Networking and Information Technology
Research and Development

The government seeks individual input; attendees/participants may provide individual advice only.

Middleware and Grid Interagency Coordination (MAGIC) Meeting Minutes
May 2, 2018, 12-2 pm
NCO, 490 L'Enfant Plaza, Ste. 8001
Washington, D.C. 20024

Participants (*In-Person Participants)

Ashley Barker	ORNL	Paul Love	NCO
Brian Bockelman	UNL	Brian Lyles	ORNL
Ben Brown	DOE/SC	David Martin	ANL
Shane Canon	LBL	David Montoya	LANL
Richard Carlson	DOE/SC	Lavayna Ramakrishnan	LBL
Adam Carlyle	ORNL	Rajiv Ramnath	NSF
James Clark	LIGO	Tim Randles	LANL
Michael Crusoe	CWL/Octiog	Don Riley	UMD
Kaushik De	UTA	Arjun Shankar	ORNL
Dave Godlove	Sylabs	Alan Sill	TTU
Robin Goldstone	LLNL	Derek Simmel	PSC
Michael Jennings	LANL	Jack Wells	ORNL
Padma Krishnaswamy	FCC	Frank Wurthwein	UCSD/OSG
Eric Lancon	BNL	Wei Yang	SLAC/Stanford
Joyce Lee	NCO	Andrew Younge	Sandia
Miron Livny	UW-Madison		

Action Items: MAGIC April minutes will be approved and posted to the website in 1 week pending submission of any comments.

Proceedings
This meeting was chaired by Richard Carlson (DOE/SC) and Rajiv Ramnath (NSF).

Containerization and Virtualization Technologies Series: Impact on Science Discussion

Fundamental questions and differences (vis-a vis the commercial industry):
How do we deal with a rapidly changing landscape?
In containerization/virtualization, we are driven by the cloud industry. How do we react quickly to huge investments occurring outside of the science community?

- While there has been adoption, forward movement is occurring outside of academia and being brought in.
    - Pro: We don't have to develop Docker and Singularity.
    - Con: Does it change how we operate? We cannot continue acting according to our academic clocks or we will fall behind.

Investment of federal resources:
Okay for a project that enhances an open source repository instead of developing new technology?

- Depends on if gaps or mismatches exist between what is delivered in industry and what the scientific community needs. Middleware's role: to identify the correct abstraction.

Commercial cloud development:

- Fundamental difference between the large-scale science communities vs. cloud in the historical provision of access to shared structures (e.g., the security concern behind root restriction is absent when working with the cloud.) Thus concerns (who can build container; how ensure security, etc.) are left to the user. The community could validate certain images and help parse out use cases.

How can we ensure that funds, etc. won't be used for mal-intentioned purposes?

- In the commercial industry, the customer assumes responsibility for his/her actions.
    - Function of resource provider vs. labs (e.g., different terms of service between labs vs. Google)? These questions assume a middleware layer that is absent in Google. Assumes resource providers want more control over their resources instead of giving away the keys. Why?
    - Instead of regulating and forcing people through certain layers, perhaps adopt an approach that gives technical latitude in achieving what is allowable (e.g., a middle level agreement that ensures resources used comply with legal and ethical restrictions).

System Development and Standardization Activities Discussion

Standardization Activities

- Good start by the Cloud Native Computing Foundation (www.cncf.io) in developing standards for containers and container networking.
- Linux foundation projects: CNCF, Open Container initiative (OCI) is a source of standardization.
- Exascale Computing Project (ECP): goal of developing best practices.

Level of standardization?

- Emergence of standardization that is closer to researcher/data analyst, so IT providers can innovate more under the hood (workflow, description standards, portability, etc.).
    - Historical context: In the early days of distributed computing, the Open Grid Forum (OGF) and its predecessors had that role and only scientists ran these projects.
    - Post-technology explosion: Science community's response has varied, from basing work on commercial and Open Source non-science based developments vs. staying close to original concepts that led to the grids. Also, lost a common forum for bringing these projects together. OGF etc., should encourage technology adoption.

How can we maintain a forum for the exchange of information focused on users' needs? How can we approach the efficient creation of large scale infrastructures?

