Monitoring and Data Integration at LLNL

EE HPC Working Group Workshop, November 12, 2018

Kathleen Shoga
shoga1@llnl.gov

This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under contract DE-AC52-07NA27344. Lawrence Livermore National Security, LLC

LLNL-PRES-761254

Lawrence Livermore National Laboratory

LLNL HPC Systems

System (Program)	Processor Architecture	Nodes	Cores	Peak (TFLOP/s)
RZ				
RZCereal (M&IC)	Intel Xeon E5530	21	169	1.6
RZHasGPU	Intel Xeon E5-2667 v3	20	320	8.2
RZMerl (ASC/M&IC)	Intel Xeon E5-2670	162	2,592	53.9
RZSLIC ***	Intel Xeon E5330	3		
RZuSeq (ASC) ****	IBM PowerPC A2	522		
RZZeus (M&IC)	Intel Xeon E5530	267		

System (Program)	Processor Architecture	Nodes	Cores	Peak (TFLOP/s)
CZ				
Ansel (M&IC)	Intel Xeon EP X5660	324	3,888	43.5
Aztec (M&IC)	Intel Xeon EP X5660	96	1,152	12.9
Catalyst (ASC/M&IC) ****	Intel Xeon E5-2695 v2	324	7,776	149.3
Cab (ASC/M&IC)	Intel Xeon E5-2670	1,296	20,736	431.3
	AMD Opteron		256	1.6
			13,216	112.7
			40	–
			23,328	261.3
	Intel Xeon E5-2670	162	2,592	53.9
	Intel Xeon E5-2670	324	5,056	107.8
PCIC)	IBM PowerPC A2	24,576	393,216	5,033

System (Program)	Processor Architecture	Nodes	Cores	Peak (TFLOP/s)
Inca (ASC)	Intel X X566			
Juno (ASC)	AMD 8354			
Max (ASC)	Intel X E5-2670			
Muir (ASC)	Intel Xeon EP X5660	1,296	15,552	174.2
Sequoia (ASC) **	IBM PowerPC A2	98,304	1,572,864	20,132
Zin (ASC)	Intel Xeon E5-2670	2,916	46,656	970.4

Stats

Max: 98,304 nodes in one system (Sequoia)

