NSF CSSI PI Meeting, Seattle, WA, Feb. 13-14, 2020

Award #: 1835135

NSCI Elements: Software – PFSTRASE – A Parallel FileSystem TRacing and Analysis SErvice to Enhance Cyberinfrastructure Performance and Reliability

PI: R. Todd Evans
Institutions: Texas Advance Computing Center, The University of Texas at Austin

Introduction

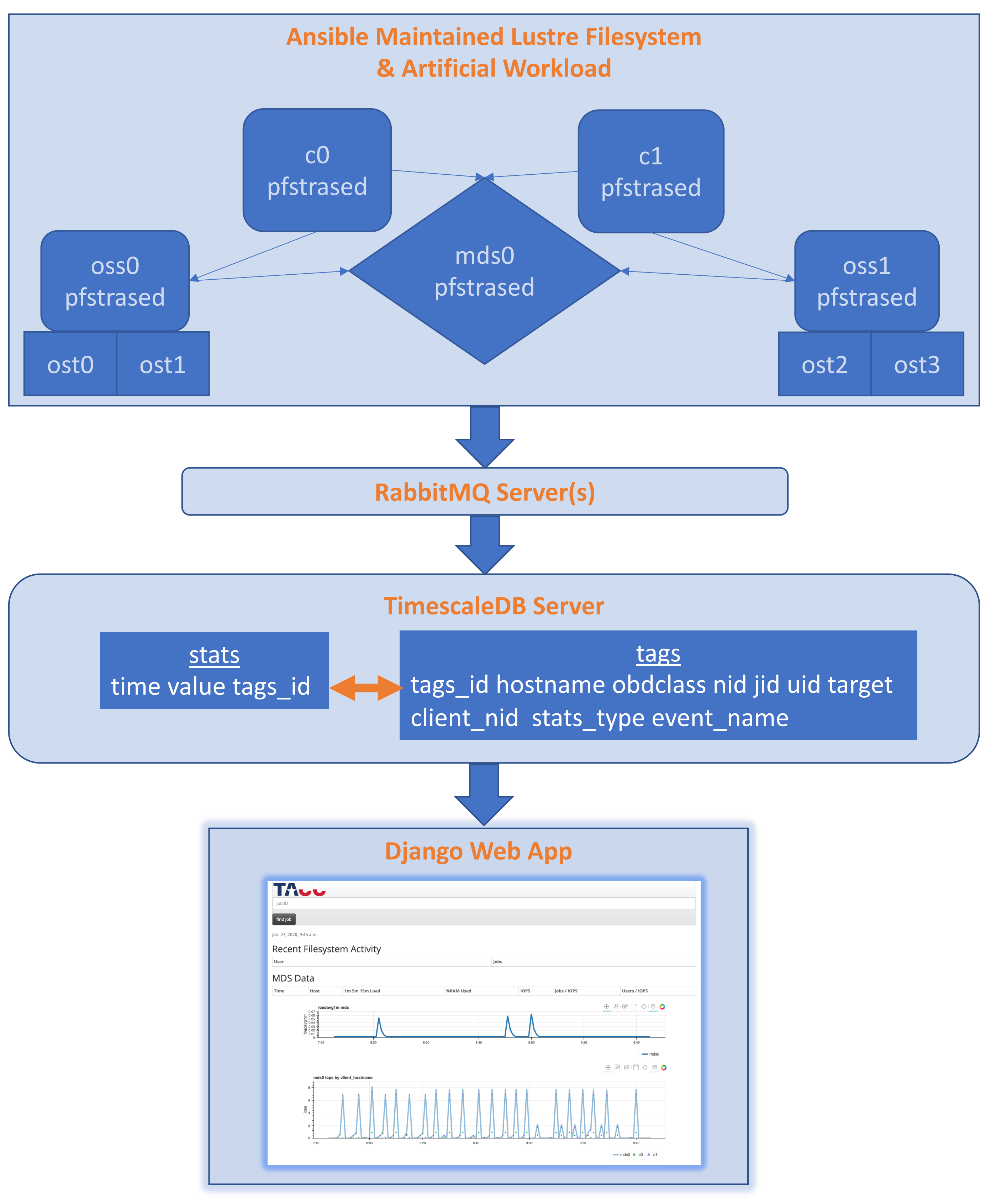
*Goal: Implement, validate, and deploy a Parallel FileSystem Tracing and Analysis SErvice (**PFSTRASE**[1]) that will enhance the reliability and performance of parallel filesystems (PFSs).*