- Perhaps adopt new methods being used.
- Also exchange information through the MAGIC team.

Building and Deploying Containers Discussion

Who builds containers?

- No restriction, but would be more productive if domains started groups to build images relevant to their sub-communities; funding agencies can facilitate/fund with some steering (best practices).

- Best practices for the scientific community: Develop to ensure container is versioned correctly and integrated into publishing and overall research. ECP working group hopes for a consensus on best practices, perhaps across science domains
- Version problem with containers? Yes, but it is a placeholder for versioning problems with software and supporting kernels. There is technology allowing users to find their executables when we built grids and could likely have similar cryptographic method of snapping a container. Already supported in most cases.
- Either root or user builds container as root on a different system. Allow folks to run containers that have been built by someone who has root (far from perfect). Elegant solutions: Singularity hub could point to GitHub repository service; build service could be part of solution.
- Fundamental change in how scientists are building applications. Different from HPC world: user has intimate relationship with every line of code submitted vs. building containers by downloading code from global sources (supply chain issue).
- Large scale science infrastructure code projects: should provide vetted containers for their produced software (NOAA code). Subcommunities build around particular codes and have some responsibility. Similar to the official repositories for Docker Hub. Not the only solution, but if it becomes available, more folks would use.
    - Technology already in existence and used (e.g., container run times). Important to link applications being developed to those running it. Recommend integrating between container and run time strategies instead of adapting our own solutions.
    - Some communities will have work to do because their software development process is built on how traditional supercomputers and HPC centers operated.
- How can we build containers that are optimized for different systems and different architectures?
    - Fundamental challenges, need tools to study container performance issues
    - Commercial solutions are available – use or develop our own?
    - Develop best practices.
    - ECP working group**:** may split up into execution paths that look more like traditional HPC workloads vs. service oriented pieces (cloud environment). Commercial space is geared towards developing tools towards the latter.

Challenges for the Individual Scientist: If the only option is commercial solutions, then the scientist's responsibility as a system administrator becomes more expensive.

- Difficult to get science done in the cloud at any scale.
- Big change from the past: Almost all the questions posed here require someone to facilitate data movement and make things work at scale (new middlemen/women to provide a platform allowing abstractions, as in the past).
- Need for expanded skillset: more software engineering skills in code team. Need to adapt and fund a slightly different technological skillset. It is one of many code optimization, performance and deployment issues encountered in deploying these infrastructures.
- The researcher is more concerned about whether it works vs. speed of solution.

Toolkits: How can we continue to develop and deploy common, simple toolkits that can be adopted in HPC centers all the way up to exascale?

- Put build tools in place, maybe build services.
- Improve the model for building containers.

- Need to fund development of security scanning and performance measurement tools.
- Default: Major supercomputers must do internally.
- Assuming no new funding, what should we stop doing? Or is it possible to provide tools that make container deployments more efficient than current tasks (e.g., If NOAA shipped containerized version of their applications, would have productivity gains)? If we have community build systems, we could share across centers and leverage them.
    - Leverage most of existing Docker ecosystem and OCI standards because they solve 90% of problems (e.g., signing layers, signing images, transport, ensuring users can build).
    - Needed focus area: problems unique to HPC. Users can build inside of user named spaces and build software and generate containers (solved or under development at LANL, NERSC and other centers)
- Find a balance between developing our own solutions and funding developments vs. using commercial solutions. How do we ensure we make adequate use of commercial solutions when we address challenges? Commercial solutions are available. Open Source and commercial tools create a rich environment, so it isn't necessarily one or the other. Would be useful to discuss tools and approaches

Best practices: Within the larger ECP project, a good development has been more focus on software engineering than in scientific application oriented efforts. Need funding and standards creation.

- Many existing projects already supply a Docker file in their GitHub repository. Supports allowing the container to be used in a safe, secure manner (Shifter and Charliecloud).
- Need to acknowledge dependency on Docker and Docker model.
- Red Hat had issues with changes in Docker upstream - mission to replace Docker with Docker equivalence that does not require Docker. Can do a lot with Buildah and Podman instead of Docker. As we are in the same ecosystem, we can leverage all the other projects instead of reinventing.