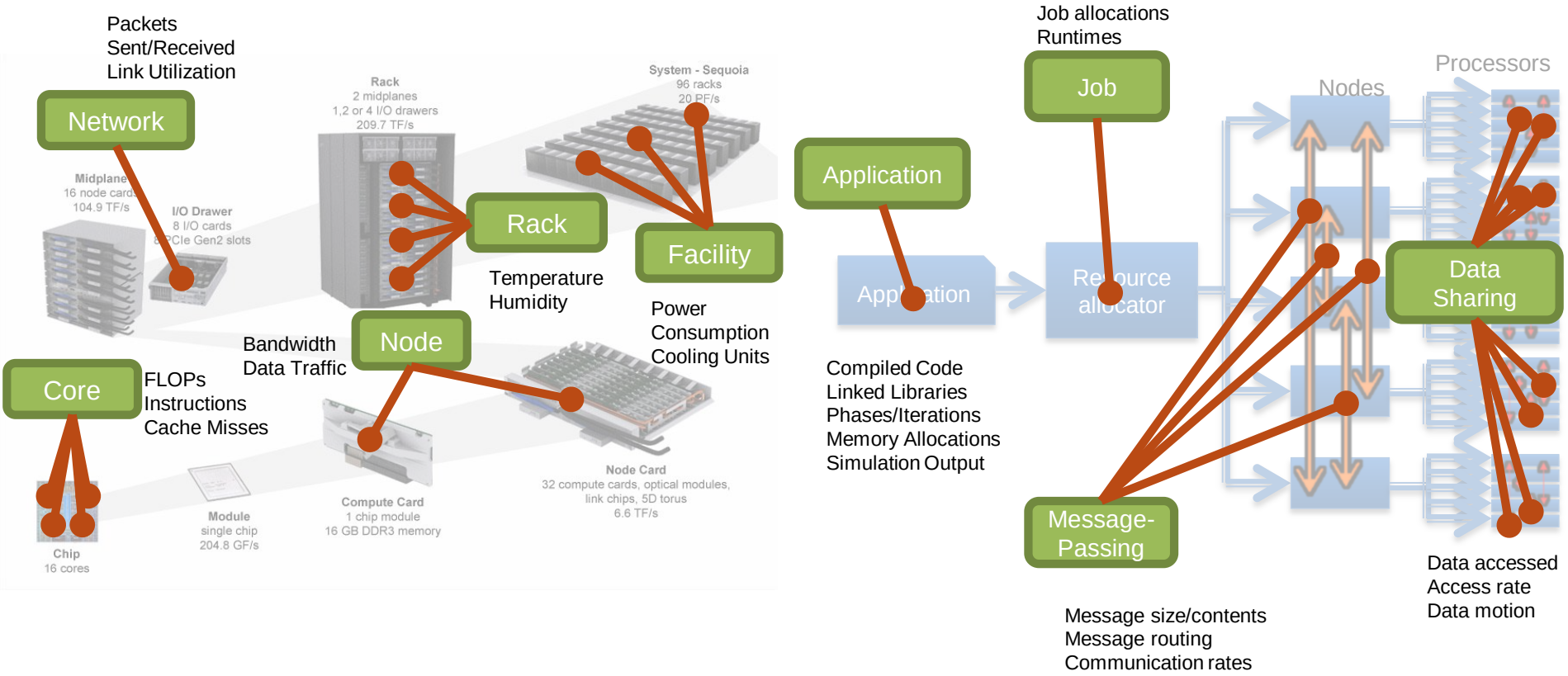
25 systems across open and closed zones

Various processor architectures

Facility Resource Data: Good…

- **Facilities data measured by OSISoft PI**
  - Big-picture power usage
  - Temperature on cluster resources
- **Provides**
  - PI provides on-demand reports
  - Only allows single-variable plots over time, simple correlation
  - Minutes timescale

