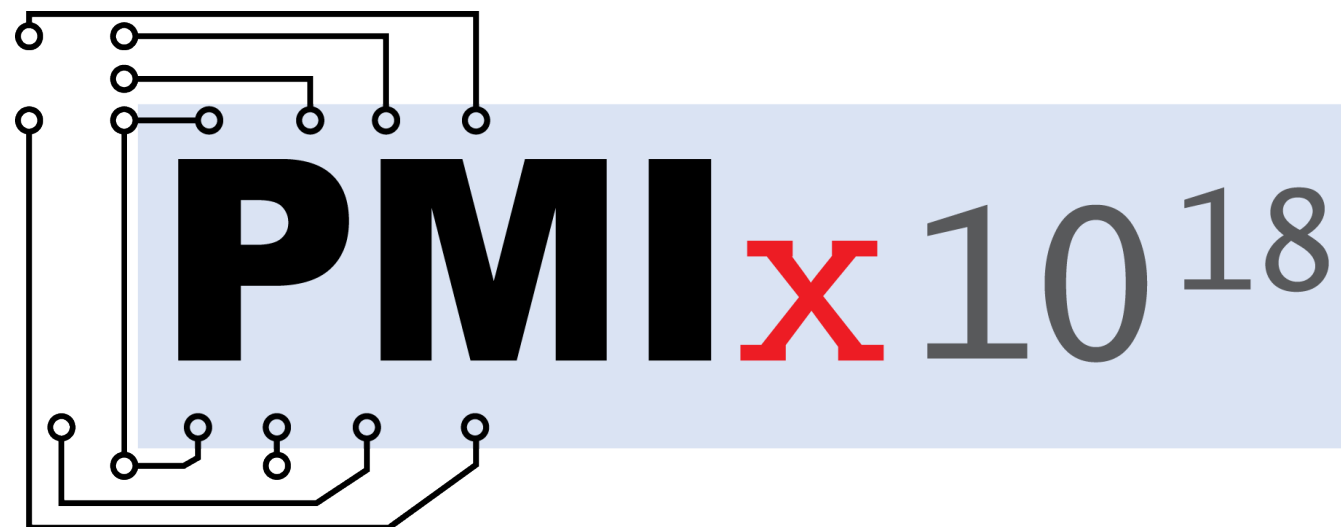


PMIx: Process Management for Exascale Environments



Agenda

- State of the Community
 - Ralph H. Castain (Intel)
- Scaled Performance
 - Aurelien Bouteiller (UTK)
- PMIx Standards Document
 - Josh Hursey (IBM)
- Q&A

Three Distinct Entities

- **PMIx Standard**

- Defined set of APIs, attribute strings
- Nothing about implementation

*Standards
Doc under
development!*

- **PMIx Reference Library**

- A full-featured implementation of the Standard
- Intended to ease adoption

- **PMIx Reference Server**

- Full-featured “shim” to a non-PMIx RM
- Provides development environment

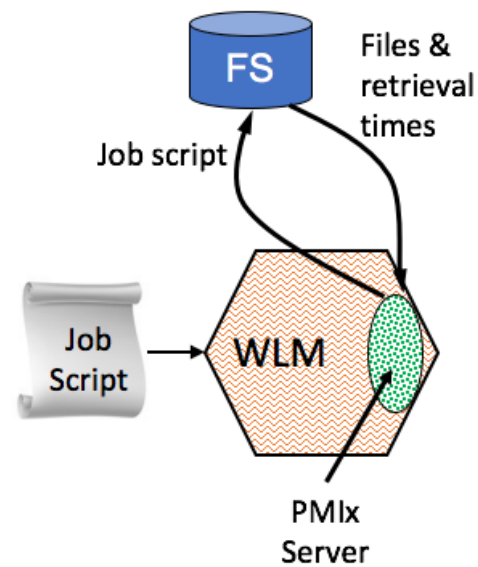
Where Are We?