While performance degradation and failure events within the PFS can be incurred by many means and negatively impact every user of a HPC system, there are no tools that precisely and accurately portray the current and past states of the PFS and its health in its entirety to assist in the prevention, diagnosis, and treatment of stability and performance issues. **PFSTRASE** fills this information gap. This project will develop and use **PFSTRASE** as follows:

1) Measure the load incurred on PFS components by various IO operations.
2) Construct a representation that quantitatively traces individual application instances' I/O activities to PFS resource usage
3) Use this representation to monitor PFS performance and stability and classify applications according to file access patters.
4) Tune PFS parameters to achieve greater performance and stability on the overall PFS and on a per application basis.

We will present this information in a graphical interface, displaying the instantaneous and historical state of the filesystem with health threatening activities highlighted. Application instances are to be tagged according to file access pattern type. This provides system administration staff the information they need to quickly resolve PFS issues and to develop best practices for PFS management.

Project Implementation Overview

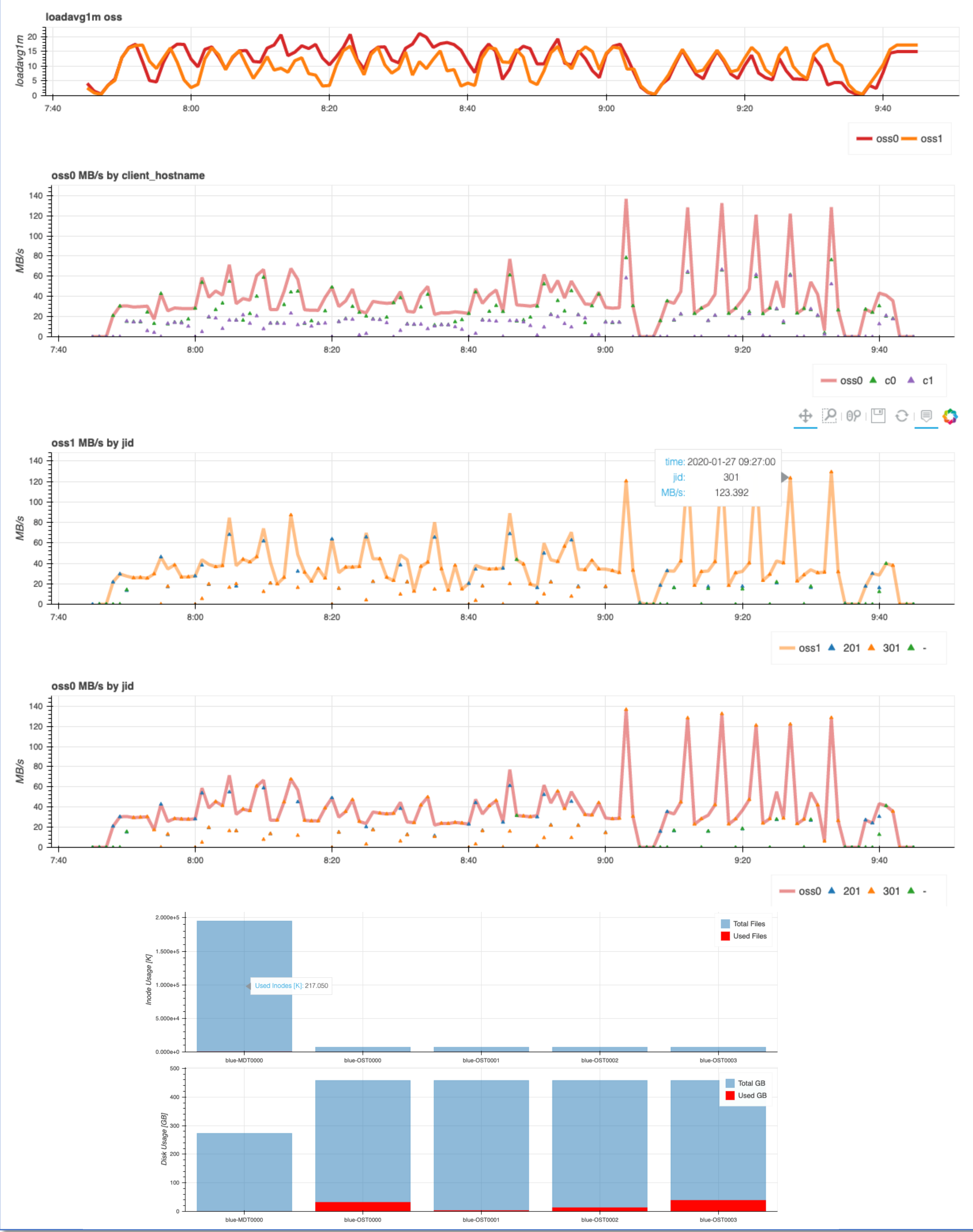


1) Ansible maintained test Lustre filesystem
   - Enables rapid deployment and reconfiguration of filesystem
   - Artificial workload managed by Ansible
2) Light-weight monitoring agent running on PFS servers and clients
   - Modern Linux OS systemd compatible daemon written in C
   - deployable as rpm with minimal dependencies
   - scrapes data from *sysfs* and *procfs*
3) Scalable AMQP-based[2] data aggregation
   - Monitoring agents are connected to RabbitMQ[3] server(s)
   - Messages are json formatted for portability and ease of parsing
4) Scalable, powerful, open-source time-series SQL database for storage
   - TimescaleDB[4] extension to PostgresQL that supports SQL natively
   - Time and space partitioning enable parallelized queries and scaling
   - Time-series oriented indexing optimizes read/write rates
5) Open-source web-browser for data exploration and analytics interface
   - Django[5] web framework
   - psycopg2 and Pandas[6] provide the interface to TimescaleDB