Implications for a long tail scientist? Ecosystem diversity: different container systems and different versions of containers. Most researchers will be lost.

- Regardless of the tools used, everyone is subscribing to OCI standards for containers. All are building containers in same format except for projects not participating in that community
- Standardization process: Funding, staff and participation implications. Needs:
    - Engagement in the container space.
    - Implement correct level of communication; supporters of these infrastructures should participate in the community.
    - Invest time and effort: support staff project participation to ensure needs are represented, ensure adequate feedback loop. To support this level of exchange, OGF will move all its community pages to Github, e.g., CERN has provided feedback into OpenStack.

Larger question: How do we enable reproducibility in computational science: Is there a fundamental difference from the commercial community regarding work replication in future? What does it mean in the container world to pull something out of the archive and replicate it?

- Industry would want a short timeframe to roll back. Need tools and procedures.
- Commercial space: Never heard of the need to recreate software to replicate results. Rather, legacy software needs to keep running in same environment.

- What about studies of challenges in running application or workflow; show that can keep reproducing or running over a long period.
- Ongoing DREAM challenge in workflow reproducibility. (See Resources section.)

Security Concerns: If a security incident is reported to Open Science Grid (OSG) or Oak Ridge Leadership Facility (OLCF), does the resource owner bear any responsibility?

- OSG: facilitates the relationship between the virtual organization (VO) and resource owner. Ensures that the 2 ends find each other.
    - For traceability, resource provider goes to the VO and specifies the desired actions for keeping track of what is running on his system. Grid base system: Trust VO to deliver to us an executable X.509 proxy that are related. But cannot guarantee that proxy owner submitted the job that OSG is running. More work needs to be done to understand.
    - Questions: What are legal requirements? What are we trying to accomplish? What vectors of attack do we have to protect? Software has vulnerabilities. What does co-existing on a machine mean? Software running for the containers has vulnerabilities
- OLCF: user programs ask a lot from our users (viewed as individual users, not as VOs). Starting to engage communities that would act like a VO.
    - Security profile: manageable with current efforts, but need to continue to examine and consider security risk as this type of use case becomes more common.

Further Questions:

What is the impact of containers on the way we do research computing? Do we understand what containers enable in terms of the end user? Is there something fundamental or more of the same? Allows smaller groups of researchers to build complex and portable application.

- Impact on user communities? Argonne National Laboratory (ANL): depends on user community (high energy, astrophysics communities, know how to package these). Others do not know. Not on the long tail; LCF I Leadership Class Facility s the head (take biggest projects with larger jobs)
- LBL Lawrence Berkeley National Laboratory users could come from long tail of physical sciences. Some have picked up containers since it is easy to get started.
- Resurgence of small community/single investigator community with large computational challenges when using light sources with high data generation rates (Advanced Light Source[ALS],etc). If containers are a way to support resurgence, communities need to get together through standardization process and ensure good use of machines.
- We are slowly responding to the reality of computing and data handling processing as part of every discipline, not just traditional hard sciences and engineering.

Do we understand the relationship between containers and accelerator hardware? Extent of dependency on the type of GPU, CUDA code? How much of software is on the hosting environment vs. on the container?

- Some tooling is being developed to increase affordability, but will take some time.
- What are we doing with Machine learners? Tensorflow successfully brought ML to the masses.
- Current state of art: Taking pieces of host environment that have been optimized for available hardware and injecting them into the container at runtime. Working with Nvidia and Mellanox to enhance that.

