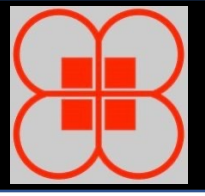


Redfish-Nagios: A Scalable Out-of-Band Data Center Monitoring Framework Based on Redfish Telemetry Model

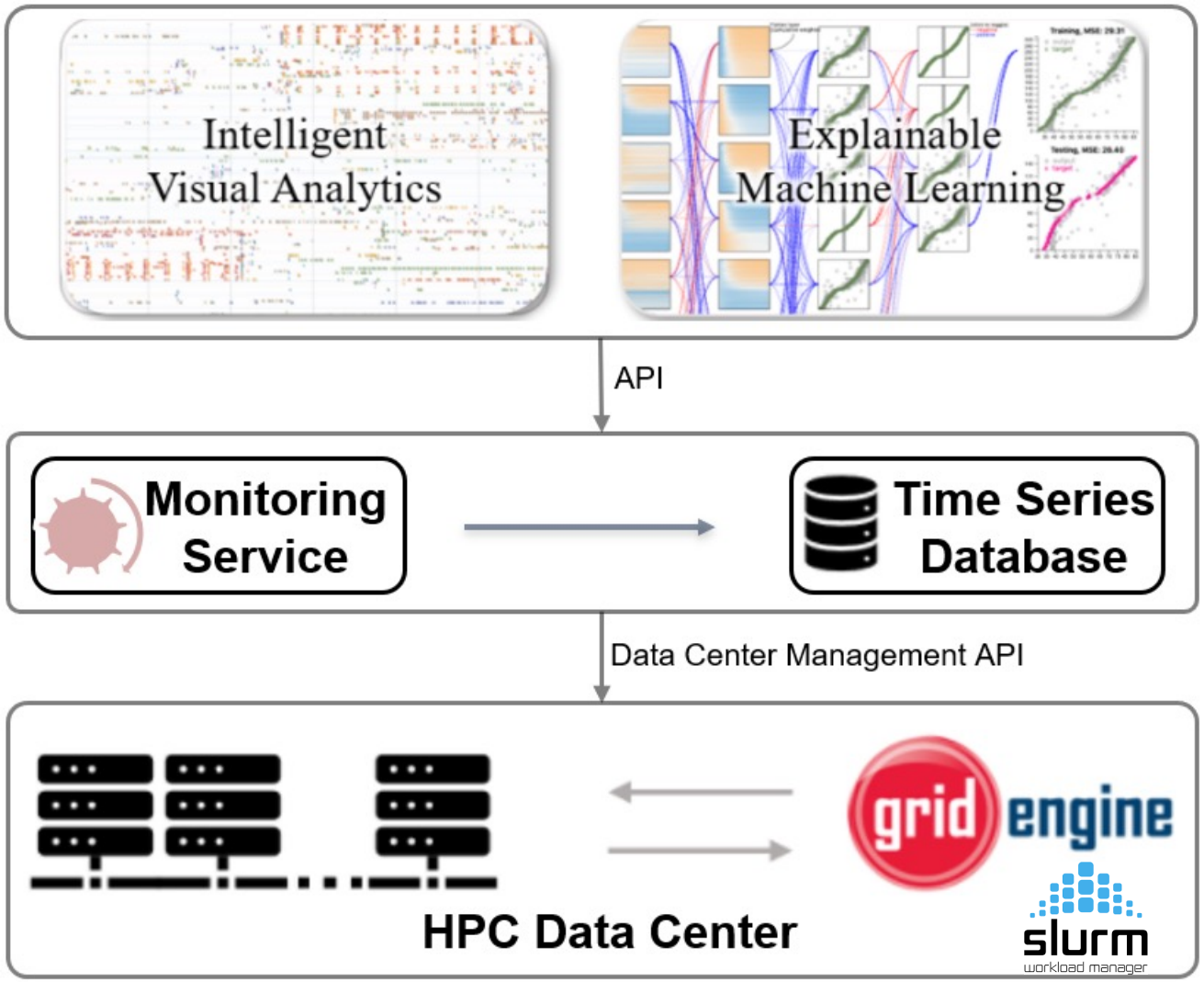
Authors: Ghazanfar Ali✧, Jon Hass†, Alan Sill✧, Elham Hojati✧, Tommy Dang✧, and Yong Chen✧

✧ Texas Tech University, Lubbock, TX, USA

† Dell Technologies, TX, USA

Typical Data Center Monitoring Framework

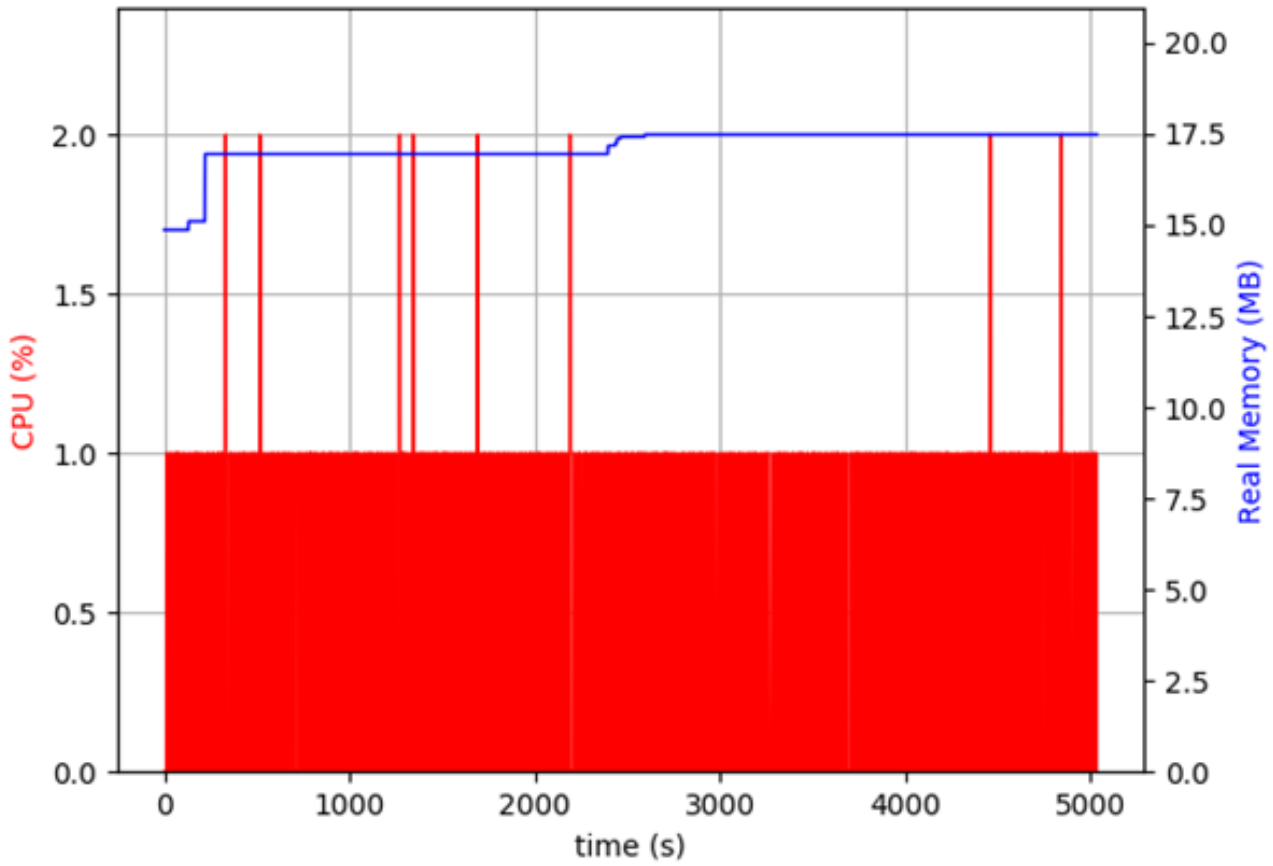
- Data center infrastructure consists of hardware and software resources
- Monitoring service acquires metrics related to data center infrastructure, often via in-band, and store them in a database (typically time series DB)
- Analytic services analyze the metrics and provide various useful insights about the applications and infrastructure resources



