Singularity

On-node Resource Manager for Containerized HPC Workloads

Geoffroy Vallee, Carlos Eduardo Arango Gutierrez, Cedric Clerget

Canopie HPC - Super Computing 2019

Introduction

- HPC workloads are becoming more complex, e.g., MPI+X
  - Multiple runtimes are now running side by side
  - Most runtimes assume by default that all available resources can be used
  - Containers are becoming a standard way to run scientific workloads
- HPC systems are becoming more complex
  - Current trend is for bigger, more powerful compute nodes
  - Multiple accelerators per node

Complex resource partitioning and sharing problems

The community started to discuss about internal resource manager for applications

Challenges

The venue of new systems, containers and workloads requiring multiple runtimes create new challenges

- ➔ It creates a deep hierarchy of execution contexts
  - ◆ processes & threads running on the host
  - ◆ containers and everything running inside the containers
- ➔ How can we precisely partition resources between?
- ➔ How can we share hardware resources?
- ➔ How can we assign hardware resources to process/threads that are potentially running deep in the hierarchy?

Goals

- Design a solution to be deployed on current and future systems
  - Integration with global scheduler
  - Support legacy scientific workloads, i.e., MPI
  - Support upcoming scientific workloads, e.g., MPI+X and beyond
- Scalable resource management that covers containers
- Hierarchical architecture
  - Scalable
  - Compatible with existing infrastructure
  - Compatible with existing programming languages (e.g., MPI) and runtimes
- Minimize modifications to existing implementations, specifications and standards

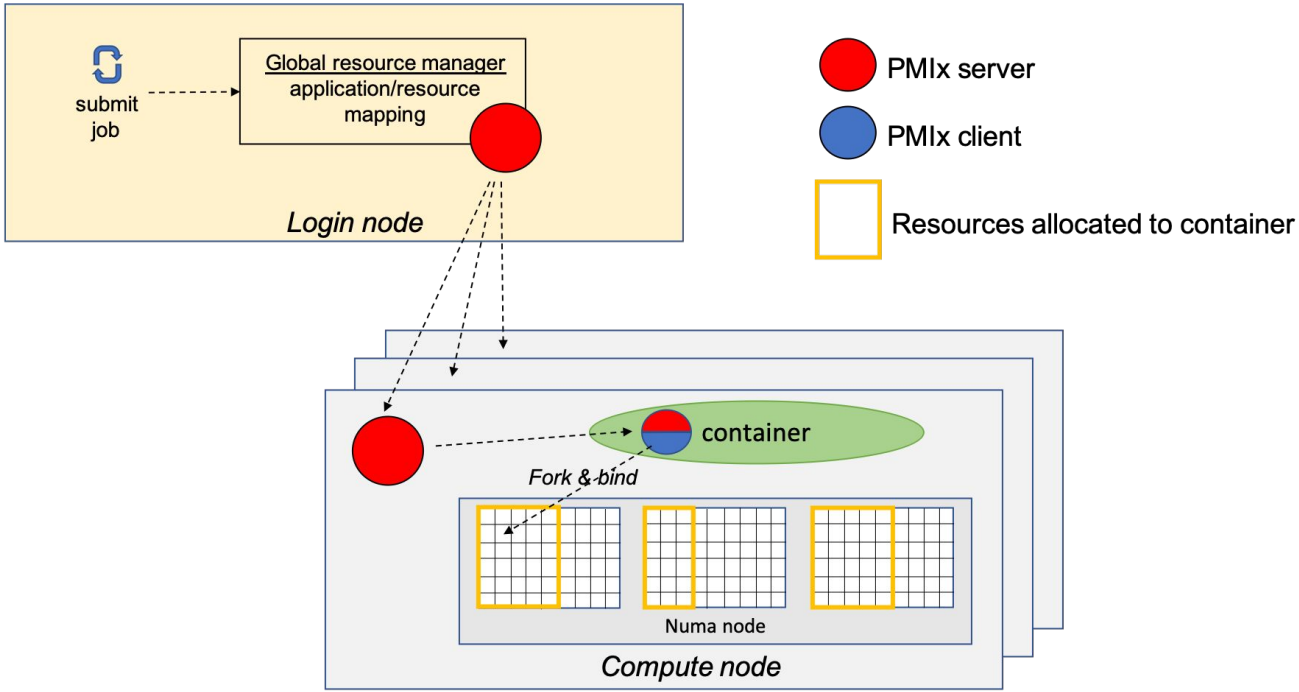
PMIx Overview

- Process Management Interface for eXascale
- PMIx is both a specification/standard and an implementation (OpenPMIx)
- Provides all the building blocks for the implementation of distributed asynchronous runtimes
  - Client/server model for scalability
  - Interface to resource managers
  - Event based architecture
- Now available for most of major MPI implementations
- Use by industry, government, academia

Singularity Overview

- Specially designed for HPC
- Container solution that runs strictly under the user's ID (no privileged access required)
- No heavy daemon required to run on the node (no system noise)
- Works well with parallel file system
- Let users
  - *Package* their application and data; bring and *execute* the package to virtually any system
  - Low to no-overhead while executing applications
  - Encrypt their containers
  - Sign and verify their containers