More HPC Facility Data: Better



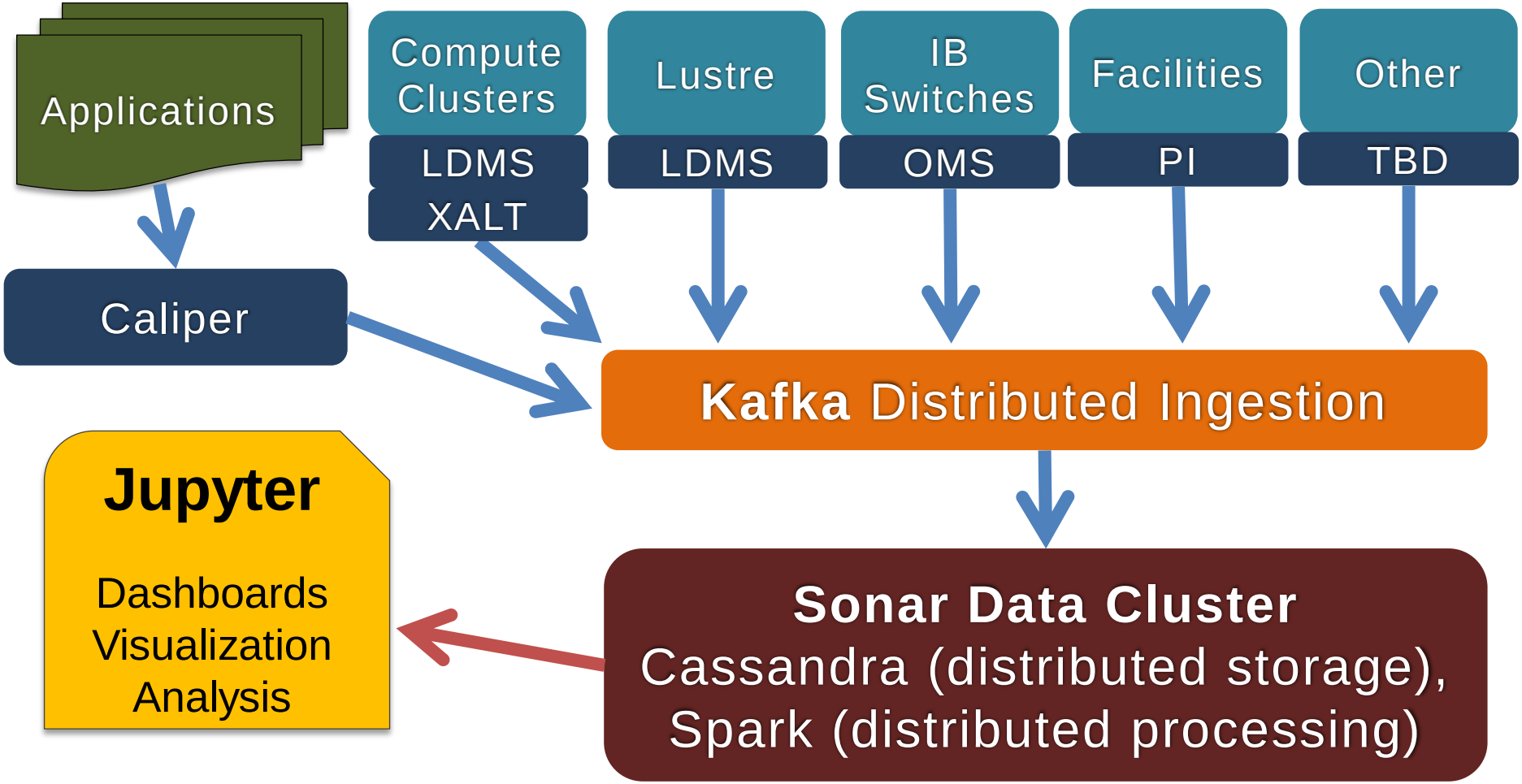
Better Data Driven Decisions

- **Full Picture View Allows**
  - Multiple paths to energy/power usage goals
  - Prediction of power usage patterns with more information
  - Scheduling jobs with previous user/job data
  - Monitoring for failures that could cause sudden drops in power draw
- **Challenges**
  - Lots of data in different formats
  - Data in different locations
  - Data collection
  - Permissions for data
  - Analysis of data

Livermore Computing's Solution: Sonar

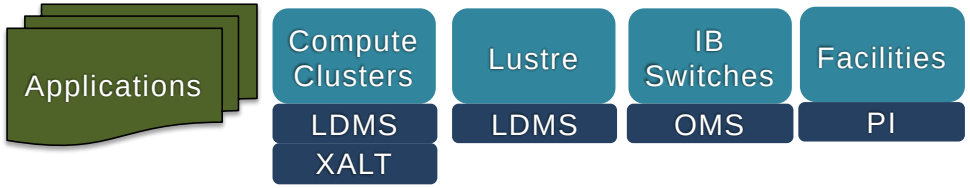
Infrastructure for Analyzing the Performance of our HPC Center