   - Bokeh[6] generated JavaScript dynamic plots enables exploration

Project Status

*The **PFSTRASE** prototype has been implemented and validated in a testbed system. This is the only Open-Source PFS monitoring solution that treats multiple HPC relevant meta-data tags (nodes, job ids, users) as first-class enabling seamless aggregations across tags.*

1) An Ansible maintained Lustre filesystem testbed is in place
   - Filesystem parameters can be rapidly reconfigured.
   - Artificial, known workloads are run continuously on the filesystem.
2) The systemd integrated monitoring daemon, *pfstrased*, has been extensively tested on all components.
   - Load is negligible at 60s samples w/ 3MB memory footprint.
   - client data tagged with jid and uid
   - Program can be rapidly modified and redeployed
3) RabbitMQ server is in place with shim to send data to TimescaleDB.
4) Database model designed in this project supports ephemeral data tags (job ids, usernames, event types etc.) in combination with permanent time-series data streams.
   - enables scalable aggregation of data across arbitrary sets of nodes, jobs and/or users. TimeScaleDB has stored 2 months of test system data using ~130GB.
5) Filesystem data is accessible through web interface:



Next Steps and Challenges

PFSTRASE must be deployed in a production system and used to:
- Correlate filesystem load with iops and bandwidths
- Expand data collection to include read and write sizes, page cache hit rates, and read ahead stats
- Build out website to allow easier and more flexible data exploration
- Build out website to highlight filesystem stressors
- Define and template file access patterns
- Tag applications by file access patterns

Reference and Contact Information

Contact Information: R. Todd Evans, rtevans@tacc.utexas.edu

References:

[1] Evans, R. T. (2020). A repository for pfstrase related scripts and source code [Software]. Retrieved from https://github.com/TACC/pfstrase
[2] Advanced Message Queuing Protocol (AMQP) www.amqp.com
[3] Pivotal Software, Inc. (2019). RabbitMQ [Software]. www.rabbitmq.com
[4] Timescale, Inc. (2020). TimescaleDB [Software]. www.timescale.com
[5] Django Softare Foundation (2020). django [software]. www.djangoproject.com
[6] Python Software Foundation (2020). Python Package Index [software]. www.pypi.org