Overheads of In-band Monitoring

In-band monitoring requires operating system to access the target service and perform monitoring functions:

- Causes the consumption of precious compute resources
- Uses ~1-2% CPU & ~17 MB memory in monitoring power metric (1-sec interval)
- Risks malfunctioning of the HW and SW components



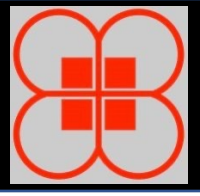
TEXAS TECH UNIVERSITY

DISCL

NSF CAC Cloud and Autonomic Computing Center

DELL Technologies

Problem Statement/Motivation



- Nagios is one of the widely used tools for data center monitoring [1]
- Nagios have limitations due to its in-band-monitoring nature:
  - Nagios monitoring requires monitoring-specific agents on each monitored node
  - Nagios Remote Plugin Executor and the Nagios Service Check Acceptor are required on the Nagios server and each monitored node
  - Manual effort is needed for the configuration of monitored nodes in the Nagios server

- These shortcomings are inherently due to Nagios' in-band implementation
- To overcome these limitations, we introduce out-of-band and scalable Redfish-Nagios monitoring solution

[1] Nagios. 2022. *Nagios-The Industry Standard In IT Infrastructure Monitoring.* Retrieved May, 2022 from https://www.nagios.org

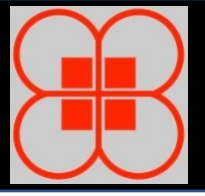
TEXAS TECH UNIVERSITY

DISCL

NSF CAC Cloud and Autonomic Computing Center

DELL Technologies

Objectives



- To replace Nagios in-band protocols
  - Redfish-Nagios eliminates in-band protocols (i.e., NRPE, NSCA) by providing the monitoring functions through the baseboard management controller (BMC) via out-of-band (OOB) protocol
- To enable agent-less monitoring
  - Offloads monitoring processing from the on-node agent to the BMC
  - Simplifies monitoring (no requirement for development, installation, and maintenance of an agent on remotely monitored nodes)
- To automate configurations
  - Nagios requires manual effort to configure the monitored infrastructure
  - Automating the configuration process is an important capability for the monitoring of large-scale modern data centers

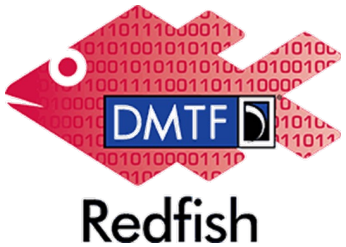
Out-of-band Monitoring Protocols

- Intelligent Platform Management Interface (IPMI):
  - Prominent initial OOB interfaces, widely adopted in HPC/cloud systems
  - Broadly used to perform remote control & acquisition of telemetry data

IPMI

BMC

- DMTF Redfish [2]:
  - Due to notable disadvantages (i.e., security, scalability, bit-wise nature) of IPMI, Redfish standard was developed to address these concerns
  - Redfish is designed to manage and monitor data center in a secure and scalable manner
  - Redfish leverages common Internet and web service standards to expose monitoring information

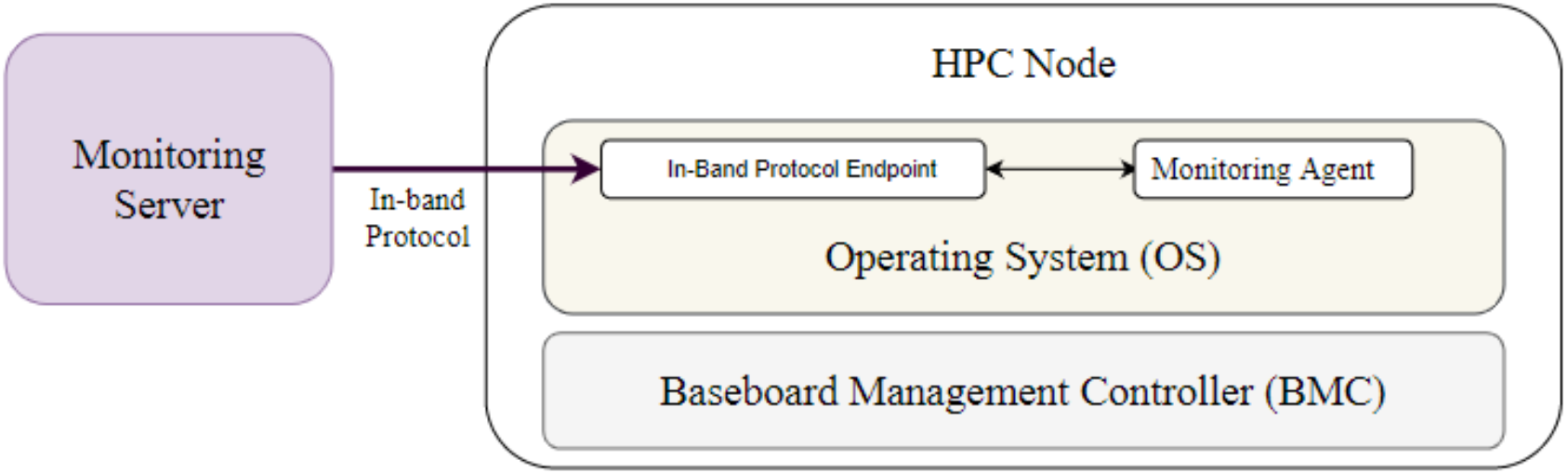


BMC

[2] Jeff Hilland. *Redfish Overview*. In Companion Proceedings of the 10th International Conference on Utility and Cloud Computing (UCC). 2017.

