Role of PMIx with Containers in HPC Environments

Joshua Hursey
Spectrum MPI Developer
jhursey@us.ibm.com

IBM

PMIx ASC 2Q 2021 Quarterly Meeting

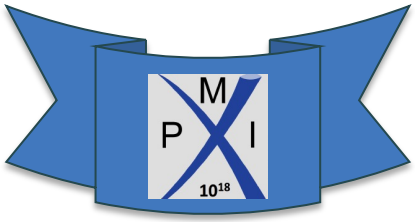
Overview

- **In this presentation:**
  - Explore the role of PMIx as a conduit cross the container boundary between the runtime environment, tools, and the PMIx client.
  - MPI used as an example PMIx client, but any PMIx client is impacted in a similar way.
- **Paper: "Design considerations for building and running containerized MPI Applications" @ CANOPIE-HPC Workshop 2020.**
  - What do you need to consider when building & running containerized MPI applications on HPC systems?
- **A few things to recall about containers:**
  - Processes running in a container instance can directly access the host OS and devices.
  - Containers cannot contain the kernel-space parts of libraries only the user-space parts.
  - Difference between a containerized process and a bare metal process:
    The file system mounts, namespaces, cgroups, capabilities, and security profiles.
    - The container runtime defines these aspects with the host OS.

https://www.canopie-hpc.org/program/
https://pmix.github.io/dynamicswg

Launching (containerized) MPI Processes

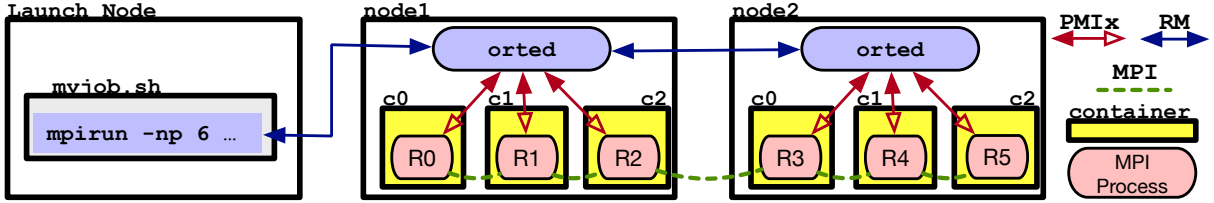
- **The Resource Manager (RM) assists the MPI library in a number of critical tasks:**
  - Launching MPI processes,
  - Enabling processes to discover information about the allocation,
  - Facilitating communication establishment between processes,
  - Event notifications, and more.



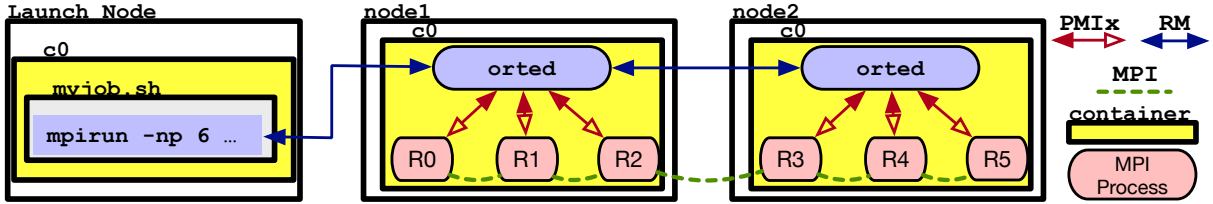
- **Direct Launch:** Launching (containerized) processes with the system provided launcher
  - **Examples**: Slurm's srun, ALPS' aprun, JSM's jsrun
  - Primary advantage: Most tightly integrated/supported on the system. Faster launch times.
  - Primary disadvantage: Might force other design decisions depending on capabilities.
- **Indirect Launch**: Launching (containerized) processes with the MPI provided generic launcher
  - **Examples**: mpirun, mpiexec
  - Primary advantage: Launcher can be included in your container image. Use the mapping, binding, and ordering options you know (and love).
  - Primary disadvantage: Requires a two-step launch (daemons then application)

One container instance per-process? or per-node?

