA -SMACK
+SECCOMP +GCRYPT +GNUTLS +OPENSSL +ACL +BLKID +CURL +ELFUTILS +FIDO2 +IDN2
-IDN +IPTC +KMOD +LIBCRYPTSETUP +LIBFDISK +PCRE2 +PWQUALITY +P11KIT
+QRENCODE +TPM2 +BZIP2 +LZ4 +XZ +ZLIB +ZSTD -XKBCOMMON +UTMP +SYSVINIT
default-hierarchy=hybrid)
2023-07-06T09:41:03.84364 systemd[1]: Detected architecture x86-64.
2023-07-06T09:41:03.84365 systemd[1]: Running in initial RAM disk.
2023-07-06T09:41:03.84367
2023-07-06T09:41:03.84368 Welcome to SUSE Linux Enterprise Server 15 SP4
dracut-055+suse.279.g3b3c36b2-150400.3.5.1 (Initramfs) 2023-07-
06T09:41:03.84369
2023-07-06T09:41:03.84370 systemd[1]: No hostname configured, using
default hostname. 2023-07-06T09:41:03.84371 systemd[1]: Hostname set to
<localhost>.
```

Node Dracut initqueue – Console

```
2023-07-06T09:41:14.03337  Starting dracut initqueue hook..
```

Dracut Diagram



Dracut Configure DHCP NMN NICs – Console

```
2023-07-06T09:41:14.03338 dracut-initqueue[2078]: INFO: Checking interface net0...
2023-07-06T09:41:14.03339 dracut-initqueue[2078]: INFO: Attempting to configure interface net0 2023-07-06T09:41:14.03340 8021q: adding VLAN 0 to HW filter on device net0
2023-07-06T09:41:14.03341 igb 0000:41:00.0 net0: igb: net0 NIC Link is Up 1000 Mbps Full Duplex, Flow Control: RX/TX 2023-07-06T09:41:14.03342 IPv6: ADDRCONF(NETDEV_CHANGE): net0: link becomes ready 2023-07-06T09:41:14.03344 dracut-initqueue[2145]: wicked: net0: Request to acquire DHCPv4 lease with UUID 318ca664-e67c-0600-6108-000001000000
2023-07-06T09:41:14.03353 dracut-initqueue[2145]: wicked: net0: Committed DHCPv4 lease with address 10.120.5.17 (lease time 3600 sec, renew in 1800 sec, rebind in 3150 sec)
2023-07-06T09:41:14.03354 dracut-initqueue[2078]: INFO: Interface net0 is now configured
2023-07-06T09:41:14.03356 dracut-initqueue[2078]: INFO: There are 1 of 1 interfaces configured
```

Dracut Rename HSN NICs– Console

```
2023-07-06T09:41:14.03357 dracut-initqueue[2217]: INFO: The system has no HSN device
drivers 2023-07-06T09:41:14.03358 sbl sbl: []: new instance (0): 1 ports x 4 serdes
2023-07-06T09:41:14.03362 sbl sbl: cxi0[]: serdes fw loaded
2023-07-06T09:41:14.03363 cxi_core 0000:21:00.0: cxi0[] CXI_EVENT_LINK_DOWN
2023-07-06T09:41:14.03365 Added user device for cxi0
2023-07-06T09:41:14.03366 cxi_core 0000:21:00.0: cxi0[eth0]: Added Ethernet device
2023-07-06T09:41:14.03367 KFI: Kernel Open Fabric Interface framework loaded.
2023-07-06T09:41:14.03368 KFI: A kcxi 1.0 provider is successfully loaded.
2023-07-06T09:41:14.03369 kfi_cxi: kCXI Device Index (0) Fabric (1): Device added
2023-07-06T09:41:14.03370 [ OK  Created slice Slicecxi_core 0000:21:00.0 tmp0: renamed
from eth0
2023-07-06T09:41:14.03372 /system/cxi_rh
2023-07-06T09:41:14.03373 Starting CXI Retry Handler on cxi0..
2023-07-06T09:41:14.03374 fuse: init (API version 7.34)
2023-07-06T09:41:14.03379 [ OK  Started CXI Retry Handler on cxi0sbl sbl: cxi0[tmp0]:
cxi0 eth if name changed to tmp0
2023-07-06T09:41:14.03380 0m.
2023-07-06T09:41:14.03382 dracut-initqueue[2226]: Warning: Interface tmp0 is not UP
(flags <BROADCAST,MULTICAST>)
```

Chronyd is started – Console

```
2023-07-06T09:41:24.03395 dracut-initqueue[2493]:
2023-07-06T09:41:22Z chronyd version 4.1 starting (+CMDMON +NTP +REFCLOCK +RTC
+PRIVDROP +SCFILTER +SIGND +ASYNCDNS +NTS +SECHASH +IPV6 -DEBUG)
2023-07-06T09:41:34.03406 dracut-initqueue[2493]:
2023-07-06T09:41:27Z System clock wrong by 0.395050 seconds (step)
2023-07-06T09:41:34.03410 dracut-initqueue[2493]:
2023-07-06T09:41:27Z chronyd exiting
2023-07-06T09:41:34.03412 dracut-initqueue[2493]:
2023-07-06T09:41:27Z Could not open /var/lib/chrony/drift.tmp : No such file or
directory
```

Spire-agent is started – Console

```
2023-07-06T09:41:34.03413 dracut-initqueue[2492]: Warning: Spire-agent
healthcheck passed
```

Configure CPS, DVS, and LNET – Console

```
2023-07-06T09:41:34.03414 dracut-initqueue[2598]: cps: All requested interfaces are UP, proceeding.
2023-07-06T09:41:34.03415 LNet: HW NUMA nodes: 8, HW CPU cores: 256, npartitions: 8
2023-07-06T09:41:34.03417 Key type ._llcrypt registered
2023-07-06T09:41:34.03418 Key type .llcrypt registered
2023-07-06T09:41:34.03419 dracut-initqueue[2610]: LNet: loaded lnet module.
2023-07-06T09:41:34.03420 LNet: 2630:0:(config.c:1535:lnet_inet_enumerate()) lnet: Ignoring interface tmp0: it's down
2023-07-06T09:41:34.03421 LNet: Added LNI 10.120.5.17@tcp99 [8/512/0/180]
2023-07-06T09:41:34.03423 LNet: Accept secure, port 988
2023-07-06T09:41:34.03424 dracut-initqueue[2610]: LNet: lnet module configured.
2023-07-06T09:41:34.03425 katlas: init_module: katlas loaded, currently disabled
2023-07-06T09:41:34.03426 dracut-initqueue[2608]: DVS: lnet setup completed.
```

Configure DVS (Continued) – Console

```
2023-07-06T09:41:34.03441 DVS debugfs: Revision: fcb083ac Built: Dec 11 2022 @ 16:16:24
against LNet 2.15.0.4 2023-07-06T09:41:34.03443 dracut-initqueue[2608]: DVS: loaded
dvsproc module.
2023-07-06T09:41:34.03444 DVS: successfully added 1 new nodes into map.
2023-07-06T09:41:41.39778 DVS: successfully added 35 new nodes into map.
2023-07-06T09:41:41.39781 dracut-initqueue[2608]: DVS: node map generated.
2023-07-06T09:41:41.39783 DVS: message size checks complete.
2023-07-06T09:41:41.39784 dvs_thread_generator: Watching pool DVS-IPC_msg (id 0)
2023-07-06T09:41:41.39792 dracut-initqueue[2608]: DVS: loaded dvs module.
2023-07-06T09:41:41.39793 dracut-initqueue[2606]: mount is: /opt/cray/cps-
utils/bin/cpsmount.sh -a api-gw-service-nmn.local -t dvs -T 300 -i nmn0 -e
98830d5f7ecf60bea0fe5a84eb8be871-231 s3://boot-images/4bceedc2-4671-448e-bc32-
f784c66b3cfe/rootfs /tmp/cps
2023-07-06T09:41:41.39794 dracut-initqueue[3283]: 2023/07/06 09:41:35 cpsmount_helper
Version: 0.14.2
2023-07-06T09:41:41.39795 dracut-initqueue[3283]: 2023/07/06 09:41:35 18 dvs servers
[10.252.1.16@tcp99 10.252.1.17@tcp99 10.252.1.18@tcp99 10.252.1.19@tcp99
10.252.1.20@tcp99 10.252.1.21@tcp99 10.252.1.22@tcp99 10.252.1.23@tcp99
10.252.1.25@tcp99 10.252.1.26@tcp99 10.252.1.27@tcp99 10.252.1.28@tcp99
10.252.1.29@tcp99 10.252.1.30@tcp99 10.252.1.31@tcp99 10.252.1.32@tcp99
10.252.1.33@tcp99 10.252.1.34@tcp99]
2023-07-06T09:41:41.39804 dracut-initqueue[2606]: cps: remote-path /var/lib/cps-
local/boot-images/4bceedc2-4671-448e-bc32-f784c66b3cfe
2023-07-06T09:41:41.39806 [ OK  Finished dracut initqueue hook
```

Mount the rootfs into /tmp/cps

- Console line:
  - mount is

```
/opt/cray/cps-utils/bin/cpsmount.sh -a api-gw-service-nmn.local -t dvs -T 300 -i nmn0 -e
98830d5f7ecf60bea0fe5a84eb8be871-231 s3://boot-images/4bceedc2-4671-448e-bc32-f784c66b3cfe/rootfs
/tmp/cps
```

DVS – Mount rootfs squashfs

- DVS projects the rootfs from S3 to the node as a Read-Only projection
  - Console line where Dracut mounts up the DVS projected squashfs

```
dracut-pre-mount[3315]: mount is: mount -t squashfs -o loop
/tmp/cps/rootfs /new_root/lower
```

- It is then overlayed with a write-able layer

Dracut moves to the dracut-pre-mount phase – Console

```
2023-07-06T09:41:41.39807 [ OK  Reached target Preparation for Remote File Systems
2023-07-06T09:41:41.39808 [ OK  Reached target Remote Encrypted Volumes
2023-07-06T09:41:41.39809 [ OK  Reached target Remote File Systems
2023-07-06T09:41:41.39810 Starting dracut pre-mount hook..
2023-07-06T09:41:41.39812 loop: module loaded
2023-07-06T09:41:41.39813 dracut-pre-mount[3315]: mount is: mount -t squashfs -o
loop /tmp/cps/rootfs /new_root/lower
2023-07-06T09:41:41.39814 loop0: detected capacity change from 0 to 3776448
2023-07-06T09:41:41.39815 dracut-pre-mount[3342]: info: No menu item 'Found
Overlay-Preload Configuration' in node '(dir)Top'
2023-07-06T09:41:41.39816 [ OK  Finished dracut pre-mount hook
2023-07-06T09:41:41.39817 [ OK  Reached target Initrd Root File System
2023-07-06T09:41:41.39819 Starting Reload Configuration from the Real Root..
2023-07-06T09:41:41.39824 [ OK  Finished Reload Configuration from the Real Root
```

Switch Root – Console

```
2023-07-06T09:41:42.34732 Starting Switch Root..
2023-07-06T09:41:42.34733 systemd-journald[1665]: Received SIGTERM from
PID 1 (systemd).
2023-07-06T09:41:42.34734 systemd[1]: systemd 249.11+suse.129.g17d488c53a
running in system mode (+PAM +AUDIT +SELINUX +APPARMOR -IMA -SMACK
+SECCOMP +GCRYPT +GNUTLS +OPENSSL +ACL +BLKID +CURL +ELFUTILS +FIDO2 +IDN2
-IDN +IPTC +KMOD +LIBCRYPTSETUP +LIBFDISK +PCRE2 +PWQUALITY +P11KIT
+QRENCODE +TPM2 +BZIP2 +LZ4 +XZ +ZLIB +ZSTD -XKBCOMMON +UTMP +SYSVINIT
default-hierarchy=hybrid)
2023-07-06T09:41:42.34736 systemd[1]: Detected architecture x86-64.
2023-07-06T09:41:42.34737
2023-07-06T09:41:42.34738 Welcome to SUSE Linux Enterprise Server 15 SP4
```

Node boots up post dracut – Console

- Node starts heartbeating
  - The node will send a heartbeat message every 30 seconds which will be processed by cray-hbtd pod

```
2023-07-06T09:42:11.97802 [ OK  Started heartbeat
```

- CFS-state-reporter checks in

```
2023-07-06T09:42:14.03671 Starting cfs-state-reporter Configuration level of the system..
2023-07-06T09:42:24.03939 [ OK  Finished cfs-state-reporter Configuration level of the system
```

- BOS-reporter checks in
  - In CSM-1.3 and later, the bos-state-reporter reports on the state of the booted node

```
2023-07-06T09:42:14.03670 [ OK  Started bos-status-reporter, periodically to the BOS API
```

- Node displays the login prompt

```
2023-07-06T09:46:14.24004 nid001417 login:
```

When Things Go Wrong

- Disable nodes which have obvious hardware issues
- Debugging Why a Node is OFF
- Debugging Why a Node is ON but fails to boot
- Troubleshooting failed boot
- Troubleshooting slow boot

Disable nodes which have obvious hardware issues

- In the Hardware State Manager (HSM) to prevent BOS from attempting to manage these nodes

```
ncn# cray hsm state components enabled update --enabled false x1000c5s4b0n0
ncn# cray hsm state components bulkEnabled update --enabled false --component-ids
x1010c0s1b0n1,x1010c0s2b1n0,x1010c1s2b1n1
```

- In the workload manager
  - Slurm:

```
ncn# scontrol update nodename=node1 state=down
```

  - PBS Pro:

```
ncn# pbsnodes -o node1
```

Debugging Why a Node is OFF

- Check console log for node x1000c3s4b0n1

```
ncn# kubectl -n services exec cray-console-node-0 -c cray-console-node -- tail -f
/var/log/conman/console.x1000c3s4b0n1
```

- Check BMC log for the node x1000c3s4b0n1

```
ncn# ssh x1000c3s4b0
ncn# cat /var/log/messages
ncn# cat /var/log/powerfault_up.Node1
ncn# cat /var/log/powerfault_dn.Node1
ncn# dmesg
```

  - Olympus nodes: (example x1000c0s0b0n1)

```
ncn# ssh x1000c0s0b0
x1000c0s0b0> egrep "\(partially powered up\)|Stopped at PS|already fully powered up"
/var/log/powerfault_up.Node1
```

  - Refer to hardware service team if these message patterns are found

- Check sat hardware history

```
ncn# sat hwhist --xname x9000c3s4e1
+-------------+-----------------------------------------+-------------------------------+-----------+
| xname       | FRUID                                   | Timestamp                     | EventType |
+-------------+-----------------------------------------+-------------------------------+-----------+
| x1000c3s4e1 | NodeEnclosure.HPE.P52291003C.LJ23210005 | 2024-01-16T19:35:10.485738Z | Detected  |
| x1000c3s4e1 | NodeEnclosure.HPE.P52291003C.LJ23210005 | 2024-02-14T15:46:50.886112Z | Removed   |
+-------------+-----------------------------------------+-------------------------------+-----------+
```

- For any nodes in the OFF state, check the power logs on their BMC (nodecontroller) for power up faults

Debugging Why a Node is ON but fails to boot

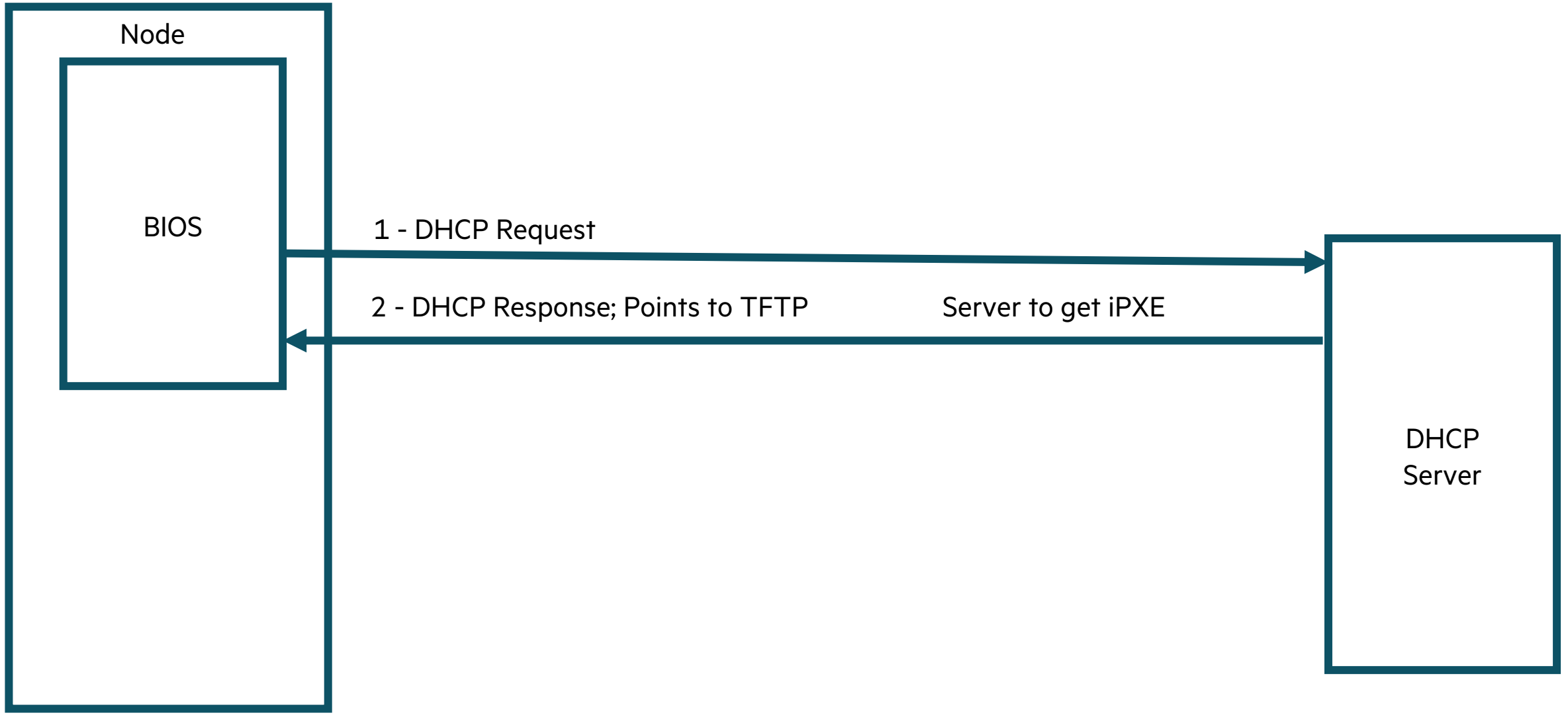
Hardware Failure during BIOS

- If hardware initialization problems are found by the BIOS, the node will drop into the UEFI shell and await manual input for the next step

Failures During PXEBoot

- Failure to receive a DHCPOFFER packet

Failure to receive a DHCPOFFER packet



DHCP Failure to receive a DHCPOFFER packet

- DHCPDISCOVER (BROADCAST) is sent from the node (client) to a DHCP server
- In the Console, it looks like this