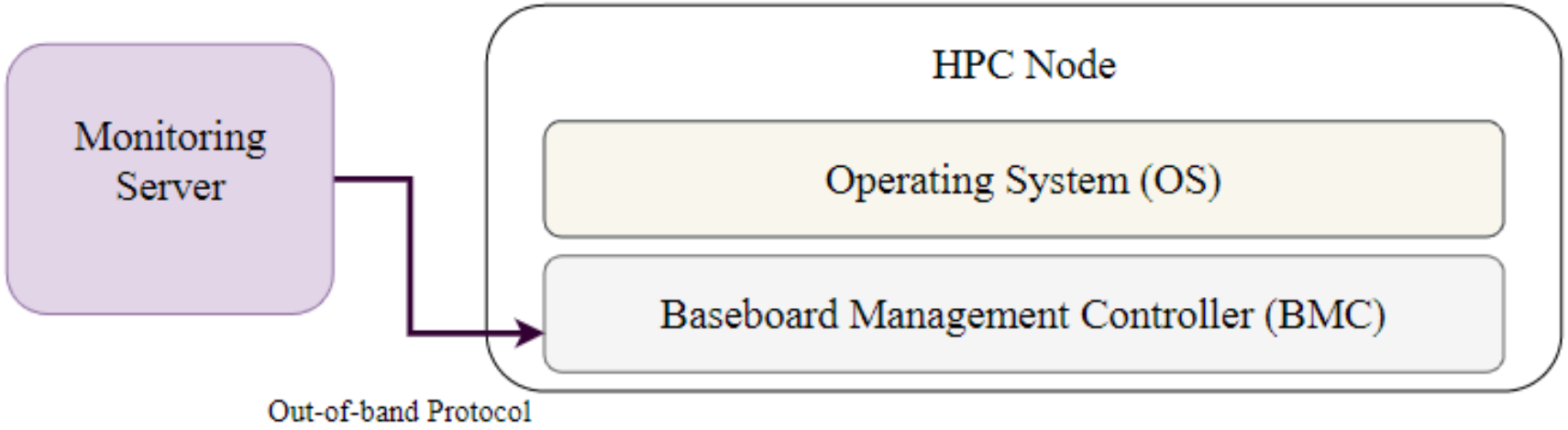
Method Comparison: In-band Monitoring

- Monitoring Server initiates the acquisition of monitoring data from the monitored node.
- The in-band endpoint is responsible for implementing a particular monitoring function (in Nagios, NRPE acts as in-band endpoint and collects metrics)



Method Comparison: Out-of-band (OOB) Monitoring

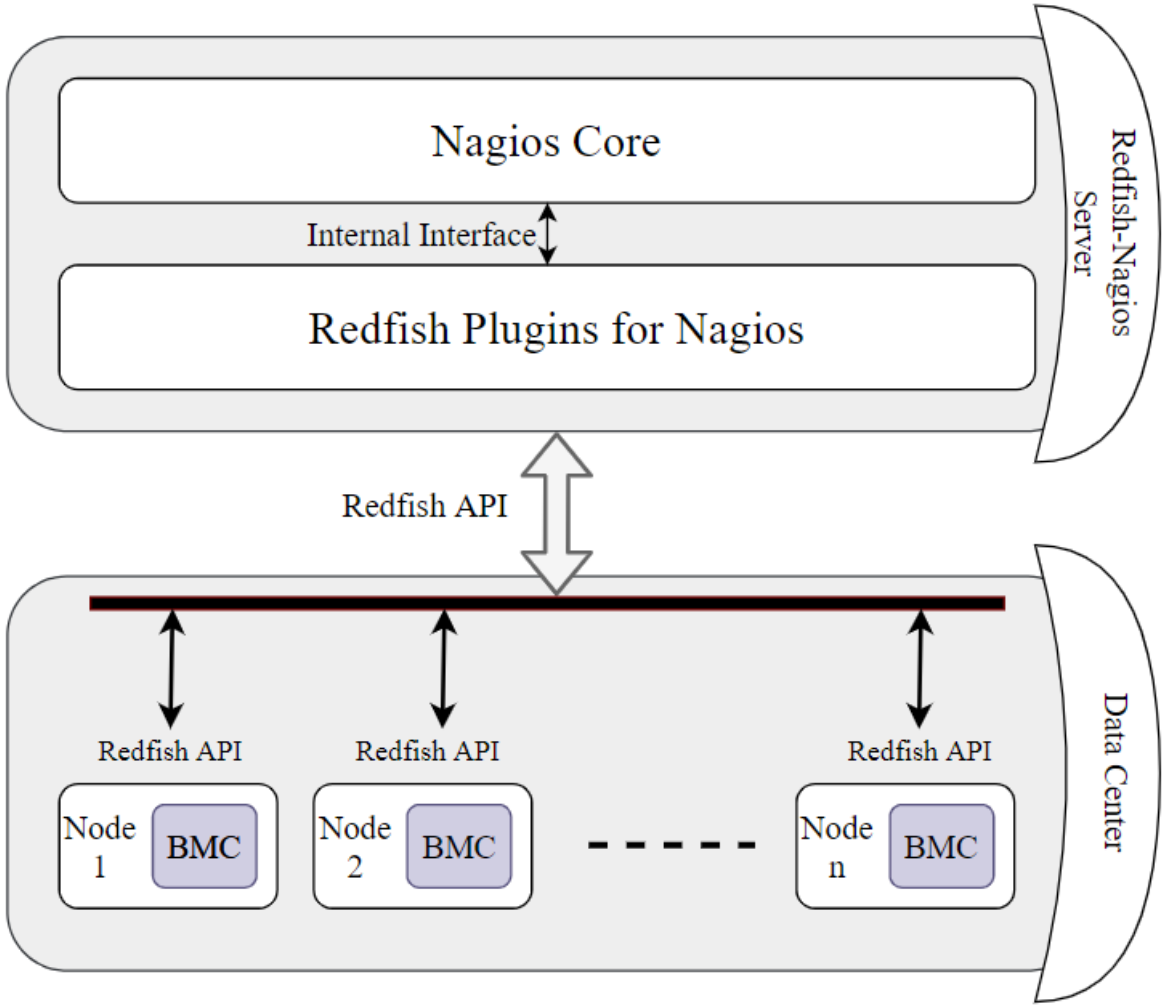
- Using OOB paradigm, monitoring server bypasses the node's OS and directly communicates with the node's BMC via OOB protocols (IPMI/Redfish)