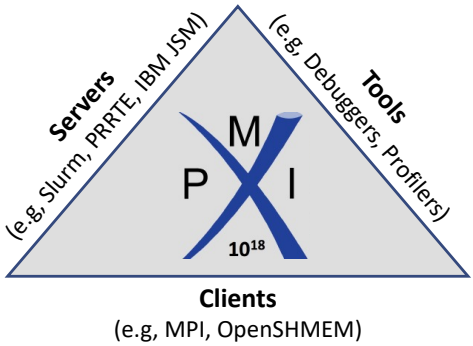
- **One container per-process (1-per-proc)**: Container as a static application binary
  - Use the resource manager to launch 1 container per process on all nodes
    - Uses the RM infrastructure on the machine including optimizations for managing images
  - Processes are contained from one another: additional container boundary MPI considerations



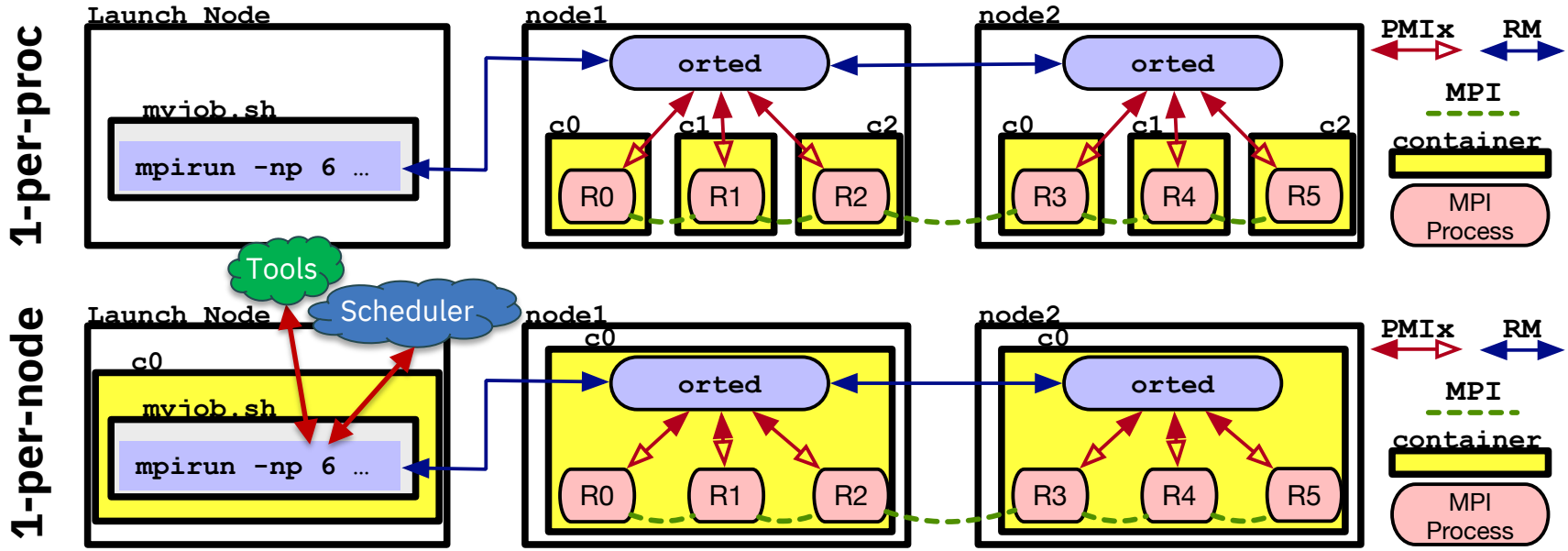
- **One container per-node (1-per-node):** Container as a static execution environment for apps
  - Resource manager launches 1 container per node, under which all processes are launched
  - No boundaries between MPI processes, and only 1 container instance per node



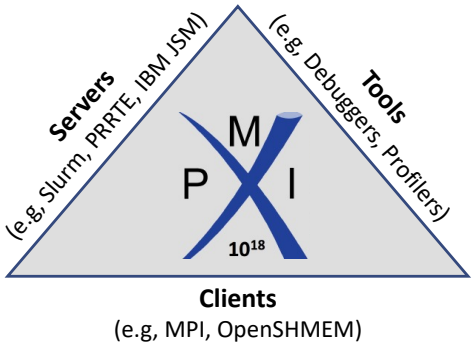
PMIx to talk across container boundaries