```
2023-06-17 13:29:51 >>Start PXE over IPv4 on MAC: 00-40-A6-84-7B-
B7InstallProtocolInterface: EfiPxeBaseCodeCallback 86293190
```

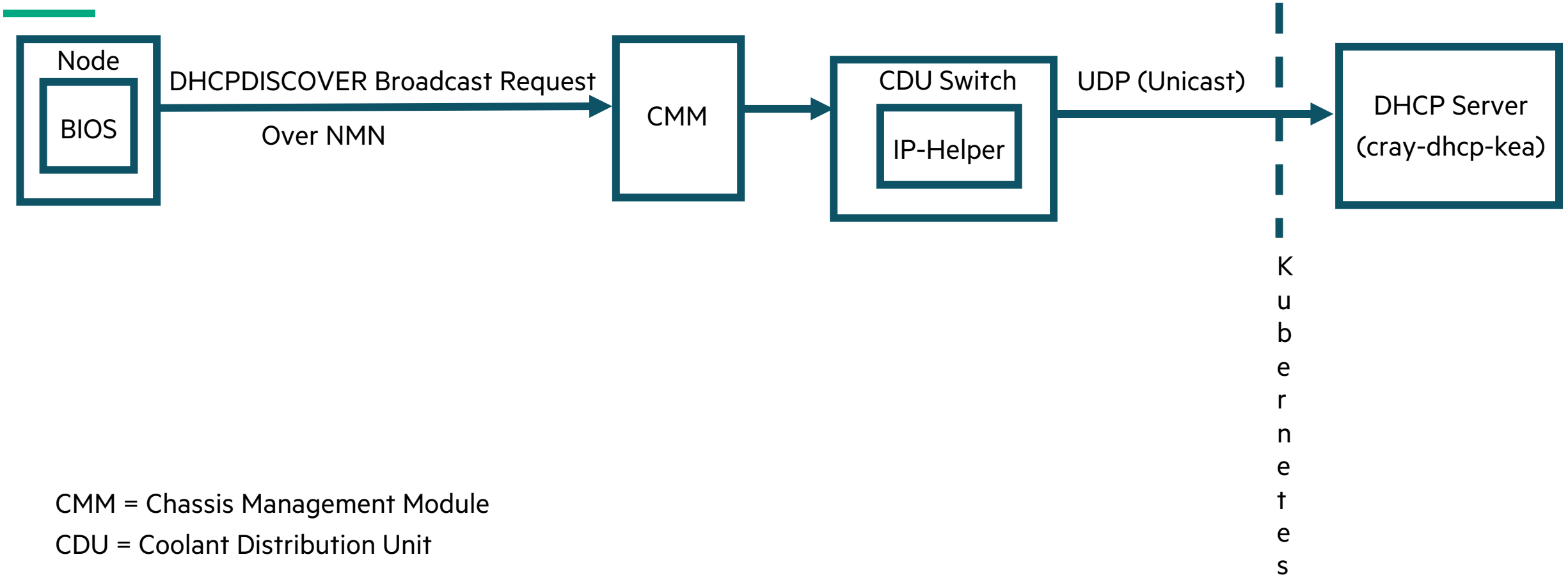
- If the node fails to receive a DHCPOFFER packet after the fourth try, it will wait about 35 seconds and then report this error message on the node console

```
2023-06-17 13:30:51 PXE-E18: Server response timeout.
```

- Total elapsed time to send 4 DHCPDISCOVER packets without receiving a DHCPOFFER and reporting PXE-E18 error is about 60 seconds for each NIC in the BIOS BootOrder

DHCPDISCOVER Failure – Tracking Broadcast request

DHCPDISCOVER Failure – Tracking the Broadcast request



CMM = Chassis Management Module

CDU = Coolant Distribution Unit

Checking the Chassis Management Module (CMM) for log messages

- If the node's xname is x9000c1s0b1n1
- The CMM is x9000c1b0

```
ncn# ssh x9000c1b0
x9000c1:> cat /var/log/messages
```

- The CDU information flows through the CMM

The DHCPDISCOVER request enters Kubernetes (k8s)

- The packet is forwarded to the k8s load balancer, which is the Linux kernel's IP Virtual Server (IPVS)
    - IPVS knows that DHCP packets destined to NMN DCHP 10.92.100.222:67 map to a pod(s) in Kubernetes (let's say 10.32.0.31) to an unprivileged (>1024) port 6067
    - IPVS creates a Linux connection tracking table entry (conntrack) to map the source (IP:PORT) to the private pod destination (10.32.0.31:6067)
        - This mapping is Destination NATting (DNAT) and the conntrack entry is used as a lookup to reverse the DNAT when the packet comes back from cray-dhcp-kea
    - The client request DHCP packet enters "the mesh" and is routed to cray-dhcp-kea
- cray-dhcp-kea pod running on a Kubernetes worker node receives DHCPDISCOVER (UNICAST) and logs it
- The cray-dhcp-kea pod issues an offer, and it reverses course all the way back to the requesting node

DHCP Problems – Debugging from the server side

Find the node's MAC address

```
ncn# cray bss hosts list --format json --name x3000c0s17b0n0 | jq
.[].MAC
[
     "00:40:a6:85:5f:0b"
]
```

See the DHCP activity related to a particular node

- Search using the MAC address "00:40:a6:85:5f:0b"

```
ncn# kubectl -n services logs cray-dhcp-kea-65b9c9cd4b-899qq -c cray-dhcp-kea --tail=500000 | grep "00:40:a6:85:5f:0b"
2024-04-19 14:49:36.968 INFO  [kea-dhcp4.leases/16.140291324160824] DHCP4_LEASE_ADVERT [hwtype=1 00:40:a6:85:5f:0b],
cid=[no info], tid=0x97c968eb: lease 10.120.0.10 will be advertised
2024-04-19 14:49:40.308 INFO  [kea-dhcp4.leases/16.140291323587384] DHCP4_LEASE_ALLOC [hwtype=1 00:40:a6:85:5f:0b], cid=[no
info], tid=0x97c968eb: lease 10.120.0.10 has been allocated for 3600 seconds
2024-04-19 14:50:30.034 INFO  [kea-dhcp4.leases/16.140291323300664] DHCP4_LEASE_ADVERT [hwtype=1 00:40:a6:85:5f:0b],
cid=[01:00:40:a6:85:5f:0b], tid=0xec5f775c: lease 10.120.0.10 will be advertised
2024-04-19 14:50:31.775 INFO  [kea-dhcp4.leases/16.140291323157304] DHCP4_LEASE_ADVERT [hwtype=1 00:40:a6:85:5f:0b],
cid=[01:00:40:a6:85:5f:0b], tid=0xec5f775c: lease 10.120.0.10 will be advertised
2024-04-19 14:50:35.290 INFO  [kea-dhcp4.leases/16.140291323013944] DHCP4_LEASE_ALLOC [hwtype=1 00:40:a6:85:5f:0b],
cid=[01:00:40:a6:85:5f:0b], tid=0xec5f775c: lease 10.120.0.10 has been allocated for 3600 seconds
2024-04-19 14:51:40.030 INFO  [kea-dhcp4.leases/16.140291324447544] DHCP4_LEASE_ADVERT [hwtype=1 00:40:a6:85:5f:0b],
cid=[ff:a6:85:5f:0b:00:01:00:01:2d:b5:41:7b:00:40:a6:85:5f:0b], tid=0x1741f1d8: lease 10.120.0.10 will be advertised
2024-04-19 14:51:40.031 INFO  [kea-dhcp4.leases/16.140291324304184] DHCP4_LEASE_ALLOC [hwtype=1 00:40:a6:85:5f:0b],
cid=[ff:a6:85:5f:0b:00:01:00:01:2d:b5:41:7b:00:40:a6:85:5f:0b], tid=0x1741f1d8: lease 10.120.0.10 has been allocated for
3600 seconds
2024-04-19 14:54:00.595 INFO  [kea-dhcp4.leases/16.140291324447544] DHCP4_LEASE_ADVERT [hwtype=1 00:40:a6:85:5f:0b],
cid=[ff:a6:85:5f:0b:00:01:00:01:2d:b5:42:08:00:40:a6:85:5f:0b], tid=0x3dd58bc5: lease 10.120.0.10 will be advertised
2024-04-19 14:54:00.596 INFO  [kea-dhcp4.leases/16.140291324304184] DHCP4_LEASE_ALLOC [hwtype=1 00:40:a6:85:5f:0b],
cid=[ff:a6:85:5f:0b:00:01:00:01:2d:b5:42:08:00:40:a6:85:5f:0b], tid=0x3dd58bc5: lease 10.120.0.10 has been allocated for
3600 seconds
2024-04-19 14:56:43.570 INFO  [kea-dhcp4.leases/16.140291324017464] DHCP4_INIT_REBOOT [hwtype=1 00:40:a6:85:5f:0b],
cid=[ff:a6:85:5f:0b:00:01:00:01:2d:b5:42:08:00:40:a6:85:5f:0b], tid=0x2830e8e2: client is in INIT-REBOOT state and requests
address 10.120.0.10
2024-04-19 14:56:43.570 INFO  [kea-dhcp4.leases/16.140291324017464] DHCP4_LEASE_ALLOC [hwtype=1 00:40:a6:85:5f:0b],
cid=[ff:a6:85:5f:0b:00:01:00:01:2d:b5:42:08:00:40:a6:85:5f:0b], tid=0x2830e8e2: lease 10.120.0.10 has been allocated for
3600 seconds
```

See just the DHCPREQUEST in the kea pod

```
ncn# kubectl -n services logs <kea-pod-name> -c cray-dhcp-kea --tail=<large number> | grep requests
2024-04-19 14:56:43.570 INFO  [kea-dhcp4.leases/16.140291324017464] DHCP4_INIT_REBOOT [hwtype=1 00:40:a6:85:5f:0b], cid=[ff:a6:85:5f:0b:00:01:00:01:2d:b5:42:08:00:40:a6:85:5f:0b], tid=0x2830e8e2: client is in INIT-REBOOT state and requests address 10.120.0.10
```

Get Token to request DHCP information from Kea

```
ncn# export TOKEN=$(curl -s -k -S -d grant_type=client_credentials
-d client_id=admin-client \ -d client_secret=`kubectl get secrets
admin-client-auth -o jsonpath='{.data.client-secret}' | base64 -d`
\ https://api-gw-service-
nmn.local/keycloak/realms/shasta/protocol/openid-connect/token | jq
-r '.access_token')
```

View DHCP Leases in Kea

- ncn# **curl -s -k -H "Authorization: Bearer ${TOKEN}" -X POST -H "Content-Type: application/json" \ -d '{ "command": "lease4-get-all", "service": ["dhcp4"] }' https://api-gw-service-nmn.local/apis/dhcp-kea | jq**


```
{
        "cltt": 1713548777,
        "fqdn-fwd": false,
        "fqdn-rev": false,
        "hostname": "x3000c0s17b0",
        "hw-address": "94:40:c9:37:e3:02",
        "ip-address": "10.254.1.58",
        "state": 0,
        "subnet-id": 4,
        "valid-lft": 3600
}
```


DHCP Kea Determine the hostname or MAC address from an IP address.

```
ncn# IP=<IP_address>
ncn# curl -s -k -H "Authorization: Bearer ${TOKEN}" -X POST -H
"Content-Type: application/json" -d "{ \"command\": \"lease4-get\",
\"service\": [ \"dhcp4\" ], \"arguments\": { \"ip-address\":
\"${IP}\" } }" \ https://api-gw-service-nmn.local/apis/dhcp-kea |
jq ".[].arguments.leases[] | select(.\"ip-address\"==\"${IP}\")"
```

DHCP Kea Determine the hostname or IP address from the MAC address

```
ncn# MAC=<MAC_address>
ncn# curl -H "Authorization: Bearer ${TOKEN}" -X POST -H "Content-
Type: application/json" -d '{ "command": "lease4-get-all",
"service": [ "dhcp4" ] }' https://api-gw-service-
nmn.local/apis/dhcp-kea | jq ".[].arguments.leases[] |
select(.\"hw-address\"==\"${MAC}\")"
```

DHCP Kea Determine MAC or IP address from the hostname.

```
ncn# HNAME=<hostname>
ncn# curl -H "Authorization: Bearer ${TOKEN}" -X POST -H "Content-
Type: application/json" -d '{ "command": "lease4-get-all",
"service": [ "dhcp4" ] }' https://api-gw-service-
nmn.local/apis/dhcp-kea |  jq ".[].arguments.leases[] |
select(.\"hostname\"==\"${HNAME}\")"
```

Problem: DHCP Duplicate IP Addresses

- A sign of a duplicate IP address is seeing a DECLINE message from the client to the server

```
10.40.0.0.337 > 10.42.0.58.67: BOOTP/DHCP, Request from b4:2e:99:be:1a:d3, length
301, hops 1, xid 0x9d1210d, Flags [none]
```

- Example output:

```
Gateway-IP 10.252.0.2
Client-Ethernet-Address b4:2e:99:be:1a:d3
Vendor-rfc1048 Extensions
  Magic Cookie 0x63825363
  DHCP-Message Option 53, length 1: Decline
  Client-ID Option 61, length 19: hardware-type 255,
  99:be:1a:d3:00:01:00:01:26:c8:55:c3:b4:2e:99:be:1a:d3
  Server-ID Option 54, length 4: 10.42.0.58
  Requested-IP Option 50, length 4: 10.252.0.26
  Agent-Information Option 82, length 22:
    Circuit-ID SubOption 1, length 20: vlan2-ethernet1/1/12
```

Problem: Numerous DHCP Messages Declined

```
IPv6: ADDRCONF(NETDEV_CHANGE): eth0: link becomes ready
dracut-initqueue[1902]: wicked: eth0: Request to acquire DHCPv4 lease with UUID
13b0675f-12cb-0a00-2f0a-000001000000
dracut-initqueue[1902]: wicked: eth0: Declining DHCPv4 lease with address 10.252.0.51
dracut-initqueue[1902]: wicked: eth0: Declining DHCPv4 lease with address 10.252.0.53
dracut-initqueue[1902]: wicked: eth0: Declining DHCPv4 lease with address 10.252.0.54
dracut-initqueue[1902]: wicked: eth0: Declining DHCPv4 lease with address 10.252.0.55
dracut-initqueue[1902]: wicked: eth0: Declining DHCPv4 lease with address 10.252.0.56
dracut-initqueue[1902]: wicked: eth0: Declining DHCPv4 lease with address 10.252.0.57
dracut-initqueue[1902]: wicked: eth0: Declining DHCPv4 lease with address 10.252.0.58
dracut-initqueue[1902]: wicked: eth0: Declining DHCPv4 lease with address 10.252.0.59
dracut-initqueue[1902]: wicked: eth0: Declining DHCPv4 lease with address 10.252.0.60
dracut-initqueue[1902]: wicked: eth0: Declining DHCPv4 lease with address 10.252.0.51
dracut-initqueue[1902]: wicked: eth0: Declining DHCPv4 lease with address 10.252.0.53
dracut-initqueue[1902]: wicked: eth0: Declining DHCPv4 lease with address 10.252.0.54
dracut-initqueue[1902]: wicked: eth0: Declining DHCPv4 lease with address 10.252.0.61
dracut-initqueue[1902]: wicked: eth0: Declining DHCPv4 lease with address 10.252.0.62
```

Numerous DHCP Messages Declined (Continued)

- Indicates an IP address being allocated is already being used
- Do the following procedure to troubleshoot
- Determine the IP address that is supposed to be set for node
  - Check by MAC address (no colons).

    ncn# **curl -f -s -k -H "Authorization: Bearer ${TOKEN}" https://apa_gw_service.local/apis/smd/hsm/v2/Inventory/EthernetInterfaces/18c04d13d73c**
  - Check by xname

    ncn# **curl -f -s -k -H "Authorization: Bearer ${TOKEN}" https://api_gw_service.local/apis/smd/hsm/v2/Inventory/EthernetInterfaces?ComponentID=x3000c0s25b0n0**
  - Should return something like this