Redfish-Nagios Architecture

Redfish-Nagios framework consists of:

- Nagios core, a key component of the Nagios framework, which performs check scheduling, check execution, event handling, alert, etc.)
- Redfish plugins (abstraction, aggregation, etc.)
- Redfish-enabled nodes (and other devices, e.g., PDU (Power Distribution Unit), etc.)
- Nagios core communicates with Redfish plugins via an internal interface and plugins communicate with the monitored nodes via Redfish



Integration of Redfish and Nagios

- Redfish plugins can be grouped into two types:
  - Health monitoring - `check_host`, `check_CPU`, `check_memory` and `check_BMC`
  - Numeric data – `check_fans`, `check_temperature` and `check_power_usage`
- Inter-working between Nagios and Redfish:
  - When the monitoring data denotes a health status of a resource, the Redfish state is directly mapped with the Nagios state
  - When the monitoring data is a numeric value, the value is translated to one of three Nagios states

Redfish Plugins for Nagios [3]

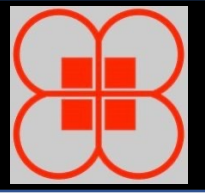
Plugin Name	Description
check_BMC	Acquires BMC health
check_host	Acquires node health
check_CPU	Acquires CPU health
check_memory	Acquires memory health
check_fans	Acquires fan health & speed
check_temperature	Acquires CPU temperature
check_power_usage	Acquires node power usage

Inter-working Between Nagios and Redfish States

Redfish Status	Nagios Status	Description
Ok	OK	Working correctly
Warning	WARNING	Working, but needs attention
Critical	CRITICAL	Not working correctly or requires attention
Unknown	UNKNOWN	Plugin was unable to determine the status

[3] Ghazanfar Ali. *Nagios Redfish API Integration: Out-of-band (BMC) based Monitoring.* Retrieved May, 2022 from https://github.com/nsfcac/Nagios-Redfish-API-Integration

Implementation and Deployment of Redfish-Nagios



- We used the Quanah cluster with 467 Redfish-enabled nodes at High Performance Computing Center of Texas Tech as an infrastructure for implementation and deployment
- Each node is based on the Intel Xeon processor architecture and consists of 36 cores
- BMC uses the integrated Dell Remote Access Controller (iDRAC) 8, which implements the Redfish API
- The operating system of the compute nodes was Linux CentOS 7.6 (now CentOS 8.1)
- The Redfish-Nagios monitoring server specs are described in the table below:

CPU:	2 x 4 cores Intel Xeon(R) E5540 @ 2.53GHz
RAM:	23 GB DDR3
STORAGE:	2TB HDD
NETWORK:	1Gbit/s, Broadcom NetXtreme II

TEXAS TECH UNIVERSITY

DISCL

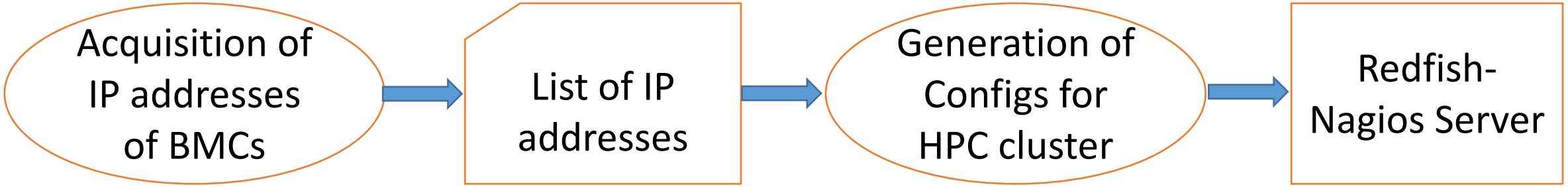
NSF CAC Cloud and Autonomic Computing Center

DELL Technologies

Automating Configuration for Monitoring

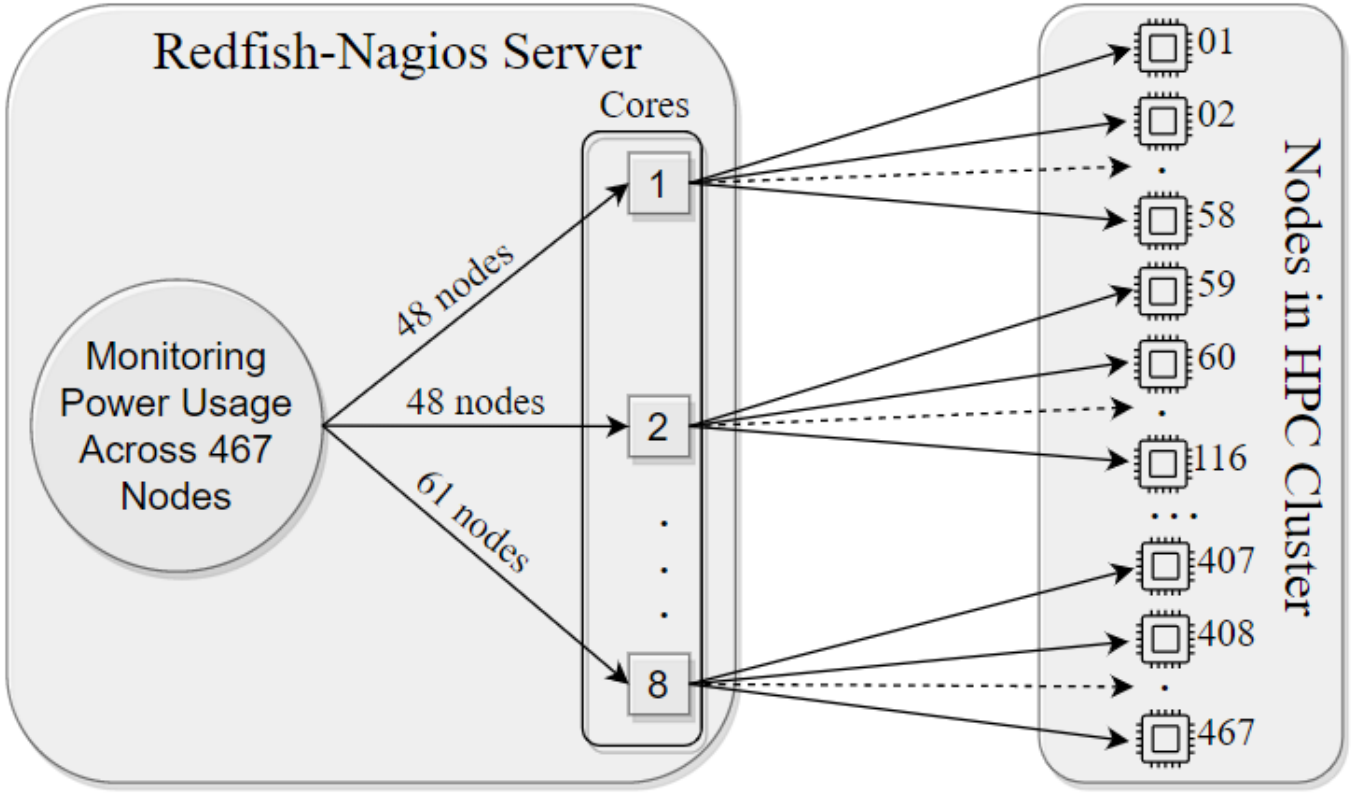
Nagios-based monitoring requires configuration settings (e.g., IP addresses, node name) of the BMC of each monitored node. We automate this process as follows:

- First, the lists of node names and IP addresses of BMCs are acquired
- Second, a script was developed and used to generate configurations of nodes in hosts.cfg file

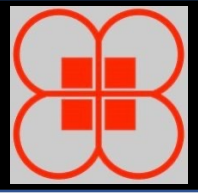


Load Balancing

- To make monitoring service efficient and load balanced, the monitoring workload is distributed among the available cores
- Concurrent Redfish requests for monitoring power usage for 467 nodes in the cluster
- E.g., seven cores handle 58 requests each and 8^{th} core handles remaining 61 requests



Component Level Monitoring



- Component level monitoring enable checking node's individual components
- Our implementation tracks seven components via Redfish and two from UGE:
  - `bmc_health`, `cpu_health`, `cpu_temperature`, `cpu_usage` (UGE), `fan_health`, `fan_speed`, `memory_health`, `memory_usage` (UGE), `system_power_usage`

Host	Service	Status	Last Check	Duration	Attempt	Status Information
compute-1-1	bmc_health	OK	03-08-2019 11:44:15	14d 18h 34m 12s	1/4	OK - BMC is OK!
	cpu_health	OK	03-08-2019 11:44:03	14d 18h 34m 24s	1/4	OK - CPU is OK!
	cpu_temperature	OK	03-08-2019 11:43:39	9d 14h 28m 11s	1/4	{'CPU2 Temp': 54, 'Inlet Temp': 21, 'CPU1 Temp': 69, 'GET_processing_time': 4.77, 'retry': 0}
	cpu_usage	OK	02-21-2019 22:30:41	14d 14h 22m 6s	1/4	CPU usage is: 0.500139
	fan_health	OK	03-08-2019 11:43:39	9d 14h 28m 11s	1/4	{'FAN_3': 'OK', 'FAN_2': 'OK', 'GET_processing_time': 4.77, 'retry': 0, 'FAN_1': 'OK', 'FAN_4': 'OK'}
	fan_speed	OK	03-08-2019 11:43:39	9d 14h 28m 11s	1/4	{'FAN_3': 9380, 'FAN_2': 9450, 'GET_processing_time': 4.77, 'retry': 0, 'FAN_1': 9380, 'FAN_4': 9450}
	memory_health	OK	03-08-2019 11:44:03	21d 13h 39m 25s	1/4	OK - Memory is OK!
	memory_usage	OK	02-21-2019 22:30:41	21d 11h 52m 30s	1/4	Total Memory: 191.908G Used Memory: 31908.0 Avaiable Memory: 160.000G
	system_power_usage	OK	03-08-2019 11:43:51	9d 14h 54m 16s	1/4	Power usage (Watts): 301