- Long term solution: abstraction layers that dynamically detect underlying hardware and functionalities. Do runtime optimization and architecture transparency to enable varieties of hardware to be utilized by more generic runtime code. Still developing this technology and hopefully we will have a better picture in 5 years. Funding opportunity to foster this development.
    - Hope it will work, but skeptical given success of having portability Message Passing Interface (MPI). Impact of all this on long tail of the hardware (where most people will run)?
    - Some ways around this but, there is a range of how much they can accommodate.
    - User perspective: GPU landscape is changing very fast; moving into very diverse hardware environment. Do we understand the relationship between diversity of the hosting environment and the container? Another topic, but not from the exascale perspective as it won't be accessible to most researchers. True in any environment. When run containers on Theta at ANL, run inefficiently because people do not take advantage of real hardware, only common hardware (e.g., may be running on containers running on one core).
        - Kubernetes: deals with it through plugins; infrastructure is still being developed. Fundamentally, MPI and communications with GPUs, etc., occur at a kernel level and should expose the proper Application Binary Interfaces (ABI) inside the container ABIs for the kernel calls to work within the container. Cannot arbitrarily put a container on a piece of hardware and expect it to work (containers work by isolating name spaces [and assigning a memory]); user education is needed. That is a computer science problem that needs to be solved. In practice, Nvidia pushed out their framework early.
        - Device plugins are probably the right technological solution; i.e., container optimized operating systems that can give the user clear warning when they're trying to call something that a normal open source can support. A container optimized operating system may be able to communicate that it is running in a container and provide instruction.
        - At minimum, in our deployments, we need clear flags when someone is trying to call a nonexistent hardware or hardware that they think they're talking to, but is being emulated somewhere.
    - ECP project website- posting confluence pages and structure for collecting information (early stages). Goal: generate public best practices.
    - DOE workshop: looked beyond CUDA and Nvidia cards, and address extreme heterogeneity as we enter and go beyond exascale. Discussed containers and virtualization tools. Report will be published.

Funding: Ways to incentivize co-teams?

- Similar to data release policies: perhaps enact similar policies regarding how they make their applications available in container format
- Runtime format of containers is about
    - coupling the application with the environment in which it will run.
    - developers being able to specify and codify how that environment is built and what goes into it, changes game for debugging, portability and reliability. Then they know, from beginning, what version their application will be running under; eliminating much of the mystery is a huge incentive.

Value in security scans of containers?

- Yes, trying to figure out how to do it. Asked to scan containers and can assert a level of CVE compliance. But, at same time, the runtime developed is geared to ensuring that the content of container is largely irrelevant from a security perspective and can only impact the user running the container. (See Resources for link to tools).

Takeaways:

- Existing forums that community can join; no need for formal workshop
    - Piggyback on existing conferences –University Corporation for Atmospheric Research (UCAR) Software Engineering Assembly SEA symposium on containers
    - Workshops enable communications
- Members from our community should engage with standards bodies (OCI, CNCF)
- Return to federal funding issue.
- Deltas between science and commercial uses: identify largest deltas and what isn't being addressed? Need more surveys. (See Resources to link to Andrew Young paper).
  Important to enable scientists' success. We need to listen to researchers' experiences.

Resources:

- Resource links from Michael Crusoe (Common Workflow):
    - Very early draft of "Recommendations to package and containerize bioinformatics software", but it is applicable for other domains of science and research: https://github.com/ypriverol/containers-rules-manuscript/blob/master/manuscript.adoc
    - Workflow Standards Development organization: https://www.commonwl.org/
    - CERN is developing a platform for reusable research data analyses via Kubernetes https://reana.readthedocs.io/en/latest/
    - Many "science app stores" using containers are in development. Dockstore.org focuses on cancer genomics, but it is open source and easily re-usable for other fields that work with software containers: https://dockstore.org/
    - Workflow Reproducibility Challenge: https://www.synapse.org/WorkflowChallenge

- Cloud-native container-based security practices: https://thenewstack.io/cloud-native-security-patching-with-devops-best-practices/

- The Linux Foundation hosts many of the most important open source projects in the world, including Linux. With more than 1,000 companies backing tens of thousands of active developers, our projects harness the power of open source development to fuel innovation at unmatched speed and scale. (This includes many of the projects mentioned earlier in the call, including the Cloud Native Computing Foundation, the Open Container Initiative, etc.): https://www.linuxfoundation.org/projects/

- Usage cases for containers and/or current best practices (Andrew J. Younge, et al)
    - https://ieeexplore.ieee.org/document/8241093/

Events: October 29- November 1, IEEE eScience 2018, Amsterdam, the Netherlands
Next MAGIC meeting: June 6 (12 noon EDT)