```
{ "ID": "18c04d13d73c",
"Description": "Ethernet Interface Lan1",
"MACAddress": "18:c0:4d:13:d7:3c",
"IPAddresses": [{,"IPAddress":"10.252.0.78"}],
"LastUpdate": "2020-09-20T19:46:04.811779Z",
"ComponentID": "x3000c0s25b0n0",
"Type": "Node" }
```


Numerous DHCP Messages Declined (Continued)

- Shut the node that is supposed to have this address off.
- Ping the IP address from the SMD entry to see if something is responding to it

```
ncn# ping -c 3 10.252.0.78
PING 10.252.0.78 (10.252.0.78) 56(84) bytes of data.
64 bytes from 10.252.0.78: icmp_seq=1 ttl=64 time=0.102 ms
64 bytes from 10.252.0.78: icmp_seq=2 ttl=64 time=0.229 ms
64 bytes from 10.252.0.78: icmp_seq=3 ttl=64 time=0.096 ms
--- 10.252.0.78 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2054ms rtt
min/avg/max/mdev = 0.096/0.142/0.229/0.062 ms
```

- If ping receives responses, then that confirms that a device is responding to the IP address
- In that case, the DHCP reservation for the device must be moved to another IP address

Numerous DHCP Messages Declined (Continued)

- Here is the remediation for the node that did not get the IP address
- Remove the entry using its MAC address (without colons) in the SMD Ethernet table

  ncn# **curl -f -X DELETE -s -k -H "Authorization: Bearer ${TOKEN}" https://api_gw_service.local/apis/smd/hsm/v1/Inventory/EthernetInterfaces/18c04d13d73c**
- Wait five minutes for HMS discovery to recreate the SMD Ethernet table entry
- Reboot the node and let the node get an IP address from DHCP
  - The standard discovery/DHCP/DNS process should complete in about 5 minutes
  - This will get the node to boot up until DVS is needed (if the node is using DVS)

Problems with TFTP Download

- Is there a failure from tftp server to provide iPXE binary?

Check TFTP Pod for files

```
ncn# kubectl -n services exec -it <cray-tftp-XXX-YYY> -- ls -l /var/lib/tftpboot
total 4000
-rw-r--r--    1 nobody   nobody     1001248 Mar  8 03:40 debug-ipxe.arm64.efi
-rw-r--r--    1 nobody   nobody     1040512 Mar  8 03:46 debug-ipxe.efi
-rw-r--r--    1 nobody   nobody     1006944 Mar  8 03:39 ipxe.arm64.efi
-rw-r--r--    1 nobody   nobody     1046336 Mar  8 03:46 ipxe.efi
```

Verifying iPXE binaries

- Verify the iPXE binary for an X86 node

```
ncn# kubectl -n services exec deployment/cray-ipxe-x86-64 -- file
/shared_tftp/ipxe.efi
/shared_tftp/ipxe.efi: MS-DOS executable PE32+ executable (DLL) (EFI
application) x86-64, for MS Windows
```

- Verify the iPXE binary for an ARM node

```
ncn# kubectl -n services exec deployment/cray-ipxe-aarch64 -- file
/shared_tftp/ipxe.arm64.efi
/shared_tftp/ipxe.arm64.efi: MS-DOS executable PE32+ executable (DLL)
(EFI application) Aarch64, for MS Windows
```

Invalid iPXE Binary

- If the output does not indicate an MS-DOS executable, the iPXE binary may be invalid and should be rebuilt
  - Example of an invalid iPXE binary

```
ncn# kubectl -n services exec deployment/cray-ipxe-x86-64 -- file
/shared_tftp/ipxe.efi
/shared_tftp/ipxe.efi: pxelinux loader (version 3.70 or newer)
```

Rebuilding iPXE binaries

- Rebuild the iPXE binary if required, it will take several minutes for the new binary to be built

- Rebuild the binary for an X86 node

```
ncn# kubectl -n services rollout restart deployment cray-ipxe-x86-64
deployment.apps/cray-ipxe-x86-64 restarted
```

- Rebuild the binary for an ARM node

```
ncn# kubectl -n services rollout restart deployment cray-ipxe-aarch64
deployment.apps/cray-ipxe-aarch64 restarted
```

Is there a failure from BSS to provide iPXE boot script (mainly for NCNs)

- Symptoms:

```
https://api-gw-service-nmn.local/apis/bss/boot/v1/bootscript...X509 chain
0x6d35c548 added
X509 0x6d360d68 "eniac.dev.cray.com" X509 chain 0x6d35c548 added X509 0x6d3d62e0
"Platform CA - L1 (a0b073c8-5c9c-4f89-b8a2-a44adce3cbdf)"
X509 chain 0x6d35c548 added X509 0x6d3d6420 "Platform CA (a0b073c8-5c9c-4f89-
b8a2-a44adce3cbdf)"
EFITIME is 2021-02-26 21:55:04
HTTP 0x6d35da88 status 404 Not Found
```

- Remedy: Restart BSS

```
ncn# kubectl -n services rollout restart deployment cray-bss
deployment.apps/cray-bss restarted
```

Is there a failure from BSS to provide iPXE boot script (mainly for NCNs)

- Wait for the roll out to complete

```
ncn# kubectl -n services rollout status deployment cray-bss
Waiting for deployment "cray-bss" rollout to finish: 1 out of 3 new
replicas have been updated...
Waiting for deployment "cray-bss" rollout to finish: 2 out of 3 new
replicas have been updated...
Waiting for deployment "cray-bss" rollout to finish: 1 old replicas are
pending termination...
Waiting for deployment "cray-bss" rollout to finish: 1 old replicas are
pending termination...
deployment "cray-bss" successfully rolled out
```

- Reboot the node that failed to get its boot script

Check the BSS contents

- Using the hostname

```
ncn# cray bss bootscript list --host HOST_NAME
```

- Using the MAC address

```
ncn# cray bss bootscript list --mac MAC_ADDRESS
```

- Using the node ID

```
ncn# cray bss bootscript list --nid NODE_ID
```

BSS Expected Contents

- Expected contents:

```
ncn# cray bss bootscript list --nid 1057 --format json
#!ipxe
kernel --name kernel http://rgw-vip.nmn/boot-images/c2ab083c-821a-49e1-a6d3-4ff008709d55/kernel?X-Amz-
Algorithm=AWS4-HMAC-SHA256&X-Amz-
Credential=UI262JX28D0QCXVQSL53%2F20240420%2Fdefault%2Fs3%2Faws4_request&X-Amz-Date=20240420T164130Z&X-Amz-
Expires=86400&X-Amz-SignedHeaders=host&X-Amz-
Signature=aa34a654079811b90a4194aa650165629f5d643e6abd27d4f6f4ef9618eee4e7 initrd=initrd
console=ttyS0,115200 bad_page=panic crashkernel=512M hugepagelist=2m-2g intel_pstate=disable
iommu.passthrough=on numa_balancing=disable numa_interleave_omit=headless oops=panic pageblock_order=14
pcie_ports=native rd.retry=10 rd.shell split_lock_detect=off systemd.unified_cgroup_hierarchy=1 ip=dhcp
quiet spire_join_token=d49d6bc3-17d8-421d-87f3-b03c872ac732 root=sbps-s3:s3://boot-images/c2ab083c-821a-
49e1-a6d3-4ff008709d55/rootfs:47748a2bf3e95b1e2fce322bbe914282-1977:sbps:v1:iqn.2023-06.csm.iscsi:_sbps-
hsn._tcp.<SYSTEM DOMAIN NAME>:30 nmd_data=url=s3://boot-images/c2ab083c-821a-49e1-a6d3-
4ff008709d55/rootfs,etag=47748a2bf3e95b1e2fce322bbe914282-1977 bos_update_frequency=4h xname=x1000c1s6b0n1
nid=1057 bss_referral_token=d739544a-0ee5-49ed-b499-0b65d0e61881 ds=nocloud-net;s=http://10.92.100.81:8888/
|| goto boot_retry

initrd --name initrd http://rgw-vip.nmn/boot-images/c2ab083c-821a-49e1-a6d3-4ff008709d55/initrd?X-Amz-
Algorithm=AWS4-HMAC-SHA256&X-Amz-
Credential=UI262JX28D0QCXVQSL53%2F20240420%2Fdefault%2Fs3%2Faws4_request&X-Amz-Date=20240420T164130Z&X-Amz-
Expires=86400&X-Amz-SignedHeaders=host&X-Amz-
Signature=d50d7fc51b17a3654fb0f92a64da913efb258675cadbaa932e6000c99cbb62df || goto boot_retry

boot || goto boot_retry
:boot_retry
sleep 30
chain https://api-gw-service-nmn.local/apis/bss/boot/v1/bootscript?mac=00:40:a6:82:f6:b0&retry=1
```

Troubleshooting failed boot BOS V1

- Tried to boot all compute nodes, but some failed to boot
- Where do you start looking?
  - BOA log shows widest view for BOSv1
    - Does the end of the BOA log have a traceback from some execution error?
    - Were there problems with BOA talking to BSS to assign the boot artifacts or to CFS to set desired configuration?
    - Were the power management calls to CAPMC smoothly done?
    - Were the node state status calls to HSM showing all nodes moving from OFF to ON to READY
      - With BOSv1, BOA has a 30-minute timeout to make these calls before giving up on the boot
    - Were the node configuration status calls to CFS showing all nodes (that had been in READY) moving to configured?
    - Did any of the calls from BOA to other services have 503 error messages?
      - This may mean that the are problems with the API gateway or that specific service might have an issue
        - Check the Kubernetes pod logs for the API gateway and the services which had 503 errors
        - Check whether any of the pods have an increasing restart count (or are in CrashLoopBackOff)
        - Check whether there have been OOMkill or CPUthrottling alerts during the boot
          - Some pods may need larger requests for memory or CPU resources if the system has recently been expanded with more nodes
    - Explore SMA-grafana dashboards for the time interval of the boot
    - Explore SMA-dashboards (OpenSearch for CSM-1.5) or SMA-kibana dashboards (ElasticSearch CSM-1.4) for the time interval of the boot
  - If BOA reports power management errors, there might be more detail in the CAPMC logs (CSM-1.3 and earlier) and PCS (CSM-1.4 and later)
    - If there has been an Emergency Power Off (EPO) event, special handling may be needed to recover from it before trying to boot

Troubleshooting failed boot BOS V2


```
{
  "actual_state": {
            SEE NEXT SLIDE
},
  "desired_state": {
             SEE NEXT SLIDE
},
  "enabled": false,
  "error": "",
  "event_stats": {
    "power_off_forceful_attempts": 0,
    "power_off_graceful_attempts": 0,
    "power_on_attempts": 0
  },
  "id": "x1000c5s7b1n1",
  "last_action": {
    "action": "powering_on",
    "failed": false,
    "last_updated": "2024-02-13T18:17:04"
  },
  "session": "3827307c-d67f-4d64-b33d-ddc8a7008afa",
  "staged_state": {
    "last_updated": "2024-01-08T23:25:58"
  },
  "status": {
    "phase": "",
    "status": "stable",
    "status_override": ""
  }
}
```


Troubleshooting failed boot BOS V2 (Continued)

```
ncn# cray bos v2 components describe x1000c1s1b0n2 --format json |
jq .status
{
  "phase": "",
  "status": "stable",
  "status_override": ""
}

ncn# cray bos v2 components describe x1000c1s1b0n3 --format json|
jq .status
{
  "phase": "powering_off",
  "status": "failed",
  "status_override": "failed"
}
```

Troubleshooting slow boot

- Booted, but some nodes appeared to be slow to boot
- Where do you start looking?
  - Console logs can help to identify which nodes are the slowest in any part of the booting process
  - BOSv1/BOA log shows widest view
    - BOA gets status from HSM and reports how many nodes are in OFF, ON, READY state and when they change states, but not to the node name granularity
      - Are nodes of the same type all changing state mostly together?
        - The slow nodes may be on the path to hardware failure
        - A system with a mixture of compute node types will often show variation in boot time by node type
    - BOS gets configuration status from CFS and reports how many nodes are not yet configured
      - Are nodes of the same type all moving into configured state together?
      - There may be tuning problems to address with Ansible plays run by CFS
      - There may be tuning need for the CFS infrastructure for the size of the system
  - BOSv2 shows boot statistics per session of how many nodes have booted (See next slide.)

BOS V2 Session status