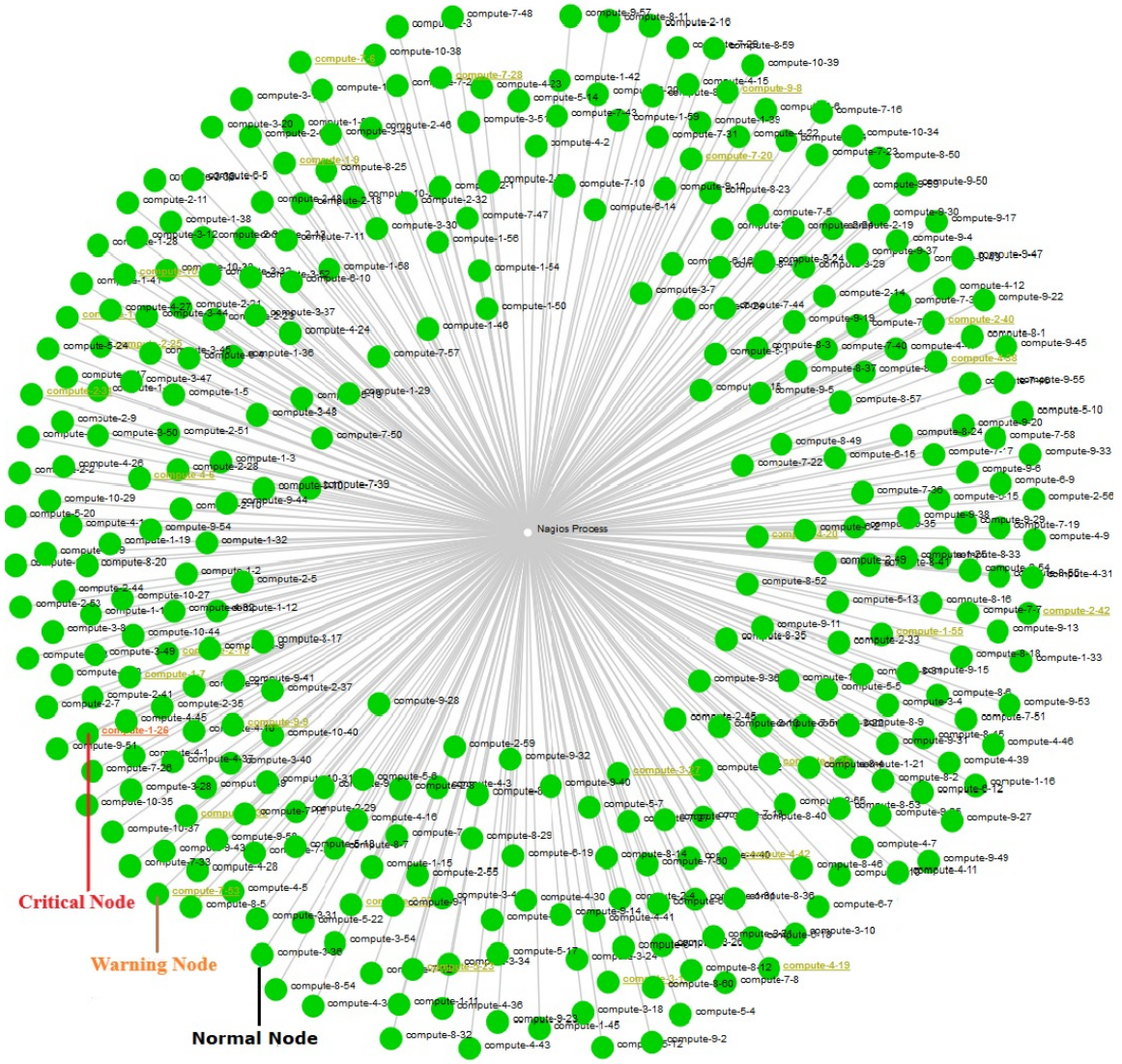
Node Level Monitoring

- At node level:
  - Redfish provides node status as `Ok`, `Warning`, or `Critical`
  - Nagios shows node status as `UP` or `DOWN`
  - Redfish `Ok` and `Warning` are translated as Nagios `UP`, and Redfish `Critical` is translated as Nagios `DOWN`

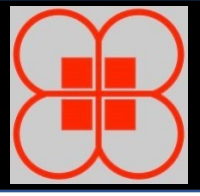
Host	Status	Last Check	Duration	Status Information
compute-1-1	UP	03-06-2019 23:20:14	13d 5h 6m 43s	OK - Host is UP!
compute-1-10	UP	03-06-2019 23:20:14	13d 5h 6m 43s	OK - Host is UP!
compute-1-11	UP	03-06-2019 23:20:14	13d 5h 6m 43s	OK - Host is UP!
compute-1-12	UP	03-06-2019 23:20:14	13d 5h 6m 43s	OK - Host is UP!
compute-1-13	UP	03-06-2019 23:20:14	13d 5h 6m 43s	OK - Host is UP!
compute-1-14	UP	03-06-2019 23:20:14	13d 5h 6m 43s	OK - Host is UP!
compute-1-15	DOWN	03-06-2019 23:20:14	20d 1h 0m 32s	CRITICAL - Host needs immediate attention!
compute-1-16	UP	03-06-2019 23:20:14	13d 5h 6m 43s	OK - Host is UP!
compute-1-17	UP	03-06-2019 23:20:14	13d 5h 6m 43s	OK - Host is UP!
compute-1-18	UP	03-06-2019 23:20:14	13d 5h 6m 43s	OK - Host is UP!
compute-1-19	UP	03-06-2019 23:20:14	13d 5h 6m 43s	OK - Host is UP!
compute-1-2	UP	03-06-2019 23:20:14	13d 5h 6m 43s	OK - Host is UP!
compute-1-20	UP	03-06-2019 23:20:14	13d 5h 6m 43s	OK - Host is UP!
compute-1-21	UP	03-06-2019 23:20:14	13d 5h 6m 43s	OK - Host is UP!
compute-1-22	UP	03-06-2019 23:20:14	13d 5h 6m 43s	OK - Host is UP!
compute-1-23	UP	03-06-2019 23:20:14	13d 5h 6m 43s	OK - Host is UP!
compute-1-24	UP	03-06-2019 23:20:14	13d 5h 6m 43s	OK - Host is UP!
compute-1-25	UP	03-06-2019 23:20:14	8d 1h 41m 2s	OK - Host is UP!

Cluster Level Monitoring

- Cluster level visualization provides high-level view of the cluster in terms of nodes' status



Summary

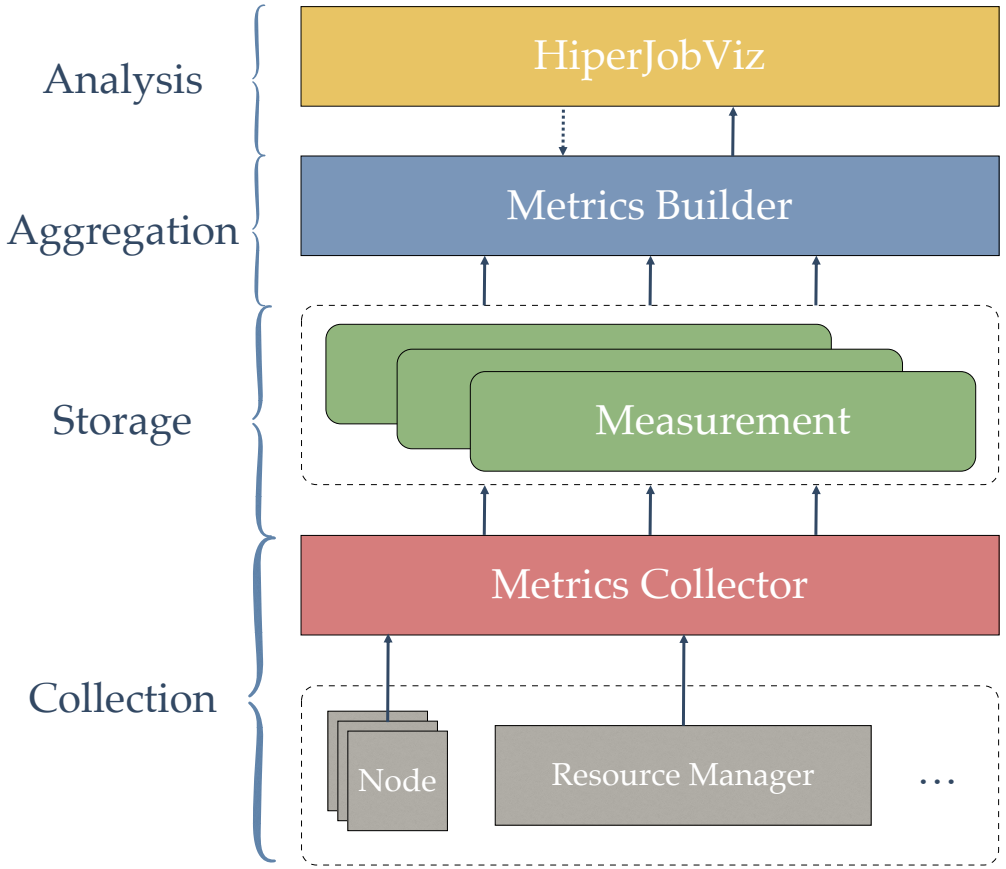


- The current Nagios monitoring tool is not efficient for modern data centers due to shortcomings originating from its in-band nature

- These inadequacies arise from Nagios protocols including:
  - Requirement of monitoring specific in-band agents and plugins on the monitored nodes
  - Consumption of computational resources of the monitored node
  - Cumbersome manual configuration of the monitored nodes