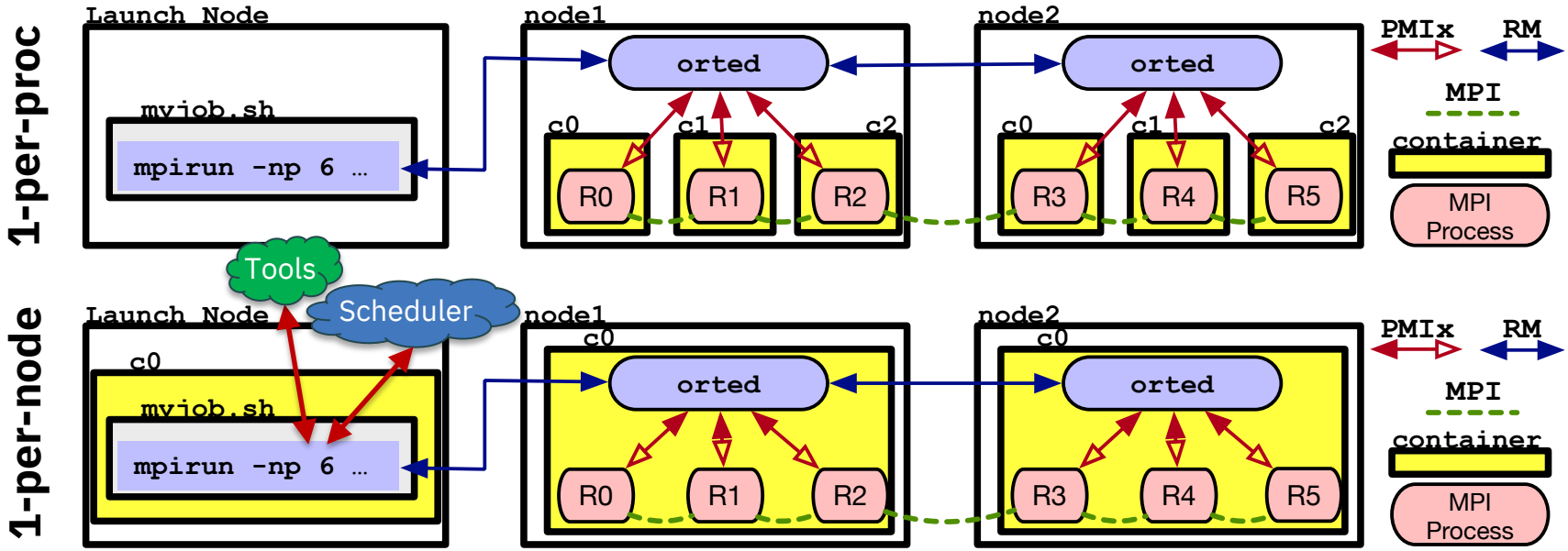
- **PMIx:** A community defined, standard API for programming libraries and tools to interact with the RM in as abstract of a manner as possible.
  - Connects RM Servers (like Slurm, JSM) with Tools (like debuggers) and Clients (like MPI libraries) across container boundaries.



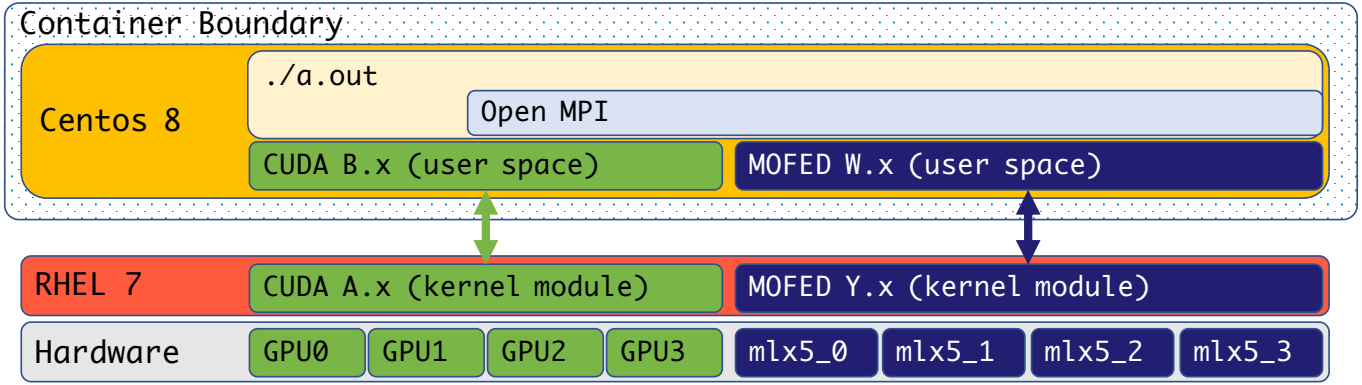
PMIx to talk across container boundaries