```
ncn # cray bos v2 sessions status list <session ID> --format json
{
  "error_summary": {},
  "managed_components_count": 2,
  "percent_failed": 0.0,
  "percent_staged": 0.0,
  "percent_successful": 100.0,
  "phases": {
    "percent_complete": 100.0,
    "percent_configuring": 0,
    "percent_powering_off": 0,
    "percent_powering_on": 0
  },
  "status": "complete",
  "timing": {
    "duration": "0:29:22",
    "end_time": "2024-04-26T16:41:16",
    "start_time": "2024-04-26T16:11:54"
  }
}
```

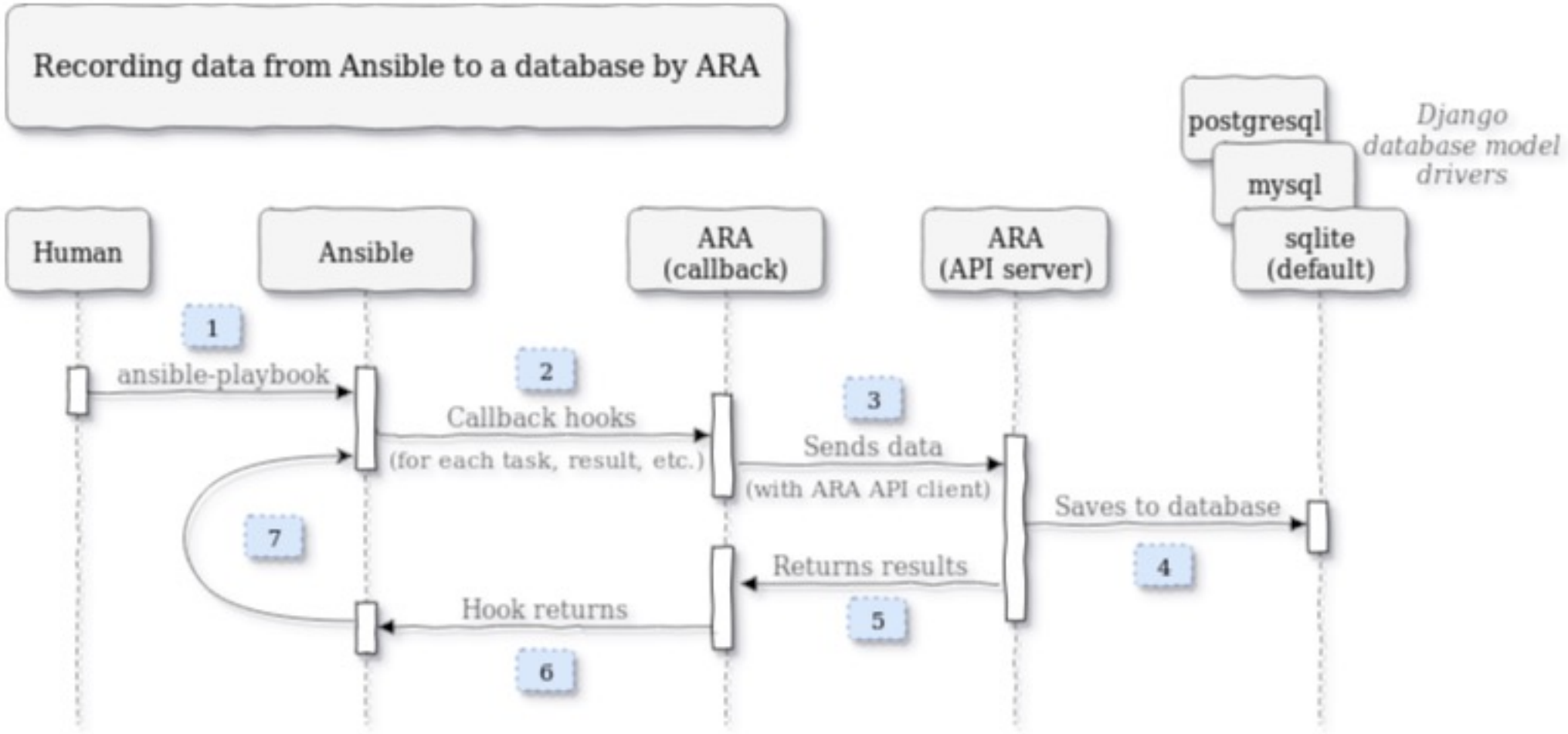
CFS troubleshooting

- CFS Debug Tools in CSM-1.4 (and later)
- ARA – ARA Records Ansible
- CFS Debugging tool
- CFS Improvements in CSM 1.5
- Debug_on_failure flag
- Debug playbooks

ARA Records Ansible (ARA)

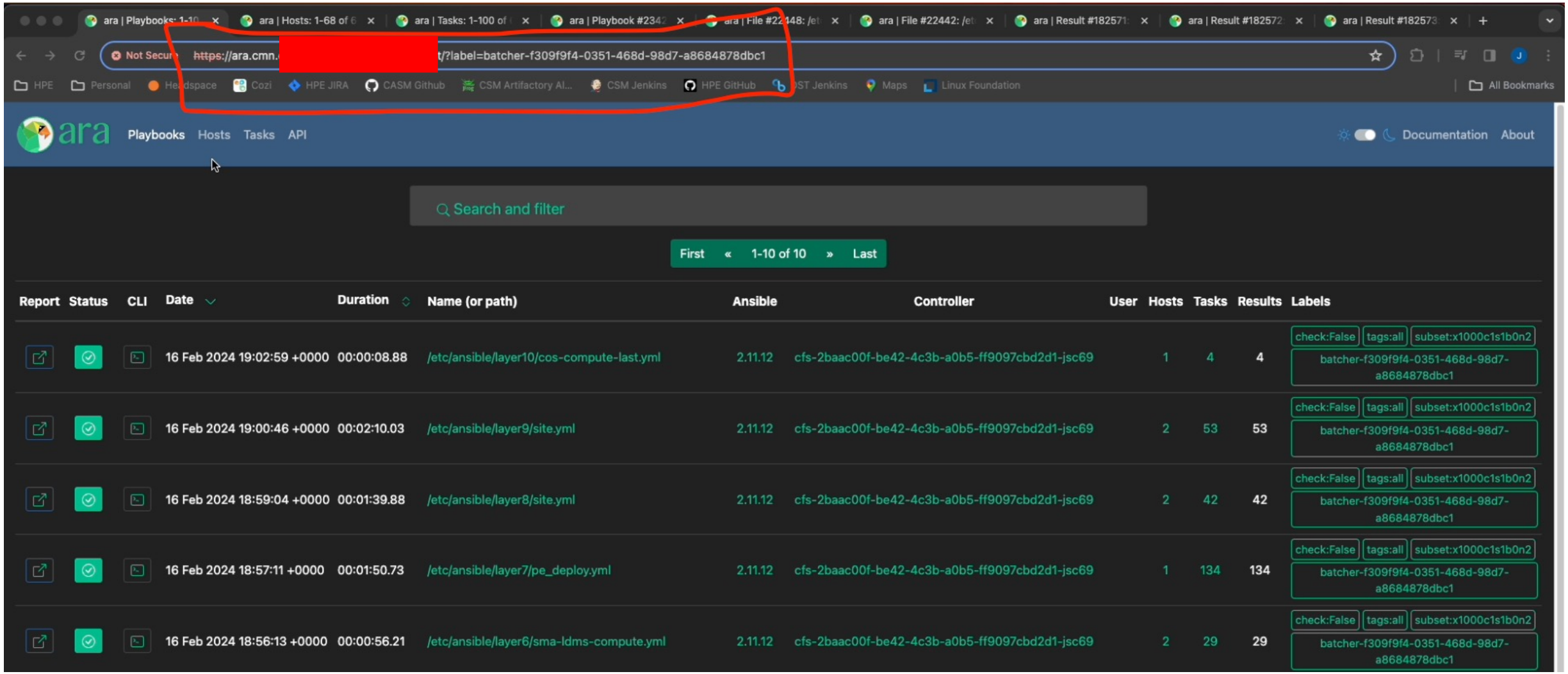
- Yes, it's another recursive acronym
- ARA is an open-source log collector, API, and UI, specifically for collecting and parsing Ansible logs
- ARA records the Ansible logs from all Configuration Framework Service (CFS) sessions
- ARA provides an Ansible friendly way to view them
- https://ara.recordsansible.org/
- ARA is installed with CSM-1.4
- ARA's GUI is accessible at: https://ara.cmn.<SYSTEM_DOMAIN_NAME>

ARA Recording Workflow



- Graphic from: https://ara.recordsansible.org/static/recording-workflow.png

ARA GUI URL

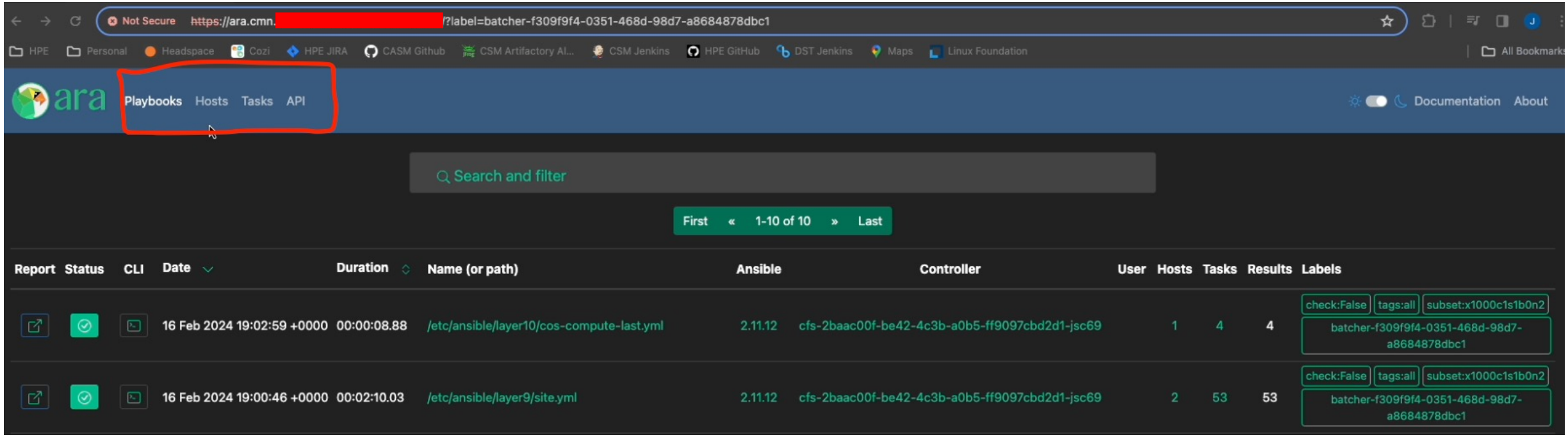


https://ara.cmn.SYSTEM_DOMAIN_NAME

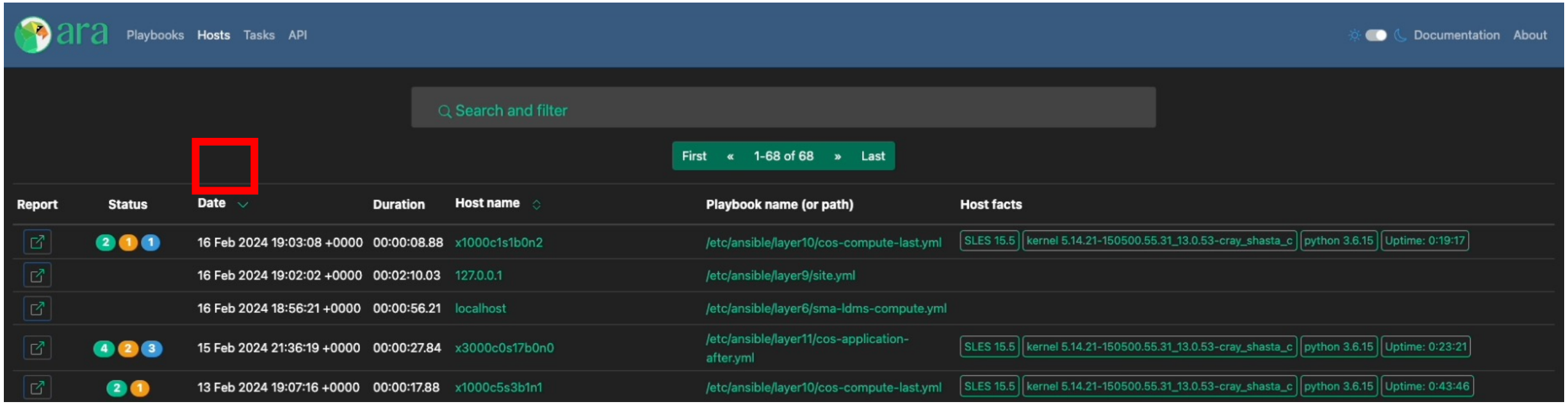
GUI Views

- Playbook
- Hosts
- Tasks
- API

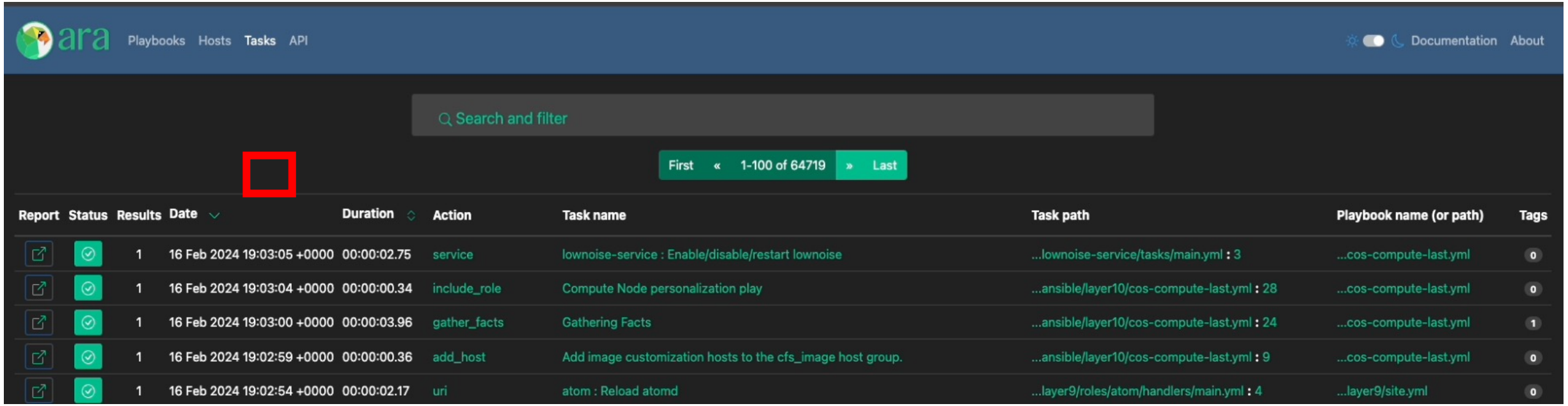
Playbooks


Not Secure https://ara.cmn. /?label=batcher-f309f9f4-0351-468d-98d7-a8684878dbc1
HPE
Personal
Headspace
Cozi
HPE JIRA
CASM Github
CSM Artifactory Al...
CSM Jenkins
HPE GitHub
DST Jenkins
Maps
Linux Foundation
All Bookmarks
ara
Playbooks
Hosts
Tasks
API
Documentation
About
Search and filter
First « 1-10 of 10 » Last
Report
Status
CLI
Date
Duration
Name (or path)
Ansible
Controller
User
Hosts
Tasks
Results
Labels
16 Feb 2024 19:02:59 +0000
00:00:08.88
/etc/ansible/layer10/cos-compute-last.yml
2.11.12
cfs-2baac00f-be42-4c3b-a0b5-ff9097cbd2d1-jsc69
1
4
4
check:False
tags:all
subset:x1000c1s1b0n2
batcher-f309f9f4-0351-468d-98d7-a8684878dbc1
16 Feb 2024 19:00:46 +0000
00:02:10.03
/etc/ansible/layer9/site.yml
2.11.12
cfs-2baac00f-be42-4c3b-a0b5-ff9097cbd2d1-jsc69
2
53
53
check:False
tags:all
subset:x1000c1s1b0n2
batcher-f309f9f4-0351-468d-98d7-a8684878dbc1

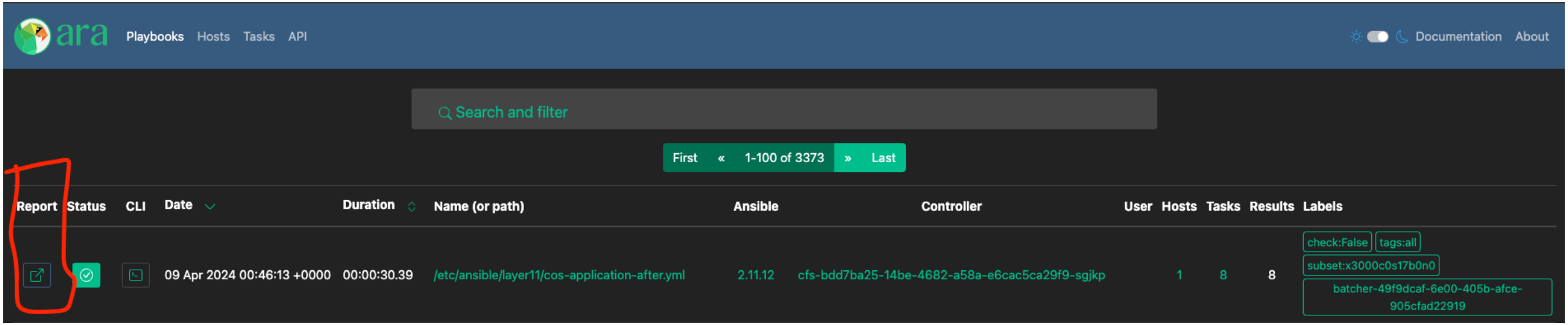
Hosts


ara
Playbooks
Hosts
Tasks
API
Documentation
About
Search and filter
First « 1-68 of 68 » Last
Report
Status
Date
Duration
Host name
Playbook name (or path)
Host facts
16 Feb 2024 19:03:08 +0000 00:00:08.88 x1000c1s1b0n2 /etc/ansible/layer10/cos-compute-last.yml SLES 15.5 kernel 5.14.21-150500.55.31_13.0.53-cray_shasta_c python 3.6.15 Uptime: 0:19:17
16 Feb 2024 19:02:02 +0000 00:02:10.03 127.0.0.1 /etc/ansible/layer9/site.yml
16 Feb 2024 18:56:21 +0000 00:00:56.21 localhost /etc/ansible/layer6/sma-ldms-compute.yml
15 Feb 2024 21:36:19 +0000 00:00:27.84 x3000c0s17b0n0 /etc/ansible/layer11/cos-application-after.yml SLES 15.5 kernel 5.14.21-150500.55.31_13.0.53-cray_shasta_c python 3.6.15 Uptime: 0:23:21
13 Feb 2024 19:07:16 +0000 00:00:17.88 x1000c5s3b1n1 /etc/ansible/layer10/cos-compute-last.yml SLES 15.5 kernel 5.14.21-150500.55.31_13.0.53-cray_shasta_c python 3.6.15 Uptime: 0:43:46

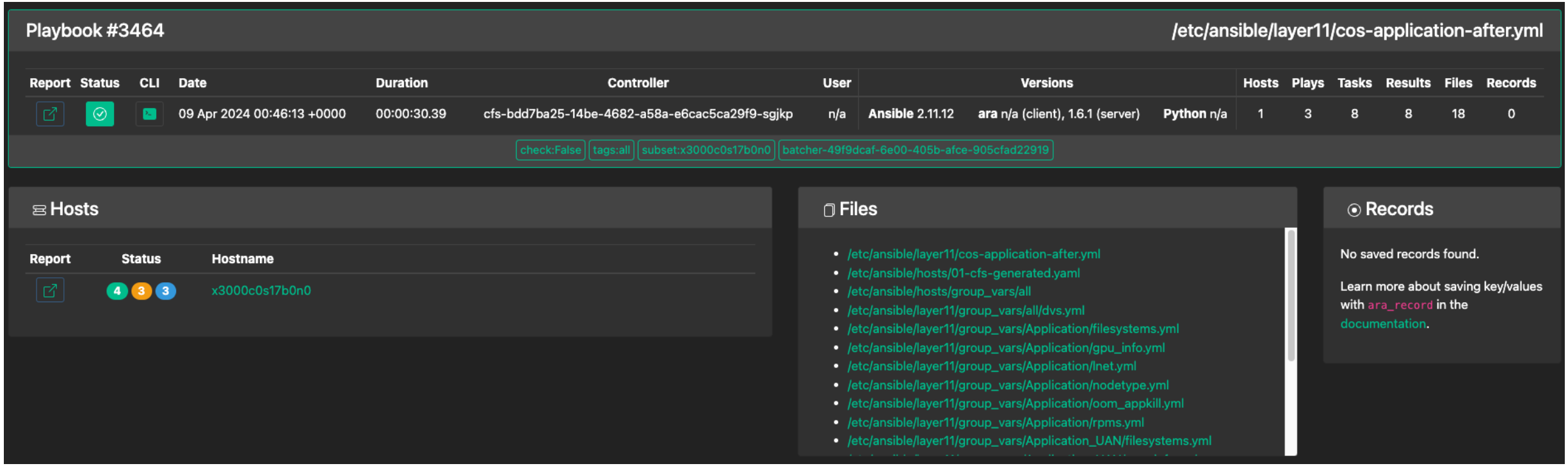
Tasks


ara
Playbooks
Hosts
Tasks
API
Documentation
About
Search and filter
First
«
1-100 of 64719
»
Last
Report
Status
Results
Date
Duration
Action
Task name
Task path
Playbook name (or path)
Tags
1 16 Feb 2024 19:03:05 +0000 00:00:02.75 service lownoise-service : Enable/disable/restart lownoise ...lownoise-service/tasks/main.yml : 3 ...cos-compute-last.yml 0
1 16 Feb 2024 19:03:04 +0000 00:00:00.34 include_role Compute Node personalization play ...ansible/layer10/cos-compute-last.yml : 28 ...cos-compute-last.yml 0
1 16 Feb 2024 19:03:00 +0000 00:00:03.96 gather_facts Gathering Facts ...ansible/layer10/cos-compute-last.yml : 24 ...cos-compute-last.yml 1
1 16 Feb 2024 19:02:59 +0000 00:00:00.36 add_host Add image customization hosts to the cfs_image host group. ...ansible/layer10/cos-compute-last.yml : 9 ...cos-compute-last.yml 0
1 16 Feb 2024 19:02:54 +0000 00:00:02.17 uri atom : Reload atomd ...layer9/roles/atom/handlers/main.yml : 4 ...layer9/site.yml 0

Playbook Reports (1)



Playbook Reports (2)



Playbook Reports (3)

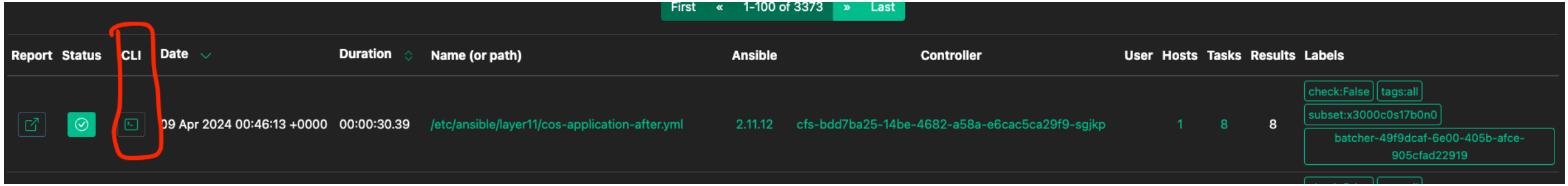
Task results

Search and filter

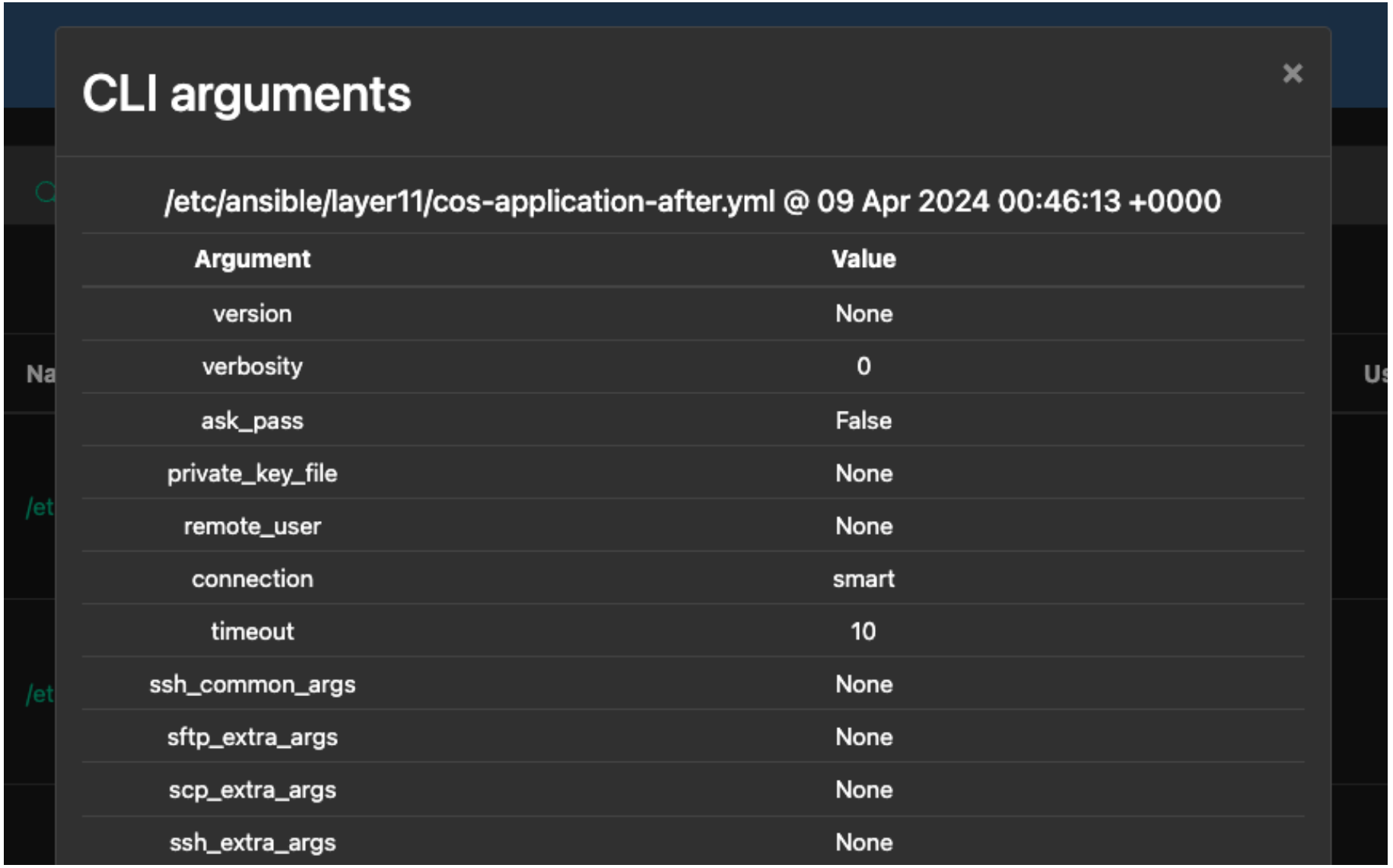
First « 1-8 of 8 » Last

Report	Status	Date	Duration	Host	Action	Task	Tags
	SKIPPED	09 Apr 2024 00:46:43 +0000	00:00:00.09	x3000c0s17b0n0	mount	configure_fs : Unmount	0
	CHANGED	09 Apr 2024 00:46:33 +0000	00:00:09.47	x3000c0s17b0n0	mount	configure_fs : Mount	0
	SKIPPED	09 Apr 2024 00:46:33 +0000	00:00:00.06	x3000c0s17b0n0	mount	configure_fs : Remove from /etc/fstab	0
	CHANGED	09 Apr 2024 00:46:28 +0000	00:00:04.15	x3000c0s17b0n0	mount	configure_fs : Add to /etc/fstab	0
	CHANGED	09 Apr 2024 00:46:24 +0000	00:00:04.04	x3000c0s17b0n0	command	lustre_config : Add route to Lustre FS	0
	OK	09 Apr 2024 00:46:23 +0000	00:00:00.02	x3000c0s17b0n0	include_role	Application node personalization play	0
	OK	09 Apr 2024 00:46:16 +0000	00:00:06.72	x3000c0s17b0n0	gather_facts	Gathering Facts	1
	SKIPPED	09 Apr 2024 00:46:14 +0000	00:00:00.01	x3000c0s17b0n0	add_host	Add image customization hosts to the cfs_image host group.	0

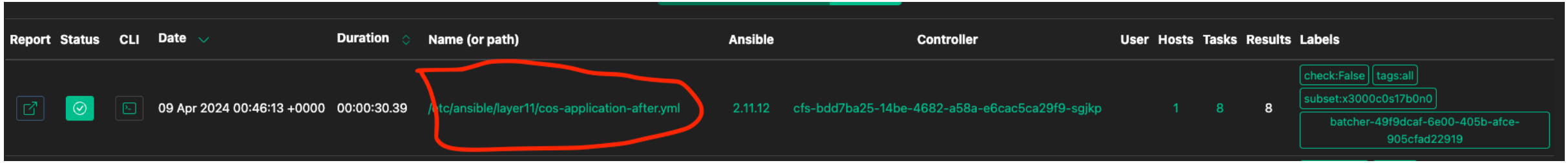
ARA Display CLI Arguments



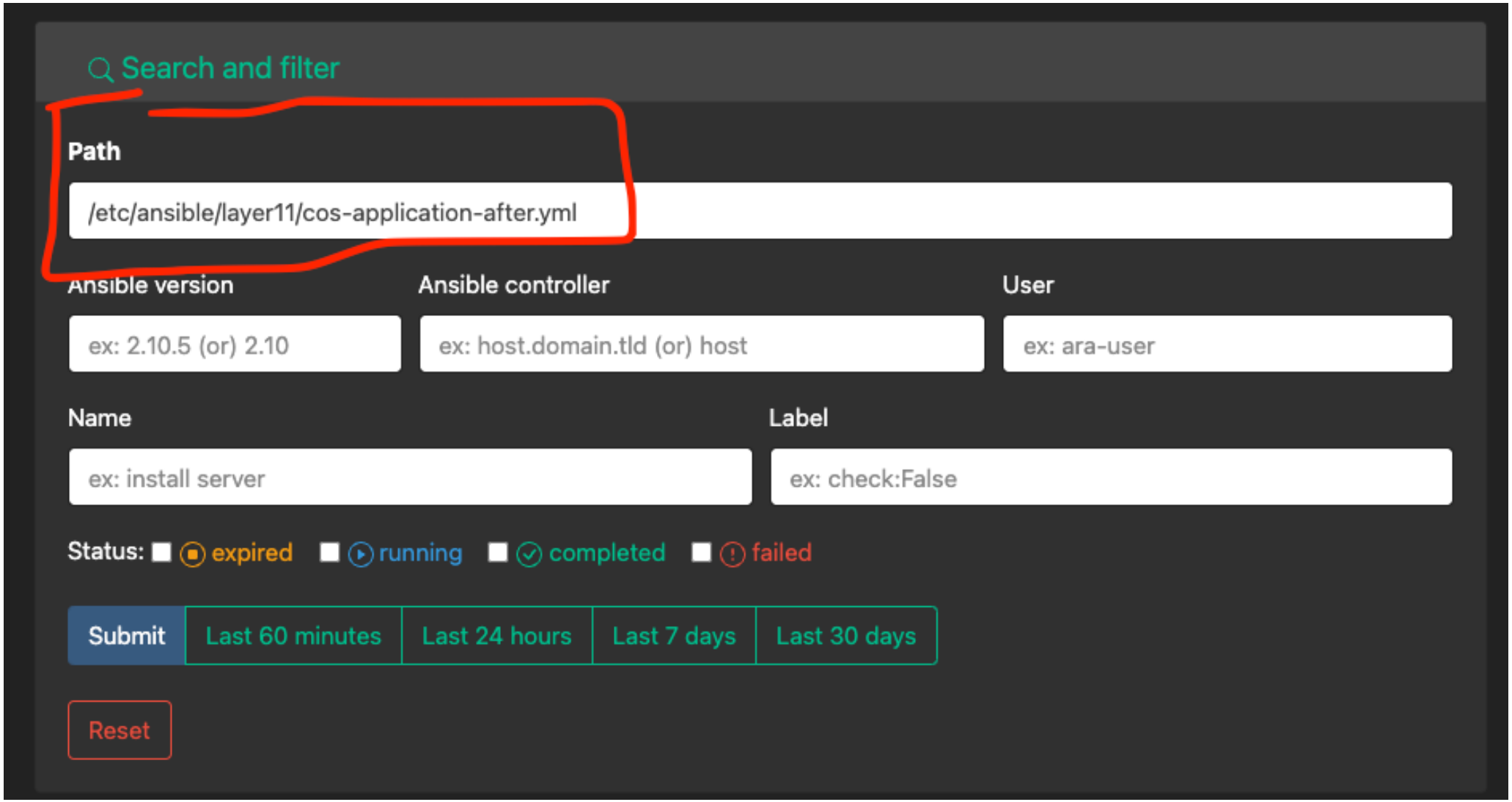
ARA CLI Pop-up



ARA Filtering



ARA Filter Input



There is also an API

- Not just a GUI

API Version

GET /api/

HTTP 200 OK
Allow: GET, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept


```
{
        "kind": "ara",
        "version": "1.6.1",
        "api": [
                "http://ara.cmn.<SYSTEM DOMAIN NAME>/api/v1/"
        ]
}
```


API Root

GET /api/v1

HTTP 200 OK
Allow: GET, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept


```
{
        "labels": "http://ara.cmn.<SYSTEM DOMAIN NAME>/api/v1/labels",
        "playbooks": "http://ara.cmn.<SYSTEM DOMAIN NAME>/api/v1/playbooks",
        "plays": "http://ara.cmn.<SYSTEM DOMAIN NAME>/api/v1/plays",
        "tasks": "http://ara.cmn.<SYSTEM DOMAIN NAME>/api/v1/tasks",
        "hosts": "http://ara.cmn.<SYSTEM DOMAIN NAME>/api/v1/hosts",
        "latesthosts": "http://ara.cmn.<SYSTEM DOMAIN NAME>/api/v1/latesthosts",
        "results": "http://ara.cmn.<SYSTEM DOMAIN NAME>/api/v1/results",
        "files": "http://ara.cmn.<SYSTEM DOMAIN NAME>/api/v1/files",
        "records": "http://ara.cmn.<SYSTEM DOMAIN NAME>/api/v1/records"
}
```


Playbook API

GET /api/v1/playbooks


```
{
   "count": 3952,
   "next": "http://ara.cmn.<SYSTEM DOMAIN NAME>/api/v1/playbooks?limit=100&offset=100",
   "previous": null,
   "results": [
                {
                "id": 4074,
                "items":
                {
                "plays": 3,
                "tasks": 4,
                "results": 4,
                "hosts": 1,
                "files": 9,
                "records": 0 },
                "arguments": {
                "version": null,
                "verbosity": 0
```


Latest Hosts API

GET /api/v1/latesthosts


```
{
"results": [
        {
        "name": "4fabcb0e-9d5d-4054-acb4-39d0087249b4",
        "host": {
                "id": 5772,
                "playbook": {
                        "id": 3523,
                        "items": {
                                "plays": 1,
                                "tasks": 2,
                                "results": 2,
                                "hosts": 1,
                                "files": 3,
                                "records": 0
                        },
```


Tasks list API

GET /api/v1/tasks


```
{
"count": 122913,
"next": "http://ara.cmn.<SYSTEM DOMAIN NAME>/api/v1/tasks?limit=100&offset=100",
"previous": null,
"results": [
{
        "id": 122974,
        "items": {
                "results": 1 },
        "path": "/etc/ansible/layer11/roles/lownoise-service/tasks/main.yml",
        "tags": [],
        "play": 9455,
        "duration": "00:00:03.085271",
        "name": "lownoise-service : Enable/disable/restart lownoise",
        "action": "service",
        "status": "completed",
        "file": 40748,
        "playbook": 4074
}
```


CFS Debugging Tool

- Tool to aid debugging failed CFS sessions and other CFS problems
- Invocation:
  - `ncn# cfs-debug`
- Example Display:
  - `Select debugger mode. Type help for details.`
    - `1) Auto-debug (default)`
    - `2) Directed-debug`
    - `3) Auto-debug report`
    - `4) Collect logs`
    - `5) Additional actions`
    - `0) Exit`

CFS-debug Mode: Auto-debug Output (part 1)

```
Running full check list...
[CFS health] The health of the CFS service
 - [ OK ] cray-cfs-api health:            healthy
 - [ OK ] cray-cfs-batcher health:        healthy
 - [ OK ] cray-cfs-operator health:       healthy
 - [ OK ] cfs-hwsync-agent health:        healthy
 - [WARN] cfs-trust health:               errors in the logs

cfs-trust health
================================================================================
cfs-trust ......................  errors in logs
Would you like to see details for this service (Y/n) # Y
 - Errors found in logs for pod: cfs-trust-9f97c6c56-jkt7g container: cfs-trust
--------------------------------------------------------------------------------
2024-02-20 19:16:12.906501+00:00        403 Client Error: Forbidden for url:
http://cray-vault.vault:8200/v1/transit/export/signing-key/cfstrust
--------------------------------------------------------------------------------
```

CFS-debug Mode: Auto-debug Output (part 2)

```
[CFS related health] The health of services related to CFS
 - [IMS health] The health of the IMS service
    - [ OK ] cray-ims health:                    healthy
 - [Kafka health] The health of the Kafka service
    - [ OK ] kafka health:                       healthy
    - [ OK ] zookeeper health:                   healthy
```

CFS-debug Mode: Auto-debug Output (part 3)

```
[CFS sessions health] The health of recent CFS sessions
 - [WARN] CFS recent session health:     Found 5 unhealthy recent sessions

CFS recent session health
================================================================================
batcher-3d48e43f-3a58-4326-8031-aff3f1ce1f68 ......  pending   ........ 3d
batcher-1c636c6c-1840-41d6-ab6d-6a563bd53ecd ......  pending   ........ 3d
batcher-bf807057-a394-4ea7-9fd7-a2c6007d8fb3 ......  pending   ........ 3d
batcher-2d9ae6c9-62f9-49e3-a713-57b6c31bd49f ......  pending   ........ 3d
batcher-5826010c-184b-4146-808b-8710a2a777b8 ......  pending   ........ 3d
--------------------------------------------------------------------------------

[CFS components health] The health of components from CFS' perspective
 - [WARN] CFS state-reporter health:     51/59 succeeded

CFS state-reporter health
================================================================================
8 components  could not be reached  :
x1000c3s2b0n3,x1000c3s2b0n2,x1000c1s1b0n3,x1000c1s2b1n0,...
--------------------------------------------------------------------------------
```

CFS Debug Mode: auto-debug failure diagnosis

```
CFS recent session health
================================================================================
sat-30d0e5f0-a61b-4c07-9b0c-ac60eb0bbccc ..........  failed   ......... 1h
Would you like to see details for this session (Y/n) # Y
 - Session sat-30d0e5f0-a61b-4c07-9b0c-ac60eb0bbccc failed on playbook sma-ldms-
application.yml (container ansible)
 - Ansible failure contained task output.
 - Parsing line starting with: "fatal: [13b55ce2-14f4-4494-ab04-8437c8f8919c]: FAILED!"
--------------------------------------------------------------------------------
warning: Found NDB Packages.db database while attempting bdb backend: using ndb
backend.
--------------------------------------------------------------------------------
<?xml version='1.0'?>
<stream>
<message type="error">No provider of &apos;+sma-system-test&apos; found.</message>
</stream>
--------------------------------------------------------------------------------
- This is not a recognized failure.  Please see the owner of this role.
```

CFS Debug Mode: Directed Debug