- We developed the Redfish-Nagios integration method, which:
  - Enables Nagios to monitor HPC/cloud systems via BMC using out-of-band Redfish API
  - Reduces the requirement of setting up any Nagios protocol, plugin, or agent
  - Reduces compute nodes' burden by shifting monitoring functions from the OS to the BMC

Ongoing Research and Development

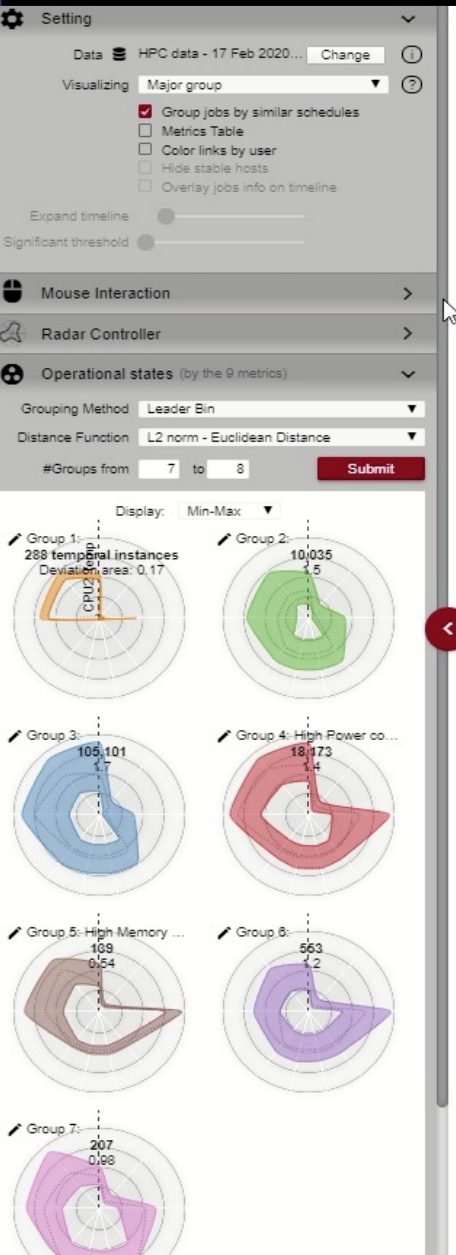


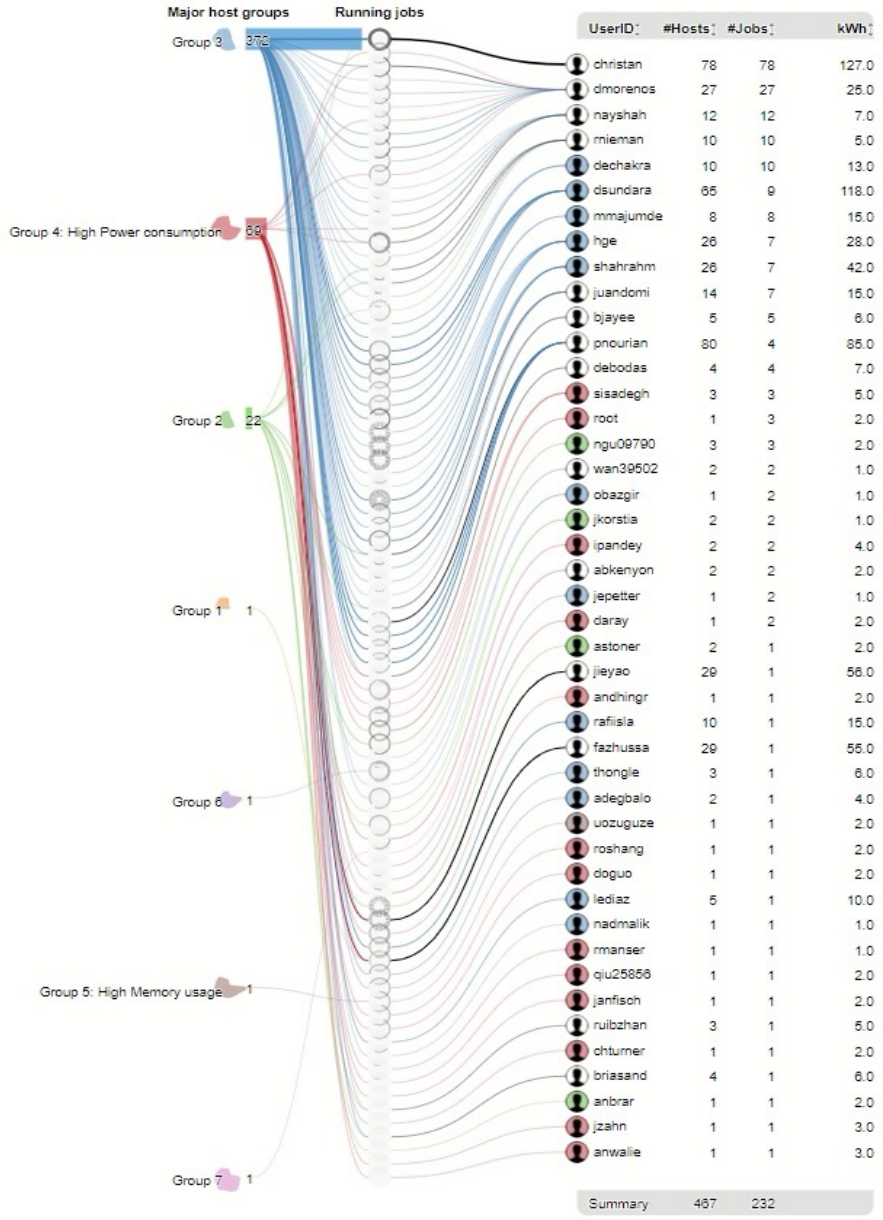
- Visualizes and analyzes job data and system status
- Accelerates the data retrieval
- Correlates and aggregates data
- Exposes an API to various consumers
- A set of measurements for storing the collected data in a time series database
- Captures interesting data
- Nodes equipped with Baseboard Management Controllers (BMCs)

https://github.com/nsfcac/MonSter

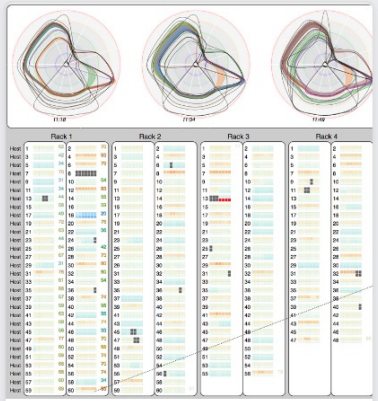
J. Li, G. Ali, N. Nguyen, J. Hass, A. Sill, T. Dang and Y. Chen. MonSTer: An Out-of-the-Box Monitoring Tool for High Performance Computing Systems, In Proceedings of IEEE International Conference on Cluster Computing (CLUSTER'20), Pages: 119 - 129.