Architecture Overview



- ➔ Add a new PMIx thread to the Singularity runtime
- ➔ Interface Singularity with the global resource manager
- ➔ Ease the mapping of resources

Sylabs.io

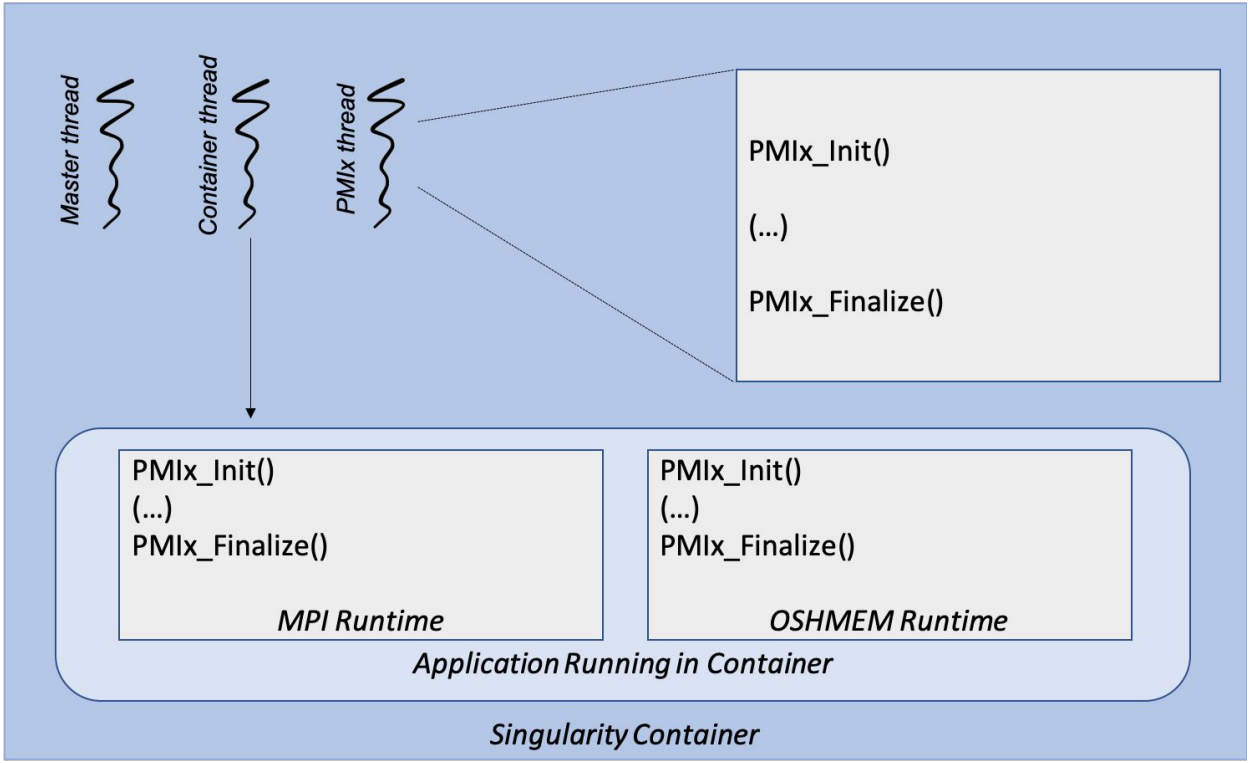
Extensions to PMIx

- From a standard point-of-view
  - PMIx already provides all the required interfaces and semantics
  - Points of interest
    - Semantic mapping between PMIx and Singularity namespaces
    - Tool interface
    - Allocation API: PMIX_ALLOC_NEW and PMIX_ALLOC_EXTEND can be leveraged to have a flexible way to add more resources to containers (when required resources are not known upfront)
- From an implementation point-of-view
  - Need to extend an implementation to support Singularity containers
    - It would be useful to have the PMIx runtime being capable of managing containers
    - OpenPMIx is a good candidate

Extensions to Singularity

Add a PMIx thread to the Singularity runtime

- ➔ Act as a PMIx client
  - What resources are being assigned to the container?
- ➔ Act as a PMIx server
  - How resources are assigned to what is running in the container



Extensions to Singularity (2)

- ➔ Ultimately creates a hierarchy of PMIx clients/servers
  - ◆ Easier to map, allocate and manage resources assigned and used by containers
  - ◆ Gives a better control over assigning processes and threads running inside the container onto resources (e.g., MPI binding)
- ➔ Leverage the PMIx tool interface for debugging and profiling
- ➔ Enable runtime coordination
  - ◆ Leverage PMIx event, communication and synchronization mechanisms

Conclusion

We propose an extension of the Singularity containers runtime and PMIx implementation to provide:

- ➔ A scalable resource manager for complex workloads that include containers
- ➔ A solution compatible with existing solutions and infrastructures
- ➔ New capabilities to assign, share and/or partition resources between different runtimes (e.g., MPI, OpenMP, Singularity)