```
What sort of issue are you having?
1) Session(s) failed/stuck/unhealthy
2) Session(s) aren't starting
3) Ansible output isn't complete
4) Unknown/Other
```

CFS Debug Mode: Auto-debug report

- This automatically outputs all failures.
- In Auto-debug (Option 1), it prompted the user with a Y/n to display each failure.

CFS Debug Mode: Collect Logs

- It collects the logs. (Surprise!)

CFS Debug Mode: Additional Actions

```
Select action
1) Reset batcher throttling
2) Enable/Disable additional Ansible output
3) Enable/Disable Ansible profiling
4) Setup debug session
5) Stop all CFS configuration
6) Clone configuration
```

CFS-debug Workaround

```
ncn# python3 -m venv .venv
. .venv/bin/activate
(.venv) pip3 install fabric
(.venv) pip3 install gitpython
(.venv) MYDIR=$(find $(pwd) -name site-packages))
    /root/.venv/lib/python3.6/site-packages
(.venv) PYTHONPATH=/${MYDIR}/ cfs-debug
```

Debug_on_failure Flag

- CFS sessions can be created with the **debug_on_failure** flag
- If set to true, this will cause sessions that fail during Ansible execution to remain running so that users can exec into the pod

```
ncn# kubectl -n services exec -it <pod> -c ansible -- /bin/sh
```

- Once debugging is complete users should touch the **/tmp/complete** file to complete and cleanup the session
  - If this is not done, the session will remain up until the **debug_wait_time** expires
- NOTE: This is only available in the v3 CFS API which was released with CSM-1.5

Debug Playbooks

- CFS supports special debug playbooks, which are part of the Ansible Execution Environment (AEE) image and always available.
- These playbooks can be used without requiring a special configuration to be created.
- The following playbooks are available and can be specified as the configuration name for a session if no other configuration has already been created with these names.
  - **debug_fail** -
    - immediately fails
    - can be used with the **debug_on_failure** flag (previous slide) to quickl create an Ansible environment for debugging
  - **debug_facts** -
    - gathers and prints the facts for all available targets
  - **debug_noop** -
    - Way to test the CFS framework without running any Ansible tasks
    - Does not gather facts
    - Useful for skipping past the Ansible container for debugging
    - Easy way to test setting up the inventory and cloning content down

- NOTE: This is only available in the v3 CFS API which was released with CSM-1.5

Node hardware troubleshooting

CSM Diags

Hardware triage tool (CSM 1.5)

- Hardware triage tool identifies node failure scenarios and provides recommended actions
  - A node fails to power on
  - A node powers on but fails to boot
    - In this case, the node might be stuck at the UEFI shell
  - A node boots but fails the health check
  - A node reboots unexpectedly
  - A node powers down unexpectedly. This could be an emergency power down
  - A node becomes unresponsive during a job run but does not power down
- Supported platforms (Olympus cabinets)
  - EX235a, EX235n, EX254n, EX255a, EX425, EX4252

hwtriage

```
ncn# /opt/clmgr/hardware-triage-tool/hwtriage -u root -n x1000c4s4b0n0
Enter the password to access redfish calls :
Log collection completed

logging path :  /var/log/hardware-triage-tool/x1000c4s4b0n0_20240424_1224/x1000c4s4b0
ex235n Hardware is supported!
Triaging :x1000c4s4b0n0

Node is in On state
Node is not booted
Triaging :x1000c4s4b0n0    Analysis file : /var/log/hardware-triage-tool/x1000c4s4b0n0_20240424_1224/x1000c4s4b0/triage_output.json
Serial Numbers information :  /var/log/hardware-triage-tool/x1000c4s4b0n0_20240424_1224/x1000c4s4b0/serial_numbers.txt
Triaging :x1000c4s4b0n0    Node stuck in UEFI shell
NON_FATAL Errors occured, COUNT: 1
ERROR ENTRY - NON_FATAL
2024-04-20T12:40:56.01921 Type:   7
2024-04-20T12:40:56.01922 Flags:  0x01
2024-04-20T12:40:56.01923 Port80: 0xB000A691
2024-04-20T12:40:56.01924 ApicId: 0x00000000
2024-04-20T12:40:56.01926 CpuNum: 0
2024-04-20T12:40:56.01927 RTC:    04/20/2024 12:40:55-00:00
2024-04-20T12:40:56.01928 ErrorData: Degraded Device High-speed NIC1
2024-04-20T12:40:56.01929 DeviceName:  High-speed NIC1
2024-04-20T12:40:56.01935 Description: High-speed NIC1 VID:0x17DB DID:0x0501 B01:D00:F0 is degraded, Speed Exp: Gen4 Act: Gen4, Width Exp: x16 Act:x8.
2024-04-20T12:40:56.01936
ERROR ENTRY - END

Stage analysis : EFI_Shell Detected!
Triaging :x1000c4s4b0n0 Stage analysis : Unexpected_Booted Detected!
Triaging :x1000c4s4b0n0 Recommended  action :  None
Triaging :x1000c4s4b0n0 MCE Error occurred
Stage analysis : MachineCheckEventsNC Detected!
Recommended  action :  None
Triaging :x1000c4s4b0n0    Recommended  action :  None
Triage completed!
```

Slingshot troubleshooting

HPE Slingshot Troubleshooting Guide

- Fabric health
  - Fabric manager health engine
  - Fabric manager health monitor services
- Slingshot-Topology-Tool (STT)
  - River cable validator
- Troubleshooting HSN links that are down
  - General strategies for debugging HSN links
  - Using linkdbg to debug downed links
  - Show link flap events
  - HSN link failovers to ClusterStor server
- DNS troubleshooting
- Certificates and authentication
- Compute node troubleshooting guide
- Fabric manager
- Slingshot NIC
- Slingshot Switch
- Network diagnostics
- CXI Diagnostics and utilities
- CXI HealthCheck
- CXI Error Handling guidance and recommendations

fmn-show-status

- Check the current fabric status

```
ncn# kubectl exec -it -n services $(kubectl get pods -A |grep fabric |awk '{print $2}') -c slingshot-fabric-manager -- /bin/bash
slingshot-fabric-manager# fmn-show-status
Topology Status
Active: template-policy
Health
------
Runtime:HEALTHY
Configuration:WARNING
Traffic:HEALTHY
Security:HEALTHY
For more detailed Health - run 'fmctl get health-engines/template-policy'

Port Policies (online / total ports for each port-policy)
-----------------------------------------------
--------------
edge-policy: 0 / 0
fabric-policy: 10330 / 10340
cassini-policy: 4651 / 4664
qos-ll_be_bd_et-cassini-policy: 4651 / 4664
qos-hpc-cassini-policy: 0 / 0
qos-ll_be_bd_et-ethernet-policy: 0 / 0
qos-ll_be_bd_et-fabric-policy: 10330 / 10340
qos-hpc-fabric-policy: 0 / 0
lacp-policy: 0 / 0
offline-policy: 0 / 0

Edge: 4651 / 4664
Fabric: 10330 / 10340
Ports Reported: 15004 / 15004
Ports in Error State: 6 / 15004
Fully Synchronized Switches: 296 / 296
```

fmn-show-status details

- Check the current fabric status with details

```
ncn# kubectl exec -it -n services $(kubectl get pods -A |grep fabric |awk '{print $2}') -c slingshot-fabric-manager -- /bin/bash
slingshot-fabric-manager# fmn-show-status -d
Topology Status
… (Same as last slide for early part of output)

Edge: 4651 / 4664
Fabric: 10330 / 10340
Ports Reported: 15004 / 15004
Ports in Error State: 6 / 15004
Fully Synchronized Switches: 296 / 296

23 Downed links:
Fabric: x1000c5r5j5p0
Fabric: x1002c4r1j17p0
Fabric: x1003c4r1j17p0
Fabric: x1004c1r1j5p0
Fabric: x1006c0r7j13p1
Fabric: x1006c1r5j13p1
Fabric: x1006c6r1j13p1
Fabric: x1006c6r3j11p1
Fabric: x3000c0r33j3p0
Fabric: x3002c0r33j32p0
Edge: x1000c0r1j104p1
Edge: x1000c0r5j102p0
Edge: x1000c0r5j102p1
Edge: x1000c0r7j101p0
Edge: x1000c0r7j101p1
Edge: x1000c0r7j102p0
Edge: x1000c0r7j102p1
Edge: x1000c0r7j103p0
Edge: x1001c1r7j107p0
Edge: x1001c1r7j107p1
Edge: x1004c2r7j105p1
Edge: x1007c0r5j107p0
Edge: x1007c0r7j107p0

Port errors:
x1000c5r5j5p0 : Port capability degraded to prevent excessive flapping
x1000c0r7j101p0 : Port disabled due to excessive flapping
x3002c0r33j32p0 : Port disabled due to excessive flapping
x1003c4r1j17p0 : Port capability degraded to prevent excessive flapping
x3000c0r33j3p0 : Port disabled due to excessive flapping
x1002c4r1j17p0 : Port capability degraded to prevent excessive flapping
```

linkdbg fabric errors

- Check fabric link errors

```
slingshot-fabric-manager# linkdbg -L fabric
Querying downed links' link partners...

+--------+-----------------------------------------+-----------+---------+---------------
+--------------+------------+-----------------+------------+---------------+
|  type  | rosprt xname (pport) <-> rosprt xname (pport) | rosswinfo | sC firmW | sw_medtype-pw  |
lp rosswinfo | lp sC firmW | lp sw_medtype-pw | action_code | lp action_code |
+--------+-----------------------------------------+-----------+---------+---------------
+--------------+------------+-----------------+------------+---------------+
| Fabric | x1000c5r5j5p0   (25) <-> x1004c1r1j5p0   (25) | tpml d S L |  2.0.2   |   Optical-
07   |  tpml d s L  |    2.0.2    |    Optical-07    |  otherport  |      ros6      |
| Fabric | x1002c4r1j17p0  (58) <-> x1003c4r1j17p0  (58) | tpml d S L |  2.0.2   |   Optical-
06   |  tpml d S L  |    2.0.2    |    Optical-07    |  otherport  |      ros5      |
| Fabric | x1006c0r7j13p1  (62) <-> x1006c6r1j13p1  (62) | tpml d s L |  2.0.2   | Electrical-N/A
|  tpml d S L  |    2.0.2    |  Electrical-N/A  |     ros6    |      ros5      |
| Fabric | x1006c1r5j13p1  (62) <-> x1006c6r3j11p1  (14) | tpml d s L |  2.0.2   | Electrical-N/A
|  tpml d S L  |    2.0.2    |  Electrical-N/A  |     ros6    |      ros5      |
| Fabric | x3000c0r33j3p0  (18) <-> x3002c0r33j32p0 (33) | tpml D S L |  2.0.2   |   Optical-
06   |  tpml D S L  |    2.0.2    |    Optical-06    |     ros4    |      ros4      |
+--------+-----------------------------------------+-----------+---------+---------------
+--------------+------------+-----------------+------------+---------------+
```

linkdbg edge errors

- Check fabric link errors

```
slingshot-fabric-manager# linkdbg -L edge
Querying downed links' link partners...
```

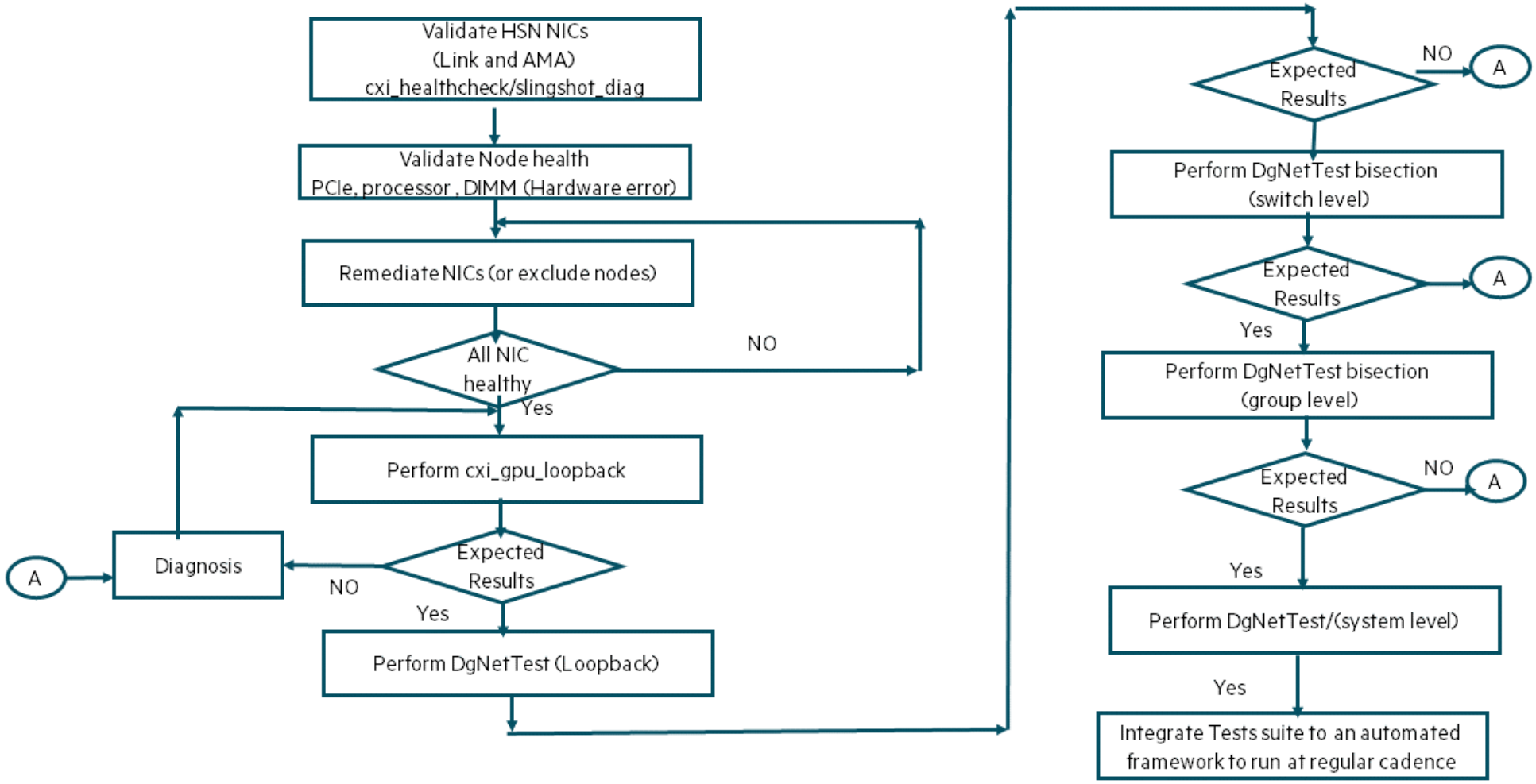
type	rosprt xname (pport) <-> link_partner	rosswinfo	sC firmW	sw_medtype-pw	action_code
Edge	x1000c0r1j104p1 (16) <-> x1000c0s4b0n1h1	tpml d S L	2.0.2	Electrical-N/A	unkn_port
Edge	x1000c0r5j102p0 (49) <-> x1000c0s2b1n1h1	tpml d S L	2.0.2	Electrical-N/A	unkn_port
Edge	x1000c0r5j102p1 (48) <-> x1000c0s2b1n0h1	tpml d S L	2.0.2	Electrical-N/A	unkn_port
Edge	x1000c0r7j101p0 (34) <-> x1000c0s1b1n0h0	tpml D S L	2.0.2	Electrical-N/A	ros4
Edge	x1000c0r7j101p1 (35) <-> x1000c0s1b1n1h0	tpml d S L	2.0.2	Electrical-N/A	unkn_port
Edge	x1000c0r7j102p0 (49) <-> x1000c0s2b1n1h0	tpml d S L	2.0.2	Electrical-N/A	unkn_port
Edge	x1000c0r7j102p1 (48) <-> x1000c0s2b1n0h0	tpml d S L	2.0.2	Electrical-N/A	unkn_port
Edge	x1000c0r7j103p0 (33) <-> x1000c0s3b1n1h0	tpml d S L	2.0.2	Electrical-N/A	unkn_port
Edge	x1001c1r7j107p0 (2) <-> x1001c1s7b1n1h0	tpml d S L	2.0.2	Electrical-N/A	unkn_port
Edge	x1001c1r7j107p1 (3) <-> x1001c1s7b1n0h0	tpml d S L	2.0.2	Electrical-N/A	unkn_port
Edge	x1004c2r7j105p1 (0) <-> x1004c2s5b1n1h0	tpml d S L	2.0.2	Electrical-N/A	unkn_port
Edge	x1007c0r5j107p0 (2) <-> x1007c0s7b1n1h1	tpml d S L	2.0.2	Electrical-N/A	unkn_port
Edge	x1007c0r7j107p0 (2) <-> x1007c0s7b1n1h0	tpml d S L	2.0.2	Electrical-N/A	unkn_port

linkdbg action codes

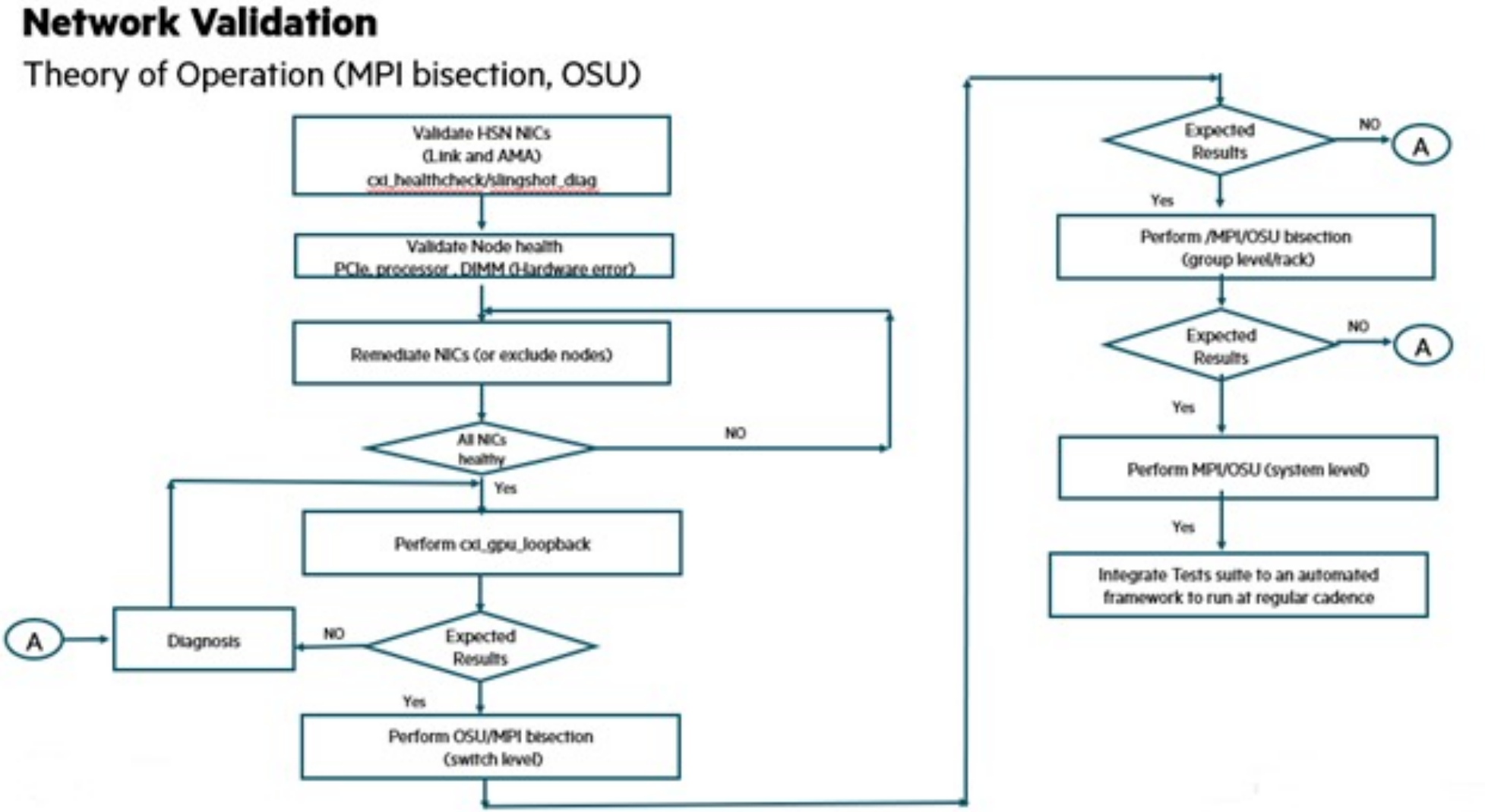
- Get explanation of action codes