- Launch scaling
 - Wireup enhancements complete
 - Fabric “instant on” enablement underway
- Results
 - Tracks spawn propagation time
 - Exascale in < 5 seconds
 - 3rd party confirmation

[EuroMPI 2017 Overview Article*](#)

*extended version to be published

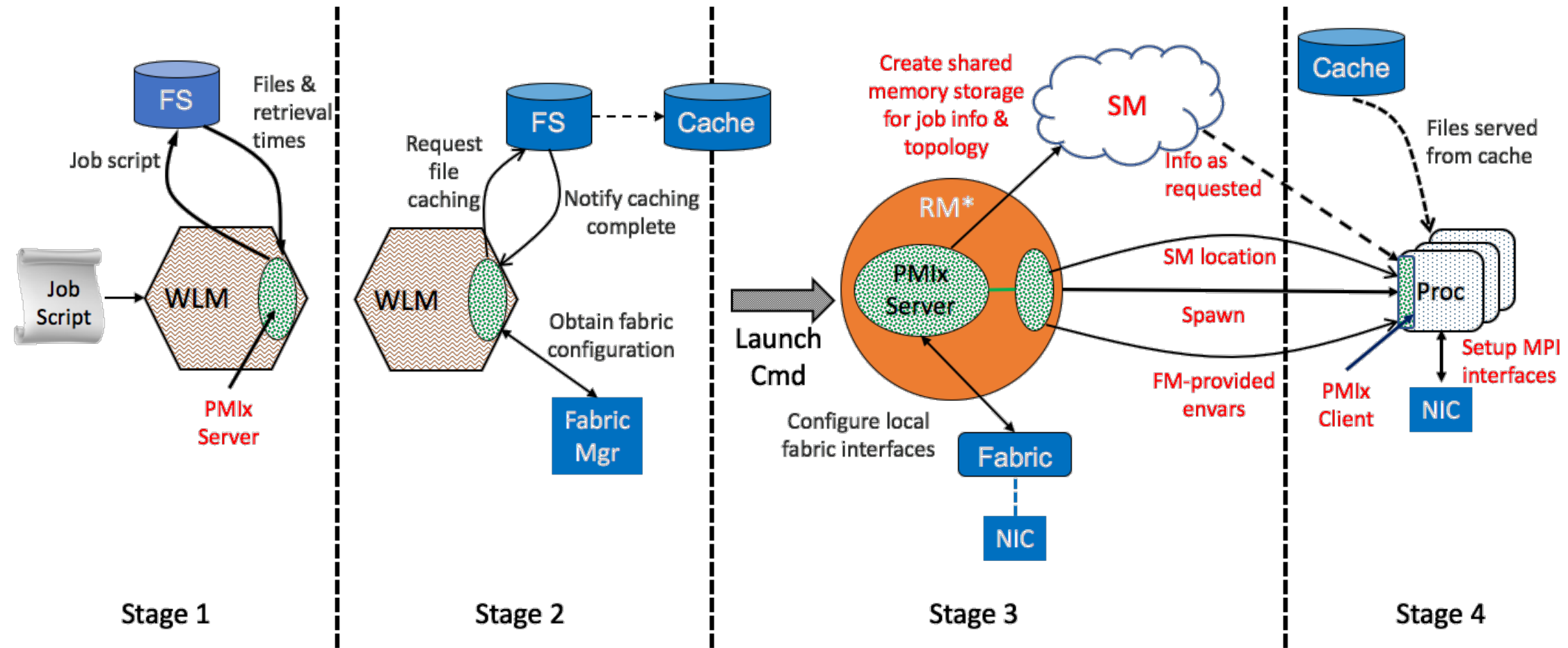
PMix Launch Sequence



Stage 1

*RM daemon, mpirun-daemon, etc.

PMix Launch Sequence



■ Completed

*RM daemon, mpirun-daemon, etc.