Ingesting Data from Many Sources



Big Data Database: Apache Cassandra

- NoSQL
  - Different types of data ingested



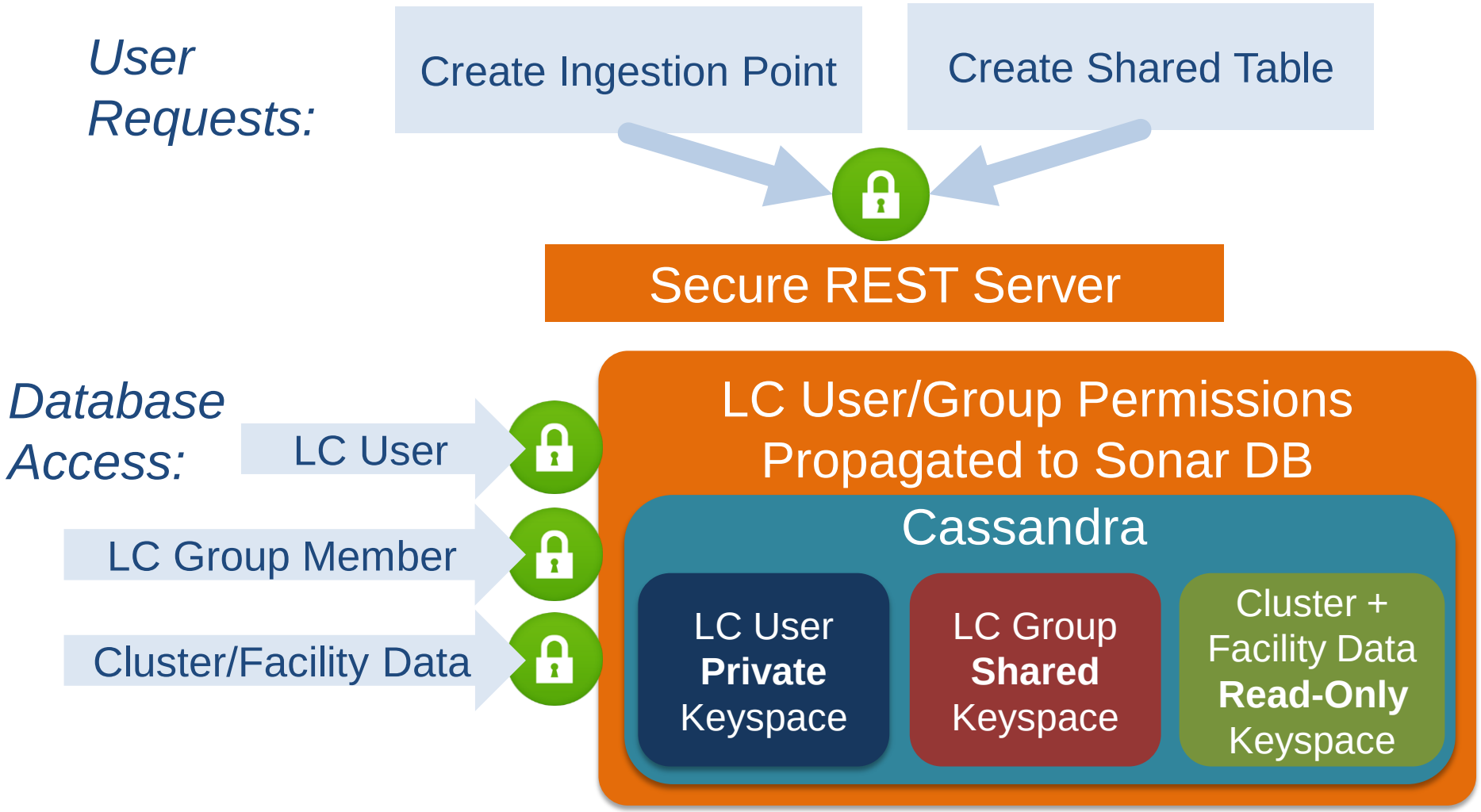
- Scalable
  - Nodes can be added without downtime
  - Add nodes of different architectures
    - Phasing out old nodes for new ones
    - Tune storage and cores

- Decentralized
  - Robust to node failures
    - If one or two nodes go down, the others continue normally

- User/group permissions available at multiple levels built in

- Previously used Hbase, but you had to choose either security or performance!

Security


User Requests:
Create Ingestion Point
Create Shared Table
Secure REST Server
Database Access:
LC User
LC Group Member
Cluster/Facility Data
LC User/Group Permissions Propagated to Sonar DB
Cassandra
LC User Private Keyspace
LC Group Shared Keyspace
Cluster + Facility Data Read-Only Keyspace

Analysis

Jupyter

De facto modern data science workflow

- “code” paragraphs:
  - the code to run on the backend cluster
- “output” paragraphs:
  - code output shown in web-based UI

Sonar: Demo

Sonar: Demo

File Edit View Insert Cell Kernel Widgets Help Trusted Sonar Python3

Run Code Dashboard View:

In [ ]:
```
from sonar_auth.cassandra import SonarCassandraSession
session = SonarCassandraSession(['rzsonar8']).session
```

In [ ]:
```
import pandas as pd
import matplotlib.pyplot as plt

# This lets us read rows directly in to a pandas dataframe
def pandas_factory(colnames, rows):
    return pd.DataFrame(rows, columns=colnames)

session.row_factory = pandas_factory
session.default_fetch_size = None
```