```
slingshot-fabric-manager# linkdbg -a ros4
PROBLEM SYNOPSIS: The link has been directed down.

WHY YOU GOT THIS ACTION CODE:
    The link monitor (lmon) state letter in the rosswinfo column, "D/d",
is
    capitalized.

HOW TO DIAGNOSE:
    Validate each issue, in order.

POSSIBLE ISSUES:
    1) The link direction could be in a transient state.  Rerun
    "linkdbg -t <portxname>" twice over a 10 second interval.
    2) Link auto retry has been exhausted.  This will be reported in fmn_status on
    the FMN.  The link was flapping too much, and needs hardware attention:

        (After each step, re-run linkdbg to see if the action resolved the issue.)

        Steps to debug Mountain Cabinet downed links between NIC and switch:
            1) Validate both switch and NIC are properly configured and attempting
            to bring up the HSN link.
            2) Perform group hug:
                Apply pressure to both the compute and switch blades simultaneously
                to seat the ExaMax connectors more firmly.
            3) Reseat the switch blade (this will act as an asic reset and reboot
as well).
            4) Reseat the compute blade.
            5) Swap NIC mezzanine cards between nodes. # look to see if follows NIC
or stays with cable.
                If failure follows NIC, replace the NIC.
                If failure stays with L0 cable, replace the L0 cable.
            6) Replace node card with known node card that has good link on the
            reporting errored HSN link.
            7) Replace switch.

        Steps to debug River Cabinet downed links between NIC and switch:
            1) Validate both switch and NIC are properly configured and attempting
            to bring up the HSN link.
            2) Reseat cable between NIC card and switch.
            3) Reboot compute node.
            4) Reset asic and reboot switch.
            5) Reseat cable between NIC card and switch a second time.
            6) Replace cable between NIC card and switch.
            7) Replace NIC card.
            8) Replace switch.

        Steps to debug downed links between switches:
            1) Validate both switches are properly configured and attempting to
            bring up the HSN link.
            2) Reseat cable between switches.
            3) Reset ASIC
            4) Slot power cycle
            5) Reseat cable between switches a second time.
            6) Replace cable between switches.
            7) Replace local switch.
            8) Replace remote switch.

    3) The link hasn't been commanded up.  Run the following from the switch, where
    "portnumber" is the number reported by linkdbg in parenthesis next to its xname:
     swtest -c "link $((1<<portnumber)) up"
```

Flowchart for Slingshot fabric validation with dgnettest



Flowchart for Slingshot fabric validation with OSU benchmark



Triage data collection for HPE Service

Triage data collection for HPE Service

- Simple command output to attach to case/ticket
  - What is the system serial number and what software is installed?

    ```
    ncn-m# sat showrev > sat.showrev.txt
    ```
  - A problem is suspected with a specific $PODNAME in a Kubernetes $NAMESPACE

    ```
    ncn# kubectl describe –$NAMESPACE pod $PODNAME > kubectl.describe.$NAMESPACE.$PODNAME.txt
    ncn# kubectl logs -n $NAMESPACE --timestamps pod $PODNAME >\
    kubectl.logs.$NAMESPACE.PODNAME.txt
    ```
  - A node has a problem

    ```
    ncn# ssh $NODE dmesg –T > $NODE.dmesg.t.txt
    ```

    Also collect console log for node from cray-console-node
- Node Memory Dump (NMD)
  - Trigger or collect a kernel crash dump
- Slingshot (hsn_triage_capture)
  - Collect Slingshot logs from fabric-manager, switches, switch consoles
- System Diagnostic Utility (SDU)
  - Comprehensive collection of triage data using multiple plugins

Node Memory Dump (NMD)

- Standard Linux kdump mechanism
  - Uses `kexec` for booting into the dump-capture kernel (`kdump boot`) immediately after kernel crash
  - Standard `kdump` not scalable to large systems
    - Standard, each node decides on its own to produce a node memory dump
  - Needs a service to initiate dumps of selected nodes
- NMD controls the `kdump` process of the panicked node
  - Provides concurrent dump capability
  - Controls automated kdump so that the dump is generated only for the requested nodes
  - Provides a configurable makedumpfile dump level option for the selected node at dump time
    - Can specify dumplevel argument of the makedumpfile command
      - 31 by default
      - 16 if it is required to retrieve user process core dump (user data pages) or non-private cache pages

```
ncn# cray nmd dumps --help
Usage: cray nmd dumps [OPTIONS] COMMAND [ARGS]...
Options:
  --help  Show this message and exit.
Commands:
  create
  delete
  describe
  list
```

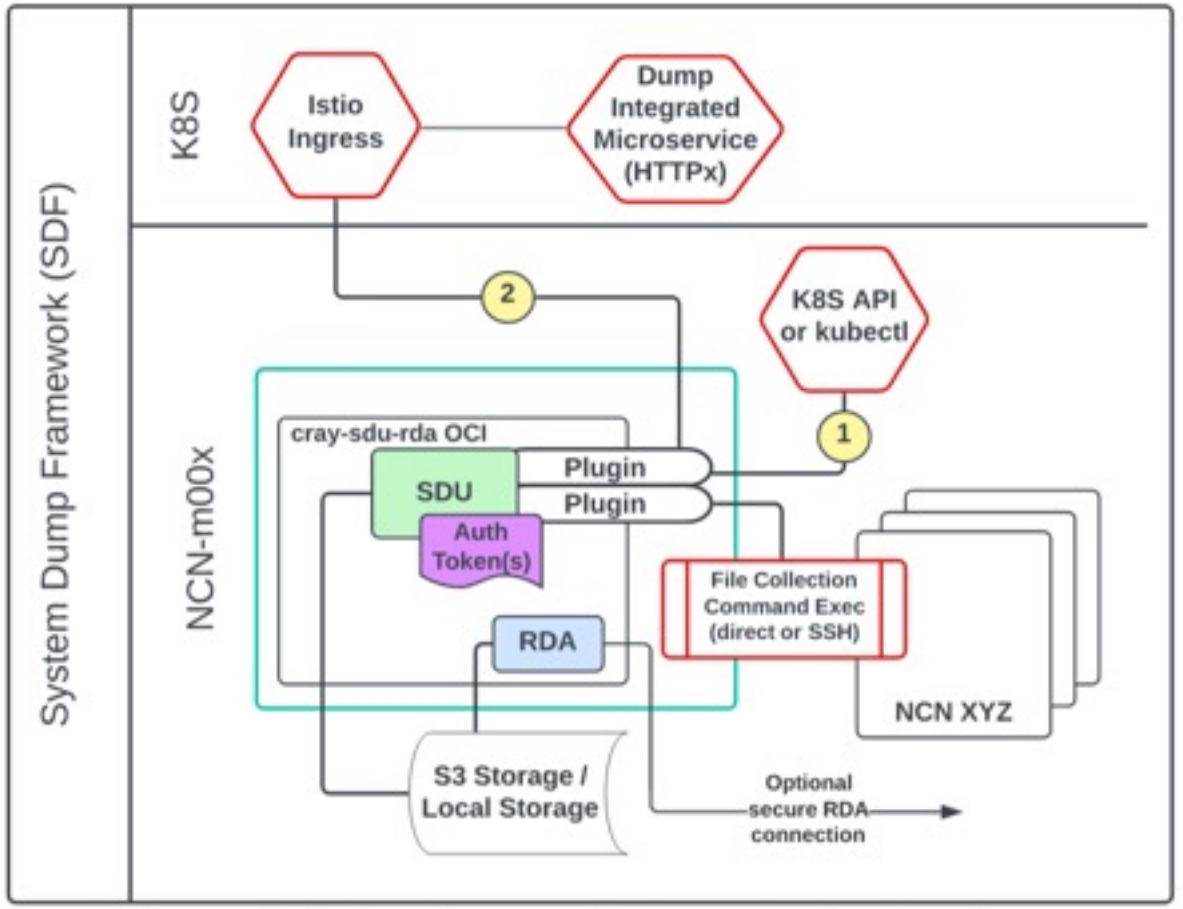
Slingshot hsn_triage_capture

- If a problem occurs while following the HSN Debug Procedure, capture system log files by running `hsn_triage_capture` to capture state and logs from the fabric manager, switches and CMMs
  - When it completes, it prints the path to the tarball of captured data

```
ncn# kubectl exec -it -n services $(kubectl get pods -A |grep fabric |awk '{print $2}') -c slingshot-fabric-manager -- /bin/bash
slingshot-fabric-manager# hsn_triage_capture -h
usage: ./hsn_triage_capture [-h] [-a] [-c] [-f] [-s] [-t TARGET_XNAME[,TARGET_XNAME]]
Collect HSN debug information from the FMN, CMMs and switches,
then aggregate into a single tarball.
Note: it is assumed that passwordless ssh is configured to all devices.
optional arguments:
-h Show usage and exit
-a Collect FMN, console, and switch data (default)
-c Collect switch console logs
-f Collect FMN data
-s Collect switch data
-t TARGET_XNAME[,TARGET_XNAME] Specific targets only
```

System Dump Framework (SDF)

- Provides a standard system dump feature
  - Onsite triage
  - Onsite to HPE support
  - Provides a structured data format
- Resiliency model
  - ncn-m001 and ncn-m002
    - but SDU can be started on ANY master node, only 1 at a time
  - Each eligible master node should have a unique RDA configuration



System Diagnostic Utility (SDU)

- Pluggable architecture to collect logs, core files, register dumps, and more
  - Can package the output to `tar` to share any useful system triage information
  - Collects data from distributed parts of the system
- Remote Device Access (RDA) is capable of securely transporting this data to HPE
  - RDA Documentation: https://midway.ext.hpe.com/home
  - Security white paper: https://support.hpe.com/hpesc/public/docDisplay?docId=a00006791en_us
  - AFT (Asynchronous File Transport) is used to securely transport SDU data to HPE
  - IDA (Interactive Device Access) is used to tunnel TCP sessions with HPE
  - AFT and IDA are independent from one another and both are opt-in features
- SDU runs in a podman container controlled as a service via systemd on master node

```
ncn-m# systemctl start cray-sdu-rda.service
```

- `sdu` commands can be run from the master node or within the container

```
ncn-m001# sdu bash
ncn-m001-sdu#   <--- prompt indicates you are inside the SDU/RDA container
```

- If multiple local copies of a triage collection are desired, then increase from default retention from 2

```
ncn-m001# vi /etc/opt/cray/sdu/sdu.conf
collect_retain_per_channel: 4
```

SDU scenarios

- Health
  - Performs a system health collection to gather health information from the system
  - Useful times to run
    - After CSM install.sh completes
    - Before and after NCN reboots
    - After the system is brought back up
    - Any time there is unexpected behavior observed
    - In order to provide relevant information to create support tickets
- Inventory
  - Performs an inventory collection to gather version information for software, firmware, and hardware
  - Useful to run after system upgrades
    - The information collected is used by the HPE Cray Service and R & D organizations to improve customer support
- Triage
  - Performs a triage collection which will gather diagnostic information and logs necessary for HPE Cray Service and R & D to perform problem determination and isolation
  - You are encouraged to provide the --ref 'sfdc:<case number>' command line option to ensure that the snapshot is associated with your SFDC service case

SDU triage scenario