Current Support (I)

- Typical startup operations
 - Put, get, commit, barrier, spawn, [dis]connect, publish/lookup
- Tool connections
 - Debugger, job submission, query
- Generalized query support
 - Job status, layout, system data, resource availability
- Event notification
 - App, system generated
 - Subscribe, chained
 - Preemption, failures, timeout warning, ...
- Logging
 - Status reports, error output
- Flexible allocations
 - Release resources, request resources

Current Support (II)

- Network support
 - Security keys, pre-spawn local driver setup
- Obsolescence protection
 - Automatic cross-version compatibility
 - Container support
- Job control
 - Pause, kill, signal, heartbeat, resilience support (C/R coordination)
- Async definition of process groups
 - Rolling startup/teardown



Singularity portability

Singularity & PMIx: Exact matching version

<u>Container OpenMPI</u>	<u>Host OMPI mpirun version</u>				
	1.2.5	1.2.5	1.2.5	1.2.5	2.1
	2.0.0	2.0.1	2.0.3	2.1.1	3.0.0
2.0.0	✓	✗	✗	✗	✓
2.0.1	✗	✓	✗	✗	✓
2.0.3	✗	✗	✓	✗	✓
2.1.1	✗	✗	✗	✓	✓
3.0.0	✓	✓	✓	✓	✓

PMIX version

*Should be all ✓
going forward*

Courtesy of:
V́ctor Sande Veiga
Supercomputing Center of Galicia

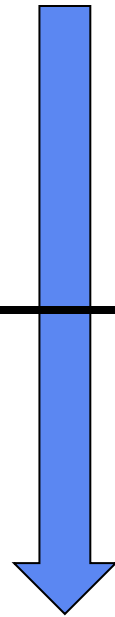
Cross-Version Interoperability

- PMIx v1.1.5
- PMIx v1.2.5⁺
- PMIx v2.0.3⁺

- PMIx v2.1.x
- PMIx v3.0.x

Server \geq Client

*Any client/server
combination*



In Pipeline

- Network support
 - Fabric topology and status, traffic reports, fabric manager interaction
- MPI Sessions support
 - Rolling startup/teardown
- Generalized data store
 - Distributed key-value storage
- Security
 - Obtain and validate credentials for application/SMS
- Power directives
- File system support
 - Dependency detection
 - Tiered storage caching strategies
- Debugger/tool support⁺⁺
 - Automatic rendezvous
 - Single interface to all launchers
 - Co-launch daemons
 - Access fabric info, etc.
- Cross-library interoperation
 - OpenMP/MPI coordination

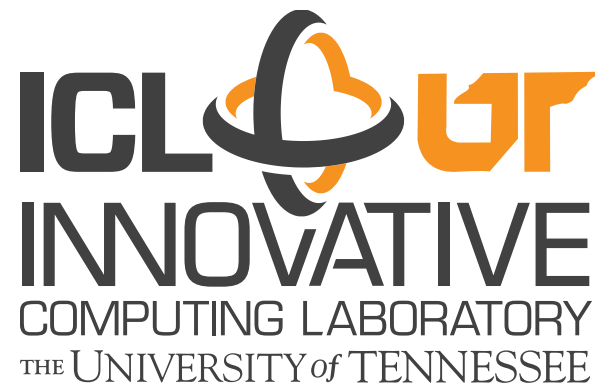
■ Nearing completion ■ Ramping up ■ Idling

Adoption

- RMs
 - SLURM, IBM's Job Step Manager (JSM) complete
 - Fujitsu underway, Altair ramping up
- Libraries
 - Open MPI, OSHMEM, SOS complete
 - GASNet, ORNLshmem – in PR
 - MPICH to come (1H2018?)
- Tools
 - Debugger integration under development