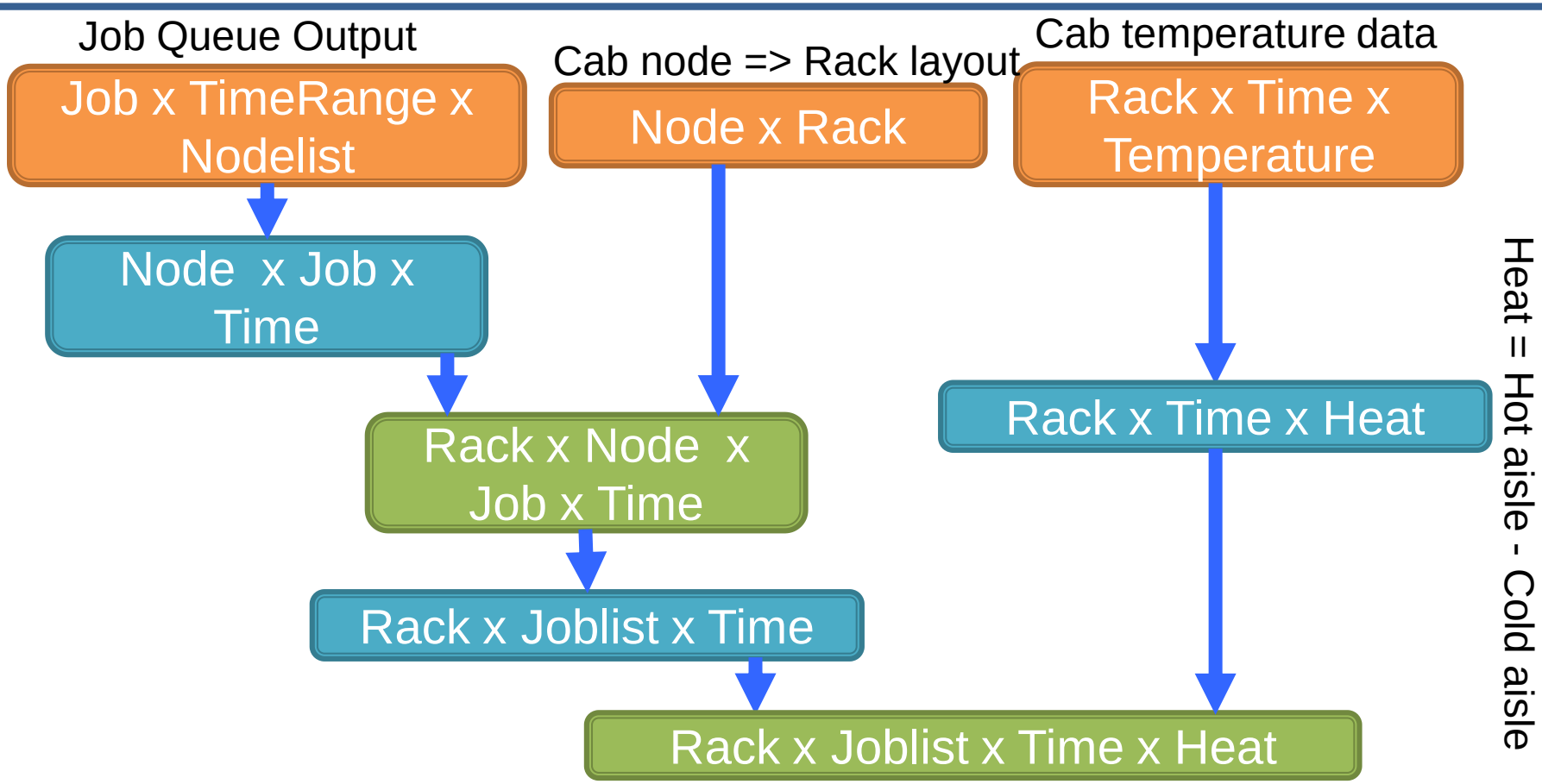
In [ ]:
```
# Run a query and look at the resulting pandas dataframe
query = (
    "SELECT * "
    "FROM lcstaff_k.rzgenie_meminfo "
    "WHERE \"Time\" > 1.5354e+09"
    "LIMIT 100 "
    "ALLOW FILTERING"
)

resultSet = session.execute(query, timeout=None)
totalDataSet = resultSet._current_rows.sort_values('Time')
totalDataSet.describe()
```

In [ ]:
```
# Plot
lines = totalDataSet.plot.line(x='Time', y='MemAvailable')
lines
```