```
ncn-m001# sdu --scenario triage --start_time '-2 days' --reason "Problem with system"
2024-04-18T18:39:59Z RUNLOG START  channel: triage, scenario: (csm.triage, {})
2024-04-18T18:39:59Z Exclusive run: Lock file created @ /var/opt/cray/sdu/lock/sdu.lock_channel-
triage_system-Creek
2024-04-18T18:39:59Z Discovering collection output.
2024-04-18T18:40:00Z
--- Collection Directory State ---
- execution context: Job Construction
- path: /var/opt/cray/sdu/collection-mount
- provisioned_bytes: 21474836480
- allocated_bytes: 93118649 (0.43%)
- free_bytes: 21381717831 (99.57%)
…
2024-04-18T19:14:28Z Plugin 'cray.sls.dump' stopped, return 0, time 3.45
2024-04-18T19:14:32Z Creating posix view
2024-04-18T19:15:00Z Summary report available at: /var/opt/cray/sdu/collection-
mount/triage/view/2024-04-18T18-39-59_UTC-
bfe307f8a6c382db9bd206b927d190a5/localhost/files/report/summary_report.txt
2024-04-18T19:15:00Z SDU session stop successfully
2024-04-18T19:15:00Z run took 2101.39 seconds
ncn-m001# cd /var/opt/cray/sdu/collection-mount/triage/view/2024-04-18T18-39-59_UTC-
bfe307f8a6c382db9bd206b927d190a5
```

All data collected from plugins will be in the view directory

Explore SDU view

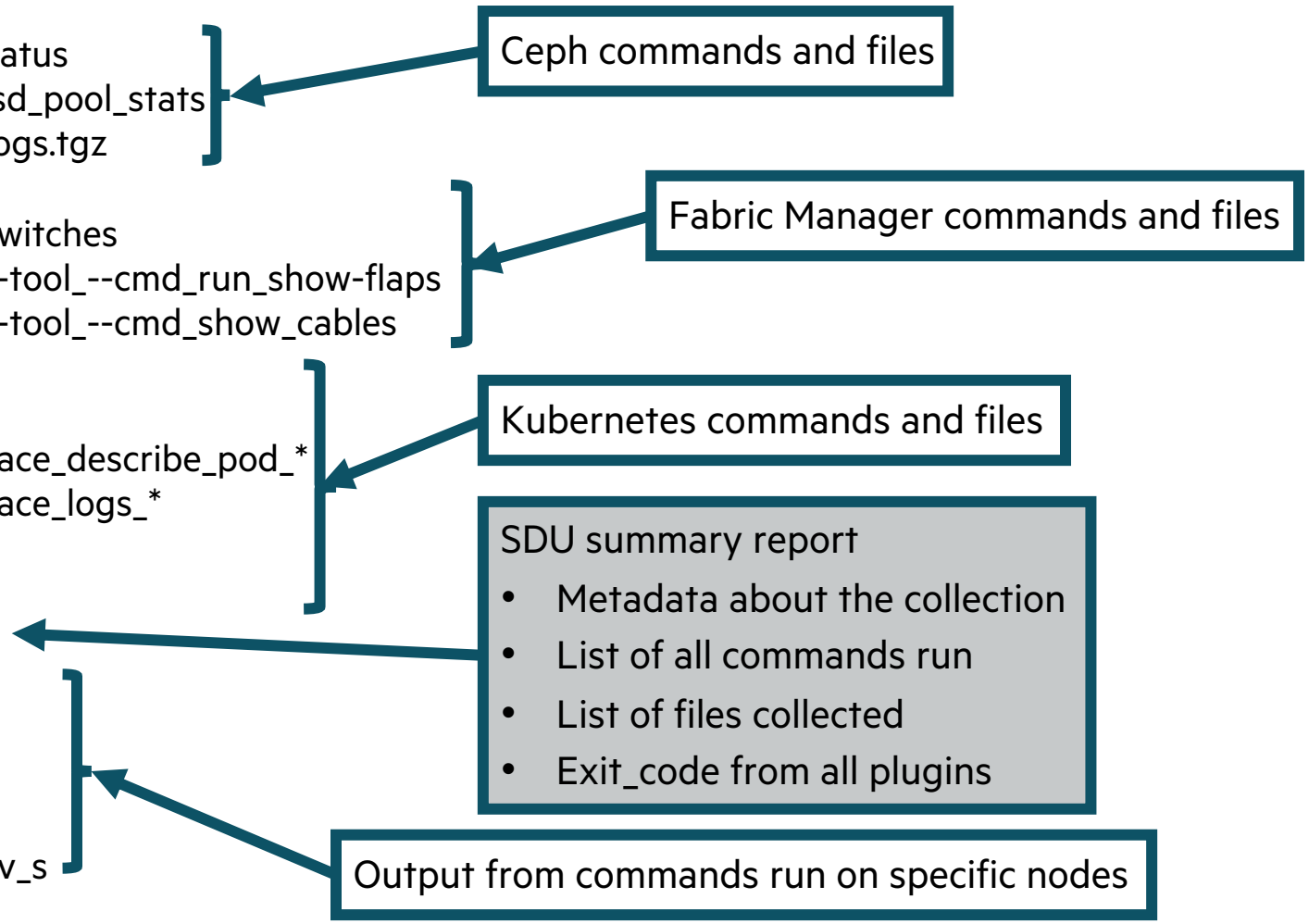
- Dump contents are organized first by host or system management component, and then by content type (files and cmds)
  - The following is an example of the directory path:

```
ncn-m001# ls -l
total 8
drwxrwx--- 1 2370 2370   0 Apr 18 19:14 ceph
drwxrwx--- 1 2370 2370   0 Apr 18 19:14 fmn
drwxrwx--- 1 2370 2370   0 Apr 18 19:14 k8s.services
drwxrwx--- 1 2370 2370   0 Apr 18 19:15 localhost
drwxrwx--- 1 2370 2370   0 Apr 18 19:14 ncn-m001
drwxrwx--- 1 2370 2370   0 Apr 18 19:14 ncn-m002
drwxrwx--- 1 2370 2370   0 Apr 18 19:14 ncn-m002-sdu
drwxrwx--- 1 2370 2370   0 Apr 18 19:14 ncn-m003
drwxrwx--- 1 2370 2370   0 Apr 18 19:14 ncn-s001
drwxrwx--- 1 2370 2370   0 Apr 18 19:14 ncn-s002
drwxrwx--- 1 2370 2370   0 Apr 18 19:14 ncn-s003
drwxrwx--- 1 2370 2370   0 Apr 18 19:14 ncn-w001
drwxrwx--- 1 2370 2370   0 Apr 18 19:14 ncn-w002
drwxrwx--- 1 2370 2370   0 Apr 18 19:14 ncn-w003
lrwxrwx--- 1 2370 2370 108 Apr 18 19:14 session-1713465599-bfe307f8a6c382db9bd206b927d190a5.json
-> /var/opt/cray/sdu/collection-mount/triage/manifests/session-1713465599-
bfe307f8a6c382db9bd206b927d190a5.json
drwxrwx--- 1 2370 2370   0 Apr 18 19:14 sma
```

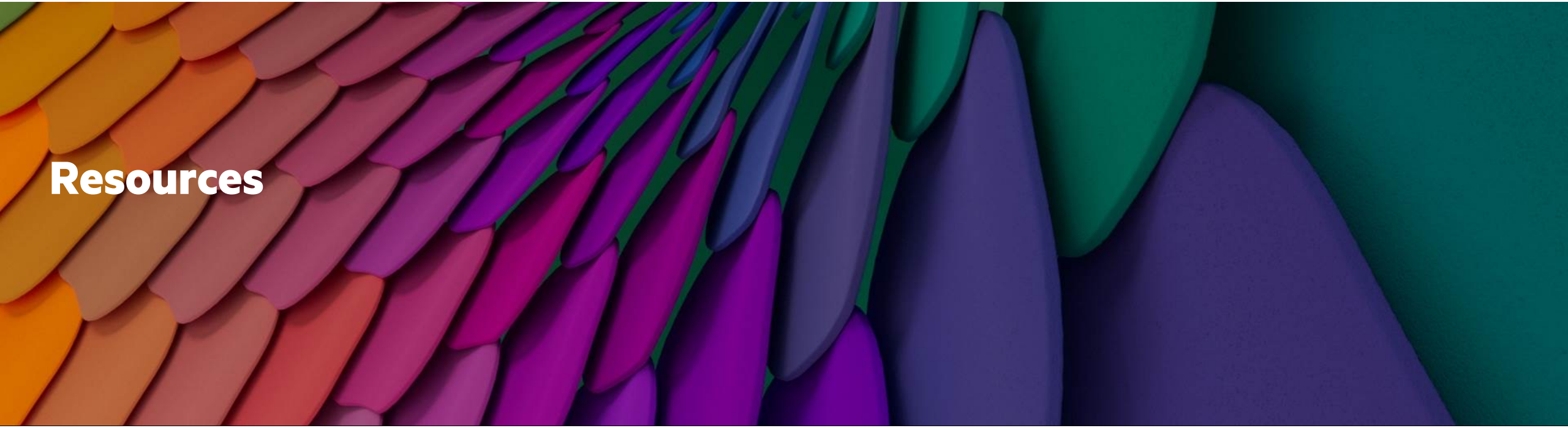
Additional subdirectories exist that contain the logs, core files, register dumps, and more

Explore SDU data

- Sample files in subdirectories
  - ceph/cmds/ncn-s001_usr_bin_ceph_status
  - ceph/cmds/ncn-s001_usr_bin_ceph_osd_pool_stats
  - ceph/files/ncn-s001/ncn-s001-ceph-logs.tgz
  - fmn/cmds/usr_bin_fmn_status
  - fmn/cmds/usr_bin_fmctl__get_fabric_switches
  - fmn/cmds/usr_bin_slingshot-topology-tool_--cmd_run_show-flaps
  - fmn/cmds/usr_bin_slingshot-topology-tool_--cmd_show_cables
  - k8s/cmds/usr_bin_kubectl_describe_*
  - k8s/cmds/usr_bin_kubectl_get_*
  - k8s/cmds/usr_bin_kubectl_-n_namespace_describe_pod_*
  - k8s/cmds/usr_bin_kubectl_-n_namespace_logs_*
  - k8s/cmds/usr_bin_kubectl_top_nodes
  - k8s/cmds/usr_bin_kubectl_top_pods
  - localhost/files/report/summary_report
  - ncn-s001/ncn-s001-ceph-logs.tgz
  - ncn-w003/cmds/usr_bin_dmesg
  - ncn-w003/cmds/sbin_lsmod
  - ncn-w003/cmds/sbin_sysctl_-a
  - ncn-w003/cmds/usr_sbin_smartctl_dev_s



Resources



Resources

- Documentation
- Open-Source Software
- Training
- Related Presentations

Documentation - Installation

- HPE Cray EX System Software Getting Started Guide S-8000
- HPE Cray System Management (CSM) Markdown
  - https://github.com/Cray-HPE/docs-csm/tree/release/1.5
- HPE Cray System Management (CSM) HTML
  - https://cray-hpe.github.io/docs-csm/en-15/
- HPE Cray EX System Admin Toolkit HTML
  - https://cray-hpe.github.io/docs-sat/en-26/
- HPE Cray EX System CSM Diagnostics Installation and Configuration Guide
- HPE Cray EX System Diagnostic Utility (SDU) Installation Guide
- HPE Cray EX System HPC Firmware Pack Installation Guide S-8037
- HPE Cray EX System Monitoring Application Installation Guide S-8030
- HPE Cray Programming Environment Installation Guide: CSM on HPE Cray EX S-8003
- HPE Cray Supercomputing Operating System Administration Guide CSM on HPE Cray EX Systems
- HPE Cray Supercomputing User Services Software Administration Guide: CSM on HPE Cray EX Systems (S-8063)
- HPE Cray User Access Node Software Installation Guide S-8032
- HPE Slingshot Release Notes
- HPE Slingshot Operations Guide
- HPE SUSE Linux Enterprise Operating System Installation Guide S-8028

Documentation - Administration

- HPE Cray System Management (CSM) Markdown
  - https://github.com/Cray-HPE/docs-csm/tree/release/1.5
  - https://github.com/Cray-HPE/docs-csm/blob/release/1.5/operations/kubernetes/Kubernetes.md
  - https://github.com/Cray-HPE/docs-csm/blob/release/1.5/glossary.md
- HPE Cray System Management (CSM) HTML
  - https://cray-hpe.github.io/docs-csm/en-15/
- HPE Cray EX System Admin Toolkit Guide S-8031
- HPE Cray EX System CSM Diagnostics Administration Guide
- HPE Cray EX System Diagnostic Utility (SDU) Administration Guide
- HPE Cray EX System Monitoring Application Administration Guide S-8029
- HPE Cray Programming Environment User Guide: CSM on HPE Cray EX S-8005
- HPE Cray Supercomputing Operating System Administration Guide CSM on HPE Cray EX Systems
- HPE Cray Supercomputing User Services Software Administration Guide: CSM on HPE Cray EX Systems (S-8063)
- HPE Cray User Access Node Software Administration Guide S-8033
- HPE Slingshot Operations Guide
- HPE Slingshot Troubleshooting
- HPE Slingshot Hardware Guide

Documentation – open-source tools

- CSM
  - MIT License
  - Github Hosted
    - https://github.com/Cray-HPE
  - Community Governance
    - https://github.com/Cray-HPE/community
- SAT
  - MIT License
  - Github Hosted
    - Primary repository for the sat CLI written in Python: https://github.com/Cray-HPE/sat
    - Podman wrapper script written in Bash: https://github.com/Cray-HPE/sat-podman
    - An important library used by sat CLI: https://github.com/Cray-HPE/python-csm-api-client
  - Documentation starting point:
    - https://github.com/Cray-HPE/sat/blob/integration/CONTRIBUTING.md
    - https://github.com/Cray-HPE/sat/blob/integration/docs/developer/README.md

- 3rd party open-source
  - https://kubernetes.io/docs/home/
  - https://kubernetes.io/docs/reference/kubectl/cheatsheet/
  - https://lmgtfy.com/?q=kubernetes+troubleshooting
  - https://www.elastic.co/guide/en/kibana/current/index.html
  - https://grafana.com/docs/
  - https://github.com/aelsabbahy/goss
  - http://docs.ansible.com/
  - https://kubernetes.io/docs/reference/kubectl/jsonpath/
  - https://stedolan.github.io/jq/manual/
  - http://www.compciv.org/recipes/cli/jq-for-parsing-json/
  - https://osinside.github.io/kiwi/
  - https://ara.recordsansible.org/

SUPERCOMPUTING: HPE CRAY EX Training

Where to start?

From HPE Edu
http://www.hpe.com/ww/training

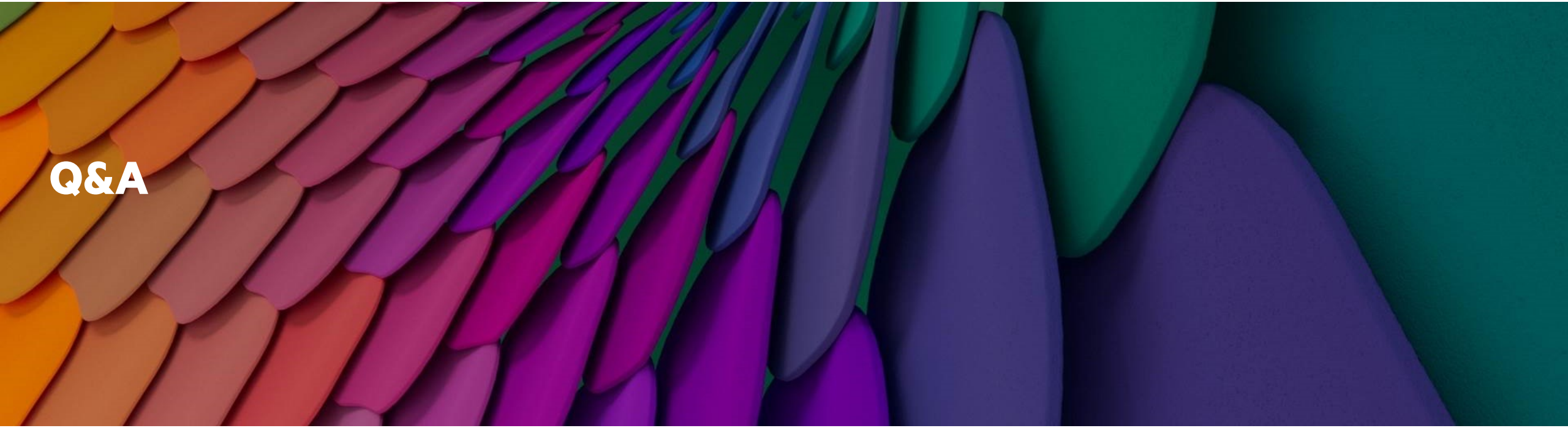
- Select HPE Cray EX Series and ClusterStor Storage

https://education.hpe.com/ww/en/training/portfolio/servers.html#ServersLearningPathsIntro

Course ID	Course Title	Duration	View Schedule
HQ7G6S	HPE Cray EX Series Prerequisite Training Bundle	15 hours	Register →
HQ7D6S	HPE Cray EX System Administration with CSM	5 days	Register →
H9TT2S	HPE Cray EX System Administration with HPE PCM	5 days	Register →
H8PG3S	HPE Cray EX Programming and Optimization	4 days	Register →
HQ6X8AAE	HPE Cray EX Series Overview, Rev. 20.31	8 hours	Register →
HQ6X5AAE	HPE Cray Supercomputer Rack System Hardware Overview, Rev. 20.31	2 hours	Register →
HQ6X6AAE	HPE Cray EX Supercomputer Hardware Overview, Rev. 20.31	3 hours	Register →
HQ6X7AAE	HPE Cray EX Series Test and Development Hardware Overview, Rev. 20.31	2 hours	Register →
HQ7D8S	Cray ClusterStor L300 System Administration	2 days	Register →
HQ7G5S	Cray ClusterStor E1000 Prerequisite Training Bundle	6 hours	Register →
H8PG4S	Cray ClusterStor E1000 System Administration	3 days	Register →
HQ7L0AAE	Cray ClusterStor E1000 System Architecture, Rev. 20.31	2 hours	Register →
HQ7K8AAE	Cray ClusterStor E1000 Overview, Rev. 20.31	2 hours	Register →
HQ7K9AAE	ClusterStor E1000 Install, Rev. 20.31	2 hours	Register →
HQ6Y6AAE	Cray ClusterStor L300 Overview, Rev. 20.31	1 hour	Register →

Related presentations and papers

- CUG 2024
  - From Frontier to Framework: Enhancing Hardware Triage for Exascale Machines
  - Reducing Mean Time to Resolution (MTTR) for complex HPC-based systems with next generation automated service tools
  - AIOPS Empowered: Failure Prediction in System Management Software Tools
  - CSM-based Software Stack Overview 2024
  - HPE Cray EX Power Monitoring Counters
  - Unification of Alerting Engines for Monitoring in System Management
- CUG 2023
  - Advanced topics for Cray System Management for HPE Cray EX Systems
  - Cray Systems Management (CSM) Security Policy Engine
  - New User Experiences with K3s and MetalLB on Managed Nodes
  - Software-defined Multi-tenancy on HPE Cray EX Supercomputers
- CUG 2022
  - HPE Cray EX Shasta 22.03 Cray System Management Overview
  - Cray System Management for HPE Cray EX Systems
  - Dealing with Metrics Data – Where is it, How to get it, What to do with it?
  - UAIs Come of Age: Hosting Multiple Custom Interactive Login Experiences Without Dedicated Hardware
- CUG 2021
  - User and Administrative Access Options for CSM-Based Shasta Systems
- CUG 2020
  - Advanced Topics in Configuration Management
- CUG 2019
  - Shasta Software Technical Workshop
  - Shasta System Management Overview
  - Reimagining Image Management in the New Shasta Environment
  - Hardware Discovery and Maintenance Workflows in Shasta Systems


Q&A

Thank you

Harold Longley, harold.longley@hpe.com
Jason Sollom, jason.sollom@hpe.com