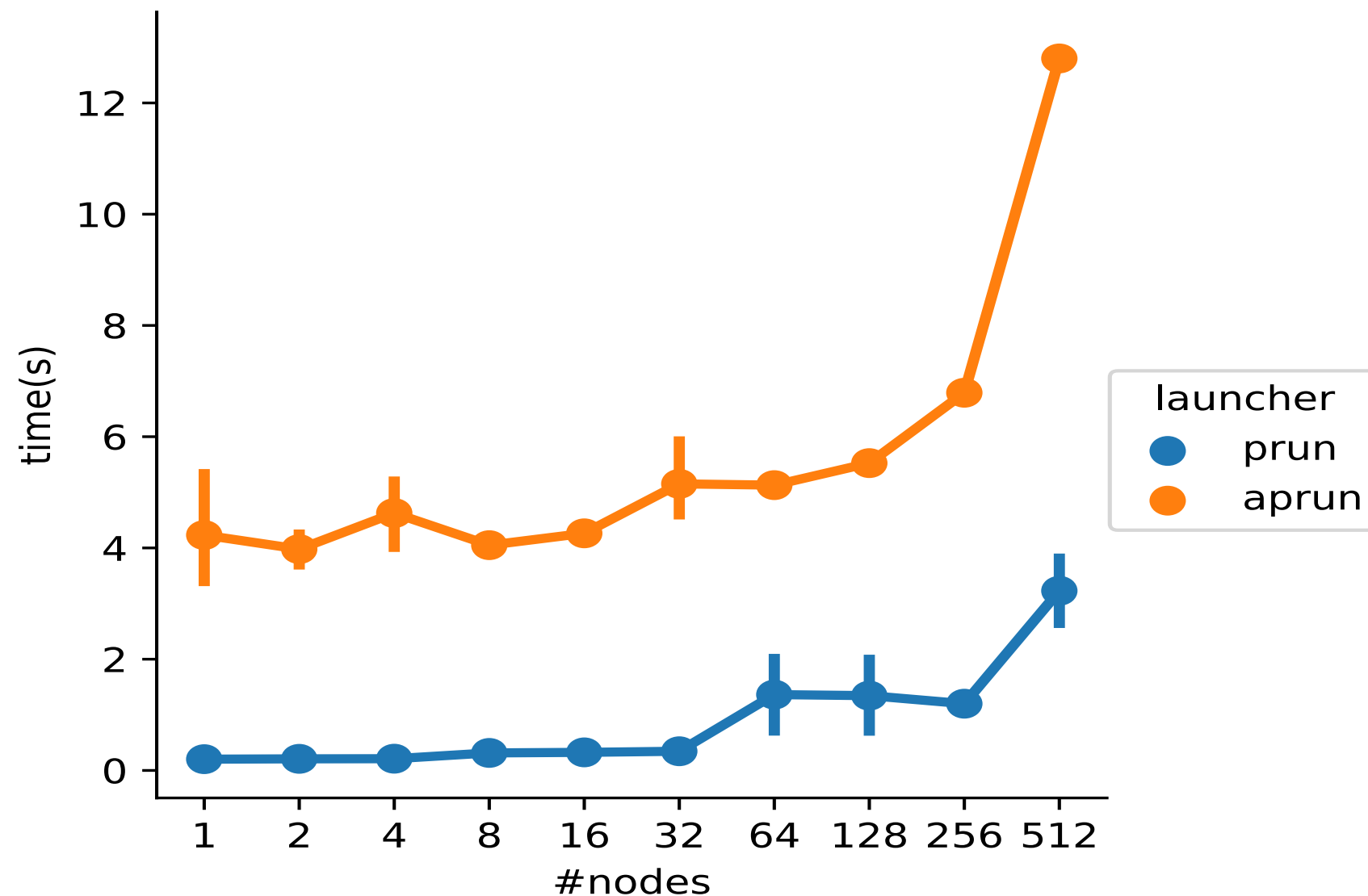
Scalable Performance

Aurelien Bouteiller