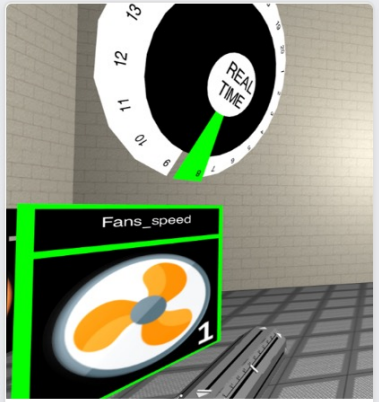
Ongoing Research and Development (cont.)



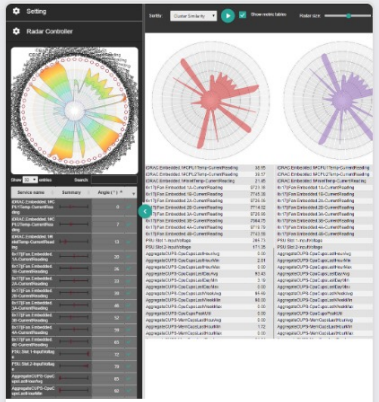


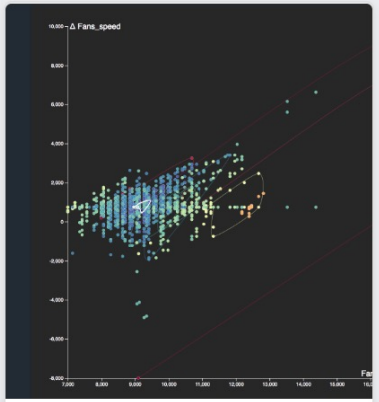
Ongoing Research and Development (cont.)

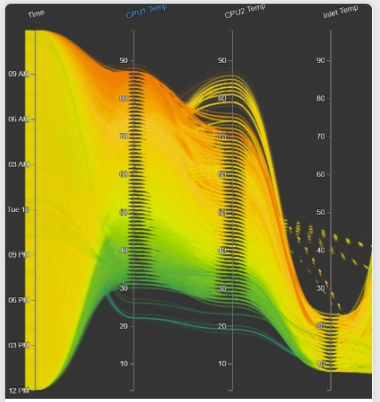

Hiperview

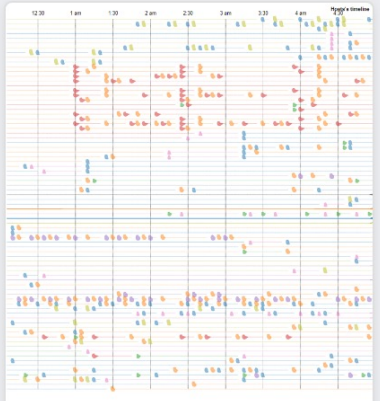


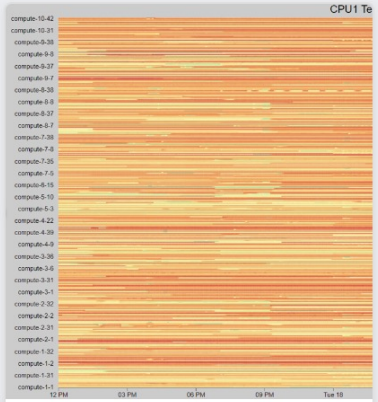
HiperVR

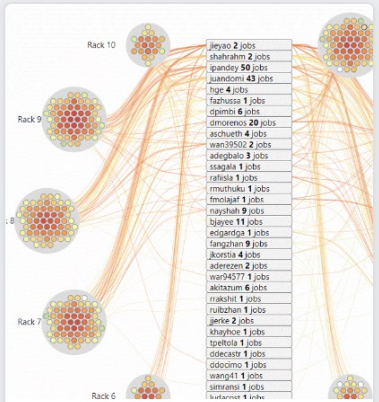

ClusterView

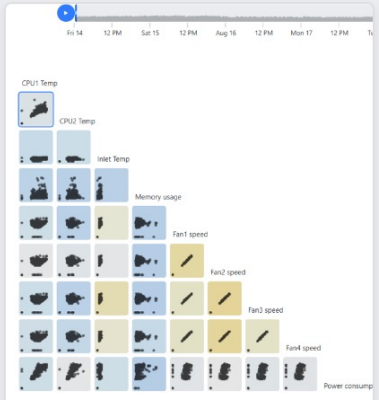

Phase Space

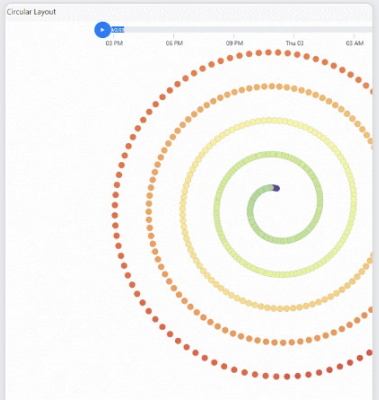

ParallelCoordinates


TimeRadar

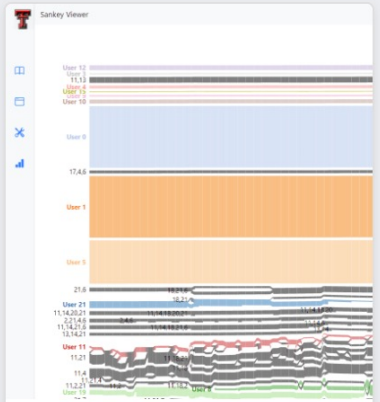

Heatmap


JobViewer


Scagnostics Viewer


Spiral Layout


JobSwarm


SankeyViewer

https://idatavisualizationlab.github.io/HPCC/

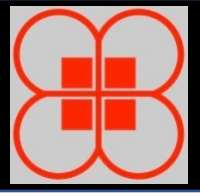
TEXAS TECH UNIVERSITY

DISCL

NSF CAC Cloud and Autonomic Computing Center

DELL Technologies

Thank You and Q&A



For more details, please check out our paper and repos:

https://github.com/nsfcac/Nagios-Redfish-API-Integration

https://github.com/nsfcac/MonSter

https://idatavisualizationlab.github.io/HPCC/

Acknowledgment: This research is supported in part by Dell Technologies and the National Science Foundation under grant CNS-1939140 (A U.S. National Science Foundation Industry-University Cooperative Research Center on Cloud and Autonomic Computing) and OAC-1835892. We are also very grateful to the High Performance Computing Center of Texas Tech University for providing HPC resources for this project.

TEXAS TECH UNIVERSITY

DISCL

NSF CAC Cloud and Autonomic Computing Center

DELL Technologies