Process Management Interface – Exascale

PMI \times 10^{18}
PMIx Roadmap

16.05


2014

1.1.3 1.1.4 1.1.5 1.2.0

Bug fixes Bug fixes Shared memory datastore

SLURM Support
PMIx/UCX job-start use case

**Hardware:**
- 32 nodes
- 2 processors (Xeon E5-2680 v2)
- 20 cores per node
- 1 proc per core

**Software:**
- Open MPI v2.1 (modified to enable ability to avoid the barrier at the end of MPI_Init)
- PMIx v1.1.5
- UCX (f3f9ad7)

* **direct-fetch/async** assumes no synchronization barrier inside MPI_Init.
PMIx/UCX job-start usecase

key exchange type:
- collective
- direct-fetch
- direct-fetch/async

"allgatherv" on all submitted keys
Synchronization overhead

MPI_Init (sec) vs nodes

0.6
0.5
0.4
0.3
0.2
0.1
0

0 5 10 15 20 25 30 35

nodes
v1.2.0

• Extension of v1.1.5
  ▪ v1.1.5
    • Each proc stores own copy of data
  ▪ v1.2
    • Data stored in shared memory owned by PMIx server
    • Each proc has read-only access

• Benefits
  ▪ Minimizes memory footprint
  ▪ Faster launch times
Shared memory data storage (architecture)

- Server provides all the data through the shared memory
- Each process can fetch all the data with **0 server-side CPU cycles**!
- In the case of direct key fetching if a key is not found in the shared memory – a process will request it from the server using regular messaging mechanism.
Shared memory data storage (synthetic performance test)

Hardware:
- 32 nodes
- 2 processors (Intel Xeon E5-2680 v2)
- 20 cores per node
- 1 proc per core

• 10 keys per process
• 100-element arrays of integers

PMIx_Get(all), sec

- Open MPI v2.1
- PMIx v1.2

https://github.com/pmix/master/tree/master/contrib/perf_tools

https://github.com/pmix/master/tree/master/contrib/perf_tools

Messages

Shmem

processes
Shared memory data storage (synthetic performance test) [3]

**CL1 Hardware:**
- 15 nodes
- 2 processors (Intel Xeon X5570)
- 8 cores per node
- 1 proc per core

**CL2 Hardware:**
- 64 nodes
- 2 processors (Intel Xeon E5-2697 v3)
- 28 cores per node
- 1 proc per core

---

**CL1**: avg(Put)

- PMI2: 0.20 ms
- PMIx msg: 0.07 ms
- PMIx shmem: 0.07 ms

**CL1**: avg(Get)

- PMI2: 0.22 ms
- PMIx msg: 0.08 ms
- PMIx shmem: 0.07 ms

**CL2**: avg(Put)

- PMI2: 0.37 ms
- PMIx msg: 0.06 ms
- PMIx shmem: 0.06 ms

**CL2**: avg(Get)

- PMI2: 0.32 ms
- PMIx msg: 0.11 ms
- PMIx shmem: 0.06 ms
PMIx Roadmap

2014

1/2016
1.1.3
Bug fixes

6/2016
1.1.4

8/2016
1.1.5
Bug fixes

11/2016
1.2.0
Shared memory datastore

12/2016
2.0.0?
Tool support

Potential intermediate release
Tool Support

- Tool connection support
  - Allow tools to connect to local PMIx server
  - Specify system vs application
Tool Support

- **Query**
  - Network topology
    - Array of proc network-relative locations
    - Overall topology (e.g., “dragonfly”)
  - Running jobs
    - Currently executing job namespaces
    - Array of proc location, status, PID
  - Resources
    - Available system resources
    - Array of proc location, resource utilization (ala “top”)
  - Queue status
    - Current scheduler queue backlog

*Examples*

Debuggers?
PMIx Roadmap

- 1/2016: 1.1.3
- 6/2016: 1.1.4 Bug fixes
- 8/2016: 1.1.5 Bug fixes
- 11/2016: 1.2.0 Shared memory datastore
- 12/2016: 2.0.0? Tool support
- 1/2017: 3.0.0 Event notification
  - Log
  - Reference server
V3.0 Features

- **Event notification**
  - System generated, app generated
  - Resolves issues in original API, implementation
  - Register for broad range of events
    - Constrained by availability of backend support
V3.0 Features

- Event notification
- Log data
  - Store desired data in system data store(s)
    - Specify hot/warm/cold, local/remote, database and type of database, …
  - Log output to stdout/err
  - Supports binary and non-binary data
    - Heterogeneity taken care of for you
V3.0 Features

• Event notification
• Log data
• Reference server
  ▪ Initial version: DVM
    • Interconnected PMIx servers
  ▪ Future updates: ”fill” mode
    • Servers proxy clients to host RM
V3.x Features

Generalized Data Store (GDS)

- Abstracted view of data store
  - Multiple plugins for different implementations
    - Local (hot) storage
    - Distributed (warm) models
    - Database (cold) storage
- Explore alternative paradigms
  - Job info, wireup data
  - Publish/lookup
  - Log

Summer 2017
V3.x Features

Network Support Framework

• Interface to 3rd party libraries
• Enable support for network features
  ▪ Precondition of network security keys
  ▪ Retrieval of endpoint assignments, topology
• Data made available
  ▪ In initial job info returned at proc start
  ▪ Retrieved by Query

Spring 2017
V3.x Features

IO Support

• Reduce launch time
  ▪ Current practices
    • Reactive cache/forward
    • Static builds
  ▪ Proactive pre-positioning
    • Examine provided job/script
    • Return array of binaries and libraries required for execution

• Enhance execution
  ▪ Request async file positioning
    • Callback when ready
  ▪ Specify persistence options

Spring 2017
Open Discussion

We now have an interface library the RMs will support for application-directed requests

*Need to collaboratively define what we want to do with it*

Project: [https://pmix.github.io/master](https://pmix.github.io/master)
Code: [https://github.com/pmix](https://github.com/pmix)