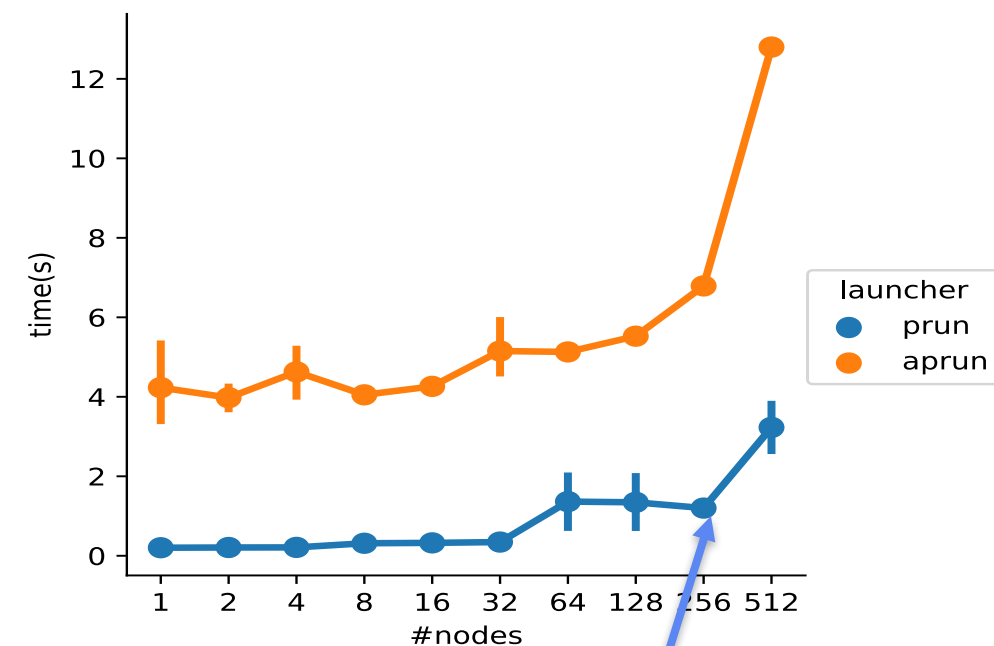
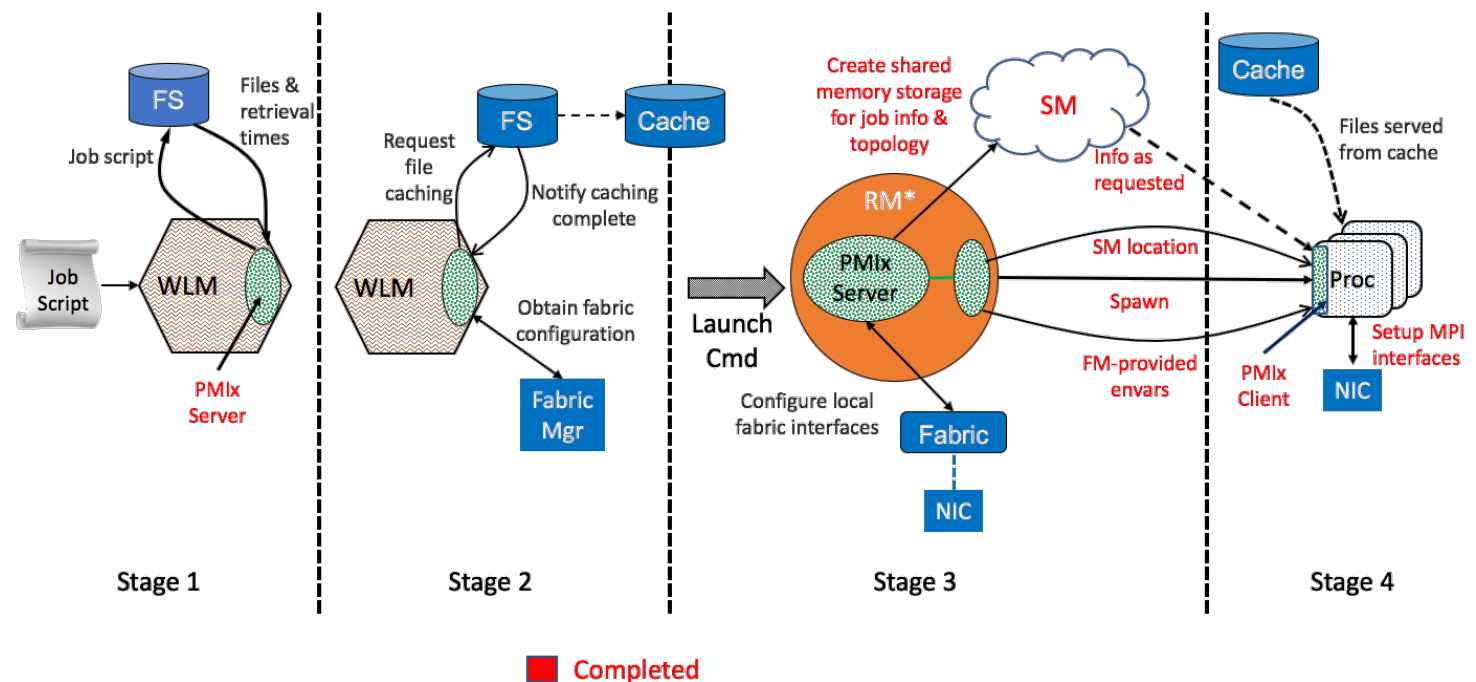