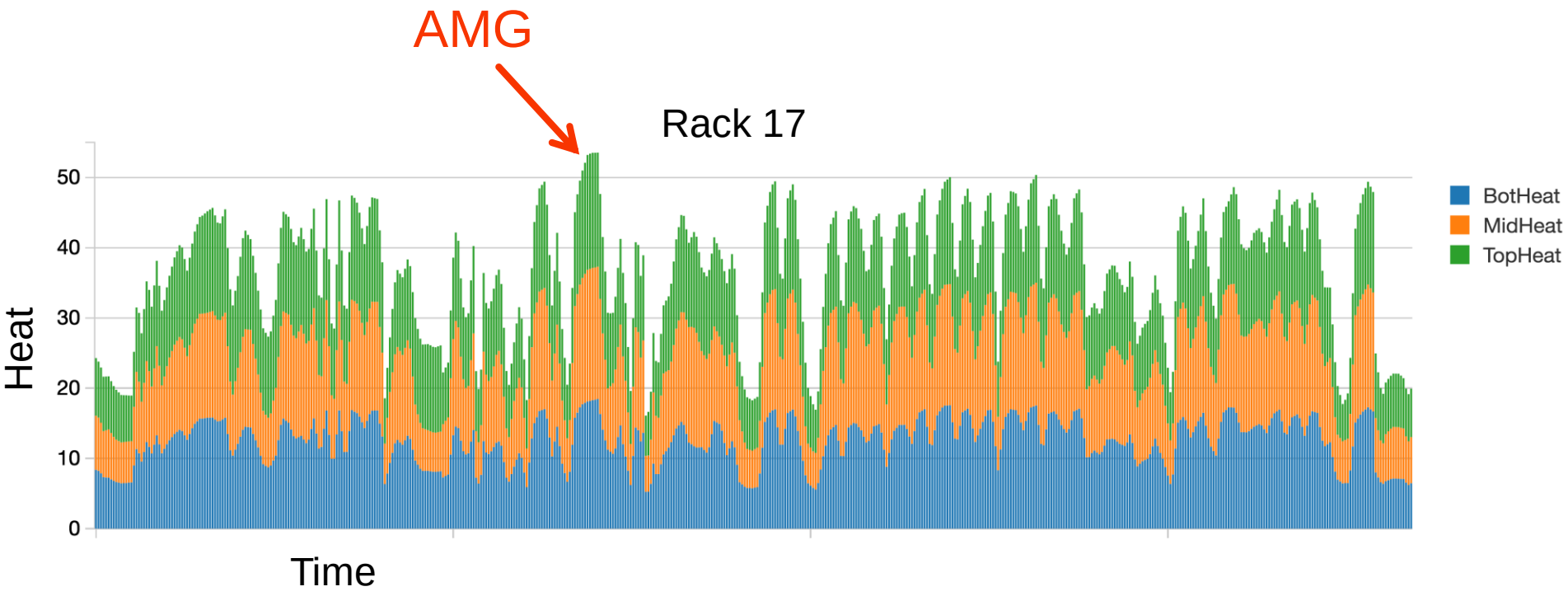
In [ ]:
```
nodes= ['rzgenie%s' % s for s in range(6,49)]
means= []
# Get the means per node
for i in nodes:
    query = (
        "SELECT \"Time\",\"MemAvailable\",\"ProducerName\" "
        "FROM lcstaff_k.rzgenie_meminfo "
        "WHERE \"Time\" > 1.5354e+09 "
        "AND \"ProducerName\"='%s' "
        "LIMIT 10 "
        "ALLOW FILTERING"
    ) %i
    #print(query)
    resultSet = session.execute(query, timeout=None)
    means.append(resultSet._current_rows.sort_values("ProducerName")["MemAvailable"].mean())
```