- **PMIx Implementation outside** the container (in the Resource Manager, Scheduler, Tool, ...) must be able to talk to the **PMIx Implementation inside** the container (in the MPI library, launcher program, ...).
  - This is a feature of a PMIx implementation, not of the PMIx Standard.



Aging in Place (cross-version considerations)

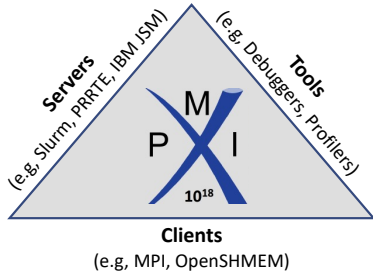


The cross-version issue with containers is not exclusive to PMIx

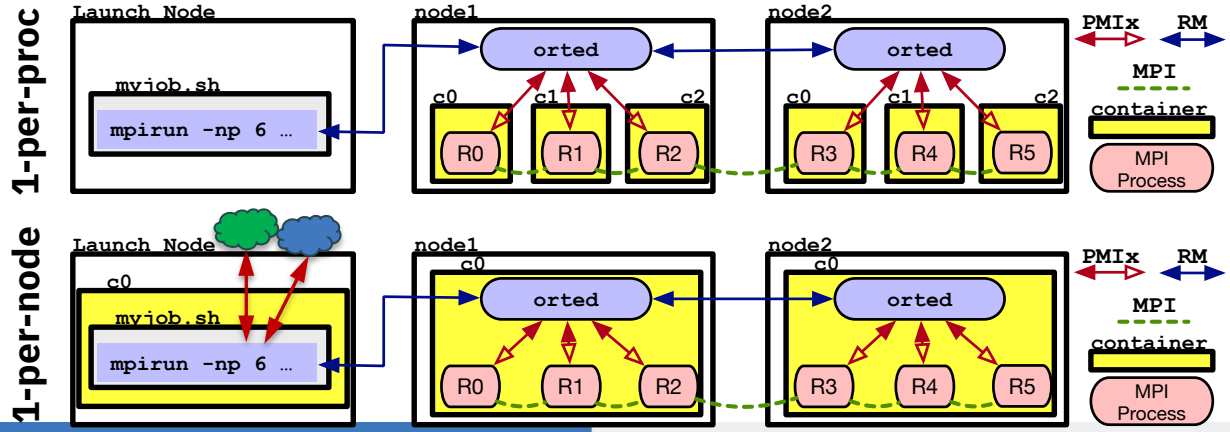
- **As the container image and the HPC system software levels evolve they will drift apart!**
- **Container "matches" the system (B == A)**
  - No problems as the software levels match
- **Container is "newer than" the system (B > A)**
  - A newer application uses an established HPC system
- **Container is "older than" the system (B < A)**
  - A HPC system software update causes this with established containers
- **A sysadmin approved "Base Image" could help "older than" more easily become "matching"**

IBM

Conclusion



- **What is the role of PMIx in a containerized HPC environment**
  - **A community defined, standard API connecting** programming library clients, tools, and resource managers in as abstract of a manner as possible.
  - Recognize that these **three roles may exist across a container boundary** so **cross-version issues are a problem**. The problem is not exclusive to PMIx.
  - PMIx implementations should have **a plan for negotiating version differences** across the container boundaries.



Thank you.