PMix Launch Early Results

- Comparing launch sequence of Open MP with pmix vs aprun
- ORNL Titan (Cray XK)
- 16 procs/node (8k ranks max)
- 4x speedup vs aprun



PMix Launch Early Results



- MPI processes startup sequence:
 - Modex Exchange (business card and NIC info)
 - Fence

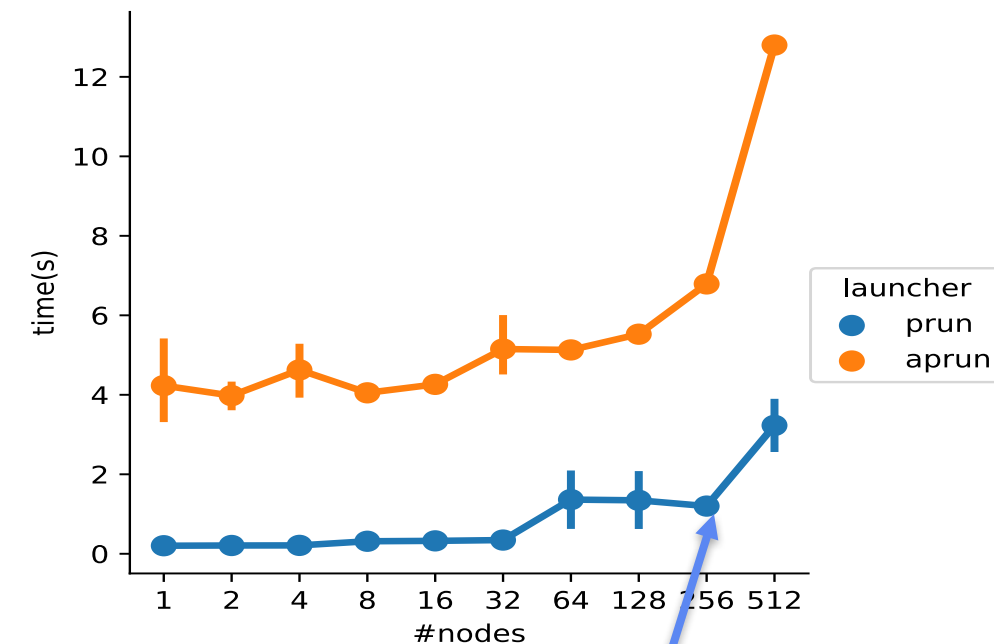
- PMix MPI processes startup sequence
 - No Modex (PMix provides business cards info from local RM)
 - Fence unnecessary

- In this experiment,
 - MPI_Init fence is still present
 - uGNI driver startup adds a significant latency (not present with e.g. EDR IB)

```
prun --npernode 16 -mca pmix_base_async_modex 1 -mca pmix_base_collect_data 0 -n 512*16
./mpi_no_op
```

PMIx Launch Early Results

- Try this at home! (and report back)
 - Run Open MPI
 - contrib/scaling/scaling.pl**
 - Will produce startup scalability data for
 - Native launcher (e.g. Slurm, PBS, Alps)
 - Open MPI `mpirun`
 - With and without MPI_Init/Finalize fence
 - PMIx `prun` with persistent daemons
 - With and without MPI_Init/Finalize fence



- In this experiment,
 - MPI_Init fence is still present
 - uGNI driver startup adds a significant latency (not present with e.g. EDR IB)

Side topic: Scalable Communication

- **External projects: not part of the PMIx specification**
- **Mellanox: Hardware Accelerated Overlay**
- **Intel/UTK: SCON: Scalable Communication Overlay Network**