Data Integration with ScrubJay

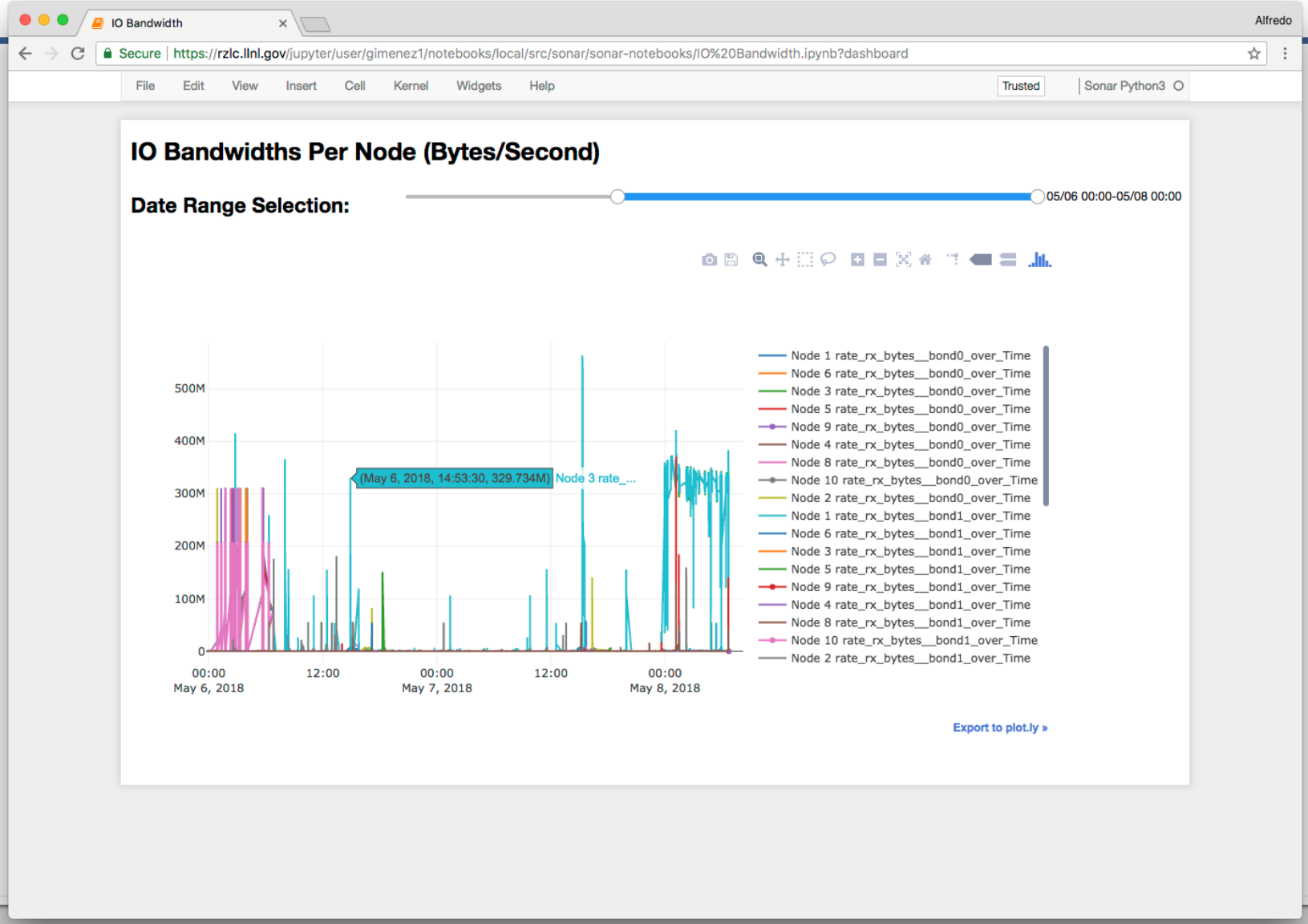


Query: Job X Rack Heat ?

Data Integration - Results



ScrubJay Jupyter Integration



Meet the Team

- **Alfredo Gimenez**
  - Project Co-Lead
  - Kafka
  - ScrubJay
  - Pi Data
  - Analytics
- **Kathleen Shoga**
  - Project Co-Lead
  - Cassandra Database
  - Hardware
  - LDMS
  - Coordinating Data Collection Efforts
- **Todd Gamblin**
  - Project Manager
- **Matt Legendre**
  - Project Manager
- **Tim Meier**
  - Cassandra Security
  - Network Data (OpenSM)
- **Danielle Sikich**
  - Slurm Prolog/Epilog Data
  - XALT
  - Analytics
- **Thomas Mendoza**
  - Jupyter Hub
- **David Poliakoff**
  - Application Monitoring
  - SPOT & Caliper
- **David Boehme**
  - Caliper
- **Joe Chavez**
  - Web Dev SPOT
- **Tapasya Patki**
  - LDMS Power Plugin

Lawrence Livermore National Laboratory

Disclaimer
This document was prepared as an account of work sponsored by an agency of the United States government. Neither the United States government nor Lawrence Livermore National Security, LLC, nor any of their employees makes any warranty, expressed or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States government or Lawrence Livermore National Security, LLC. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States government or Lawrence Livermore National Security, LLC, and shall not be used for advertising or product endorsement purposes.