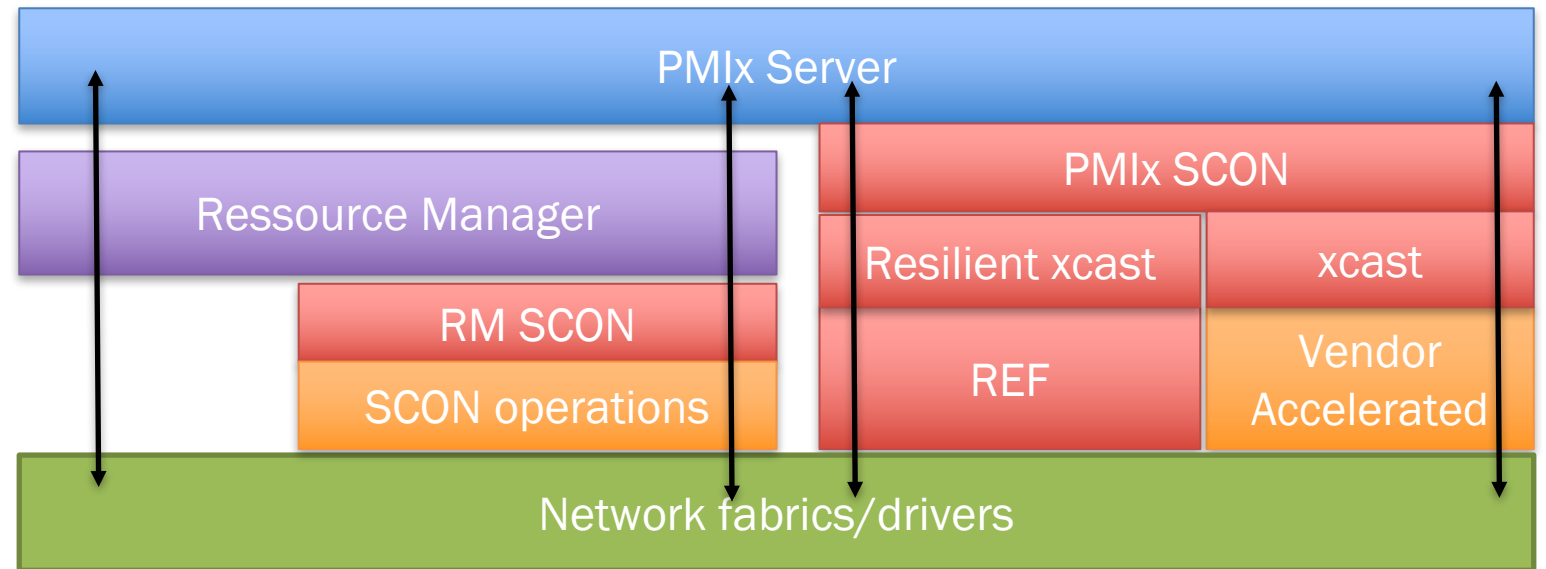
- **Goals:** scalable, generic, fault tolerant communication infrastructure for RTE systems
- New partners welcome 😊

• SCON supply for communication features not provided from a supplier

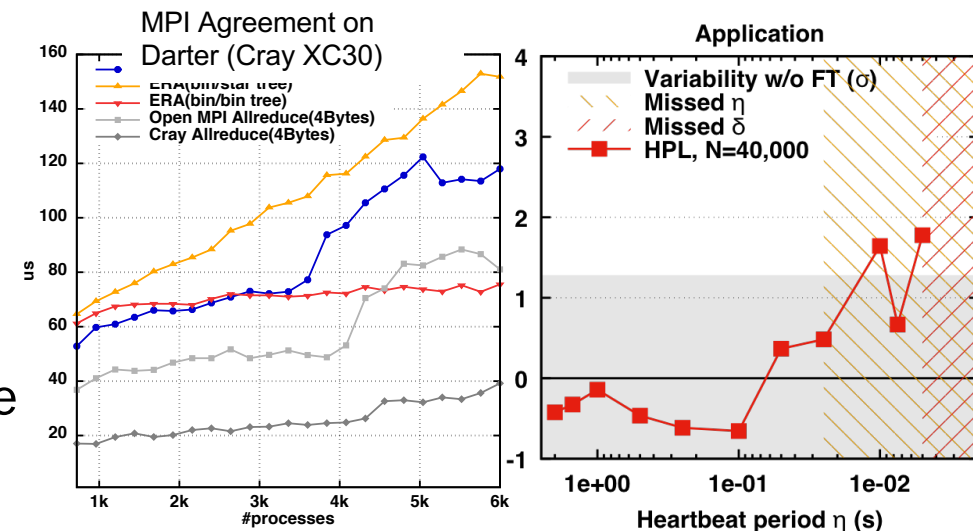
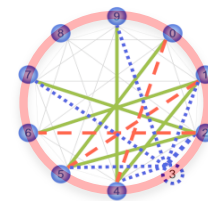
- PMIx uses RM communication features, if available
- Similarly, RM uses internal communication features, if available

• SCON will use vendor accelerated modules, when available

- Fallback to reference SCON operations when no vendor specific



- Failure Detector (core function)
- Reliable Broadcast (notifications)
- Agreement (allgather/modex/fence)



Algorithms proven at the MPI level

PMix Standard Document

Josh Hursey



IBM **Spectrum MPI**

PMix Standard

- Goal:
 - Create a document that details the PMix project goals, architecture, interfaces and semantics, and language bindings.
 - Separate from the PMix Reference Implementation.
 - Ratify that document with the PMix community.
 - Open process to encourage broad participation.
- Start with the interface from the PMix Reference Implementation v2.0.

PMix Standard



Process Management Interface for Exascale (PMIx) Standard

Version 2.0 (draft)

November 2017

Lots of work to do!

Requesting:

- Assistance in crafting standard language,
- Vigorous debate on semantics, and
- Participation in the ratification process.

This document describes the Process Management Interface for Exascale (PMIx) Standard, version 2.0 (draft).

<https://github.com/pmix/pmix-standard>

PMIx Standard - Organization

1. Introduction to PMIx

1. Overview, Goals, Architecture

2. PMIx Terms and Conventions

3. Data Structures, Types, Constants

1. Includes: Reserved attributes, Keys

4. Initialization & Finalization

1. Client, Server, Tool interfaces

5. Key/Value Management

1. put, get, commit, fence
2. (un)publish, lookup

6. Process Management

1. spawn, (dis)connect, resolve_peers

7. Job Allocation Management

1. Allocation request, process monitor

8. Event Handling

1. (de)register_event, notify_event

9. Data Packing & Unpacking

1. (un)pack, copy

10. PMIx Server Specific Interfaces

1. setup_fork, (de)register_nspace
2. pmix_server_module_t

A. Revisions, Acknowledgements

B. Bibliography

C. Index

<https://github.com/pmix/pmix-standard>

PMix Standard

- Participation:
 - File PRs against the GitHub repo
 - <https://github.com/pmix/pmix-standard>
 - Start a discussion on the PMix community mailing list
 - <https://groups.google.com/forum/#!forum/pmix>
 - Join the weekly PMix teleconference
 - Thursday's 3 pm (Eastern)
 - The approval process is still being determined, but will likely match that of other HPC standardization bodies (e.g., MPI, OpenMP)

<https://github.com/pmix/pmix-standard>

PMix Standard

- Questions for the community
 - Is it important to have a v1.0 version of the standard to match the interface from the PMix Reference implementation v1.x?
 - The standard focuses on the C interface, how important are language bindings? Which additional languages should we focus on?
 - What examples would you like to see in the standard to help illustrate how PMix should be used?
 - How should the standardization ratification process be structured? What determines a voting member of the community?

PMIx Standard

- New interfaces & Semantic changes for discussion
 - A PMIx server can return **PMIX_ERR_NOT_SUPPORTED** if a given attribute to, for example, `PMIx_Get()` is not supported.
We do not have an interface for the PMIx client to query this information so they know in advance what attribute is supported.
 - Is this functionality that would be useful to PMIx clients/applications?
 - In what form should we provide that query mechanism?
 - a) `PMIx_Query_supported_attributes(...)`
 - b) Special PMIx attribute `PMIX_PROBE_SUPPORT` passed with the attribute?
 - c) ???

PMix Standard

- New interfaces & Semantic changes for discussion
 - Attributes can be required or not. What should be the default?
 - Currently we default to **not required**
 - Other suggestions?

Agenda

- State of the Community
 - Ralph H. Castain (Intel)
- Scaled Performance
 - Aurelien Bouteiller (UTK)
- PMIx Standards Document
 - Josh Hursey (IBM)
- Q&A

*Reminder:
Enhanced launch scaling talk
SLURM booth @ 2pm*