Containers and HPC, What’s the big deal?

Olli-Pekka Lehto (@ople)
CSC – IT Center for Science Ltd.
HPC Knowledge Meeting
Apr 20\textsuperscript{th} 2016
Or rather..

Making It Easy to Do Custom HPC Environments

Olli-Pekka Lehto (@ople)
CSC – IT Center for Science Ltd.
HPC Knowledge Meeting
Apr 20th 2016
Containers?

- Isolated namespace “sandbox”
- Lightweight virtualization
  - Kernel shared with host system
  - Very fast initialization
  - Low memory overhead
  - Negligible performance overhead
- Not a new concept
  - Solaris Zones, IBM, LXC etc.
  - Popularized by Docker, LXD, Rocket...
What is Docker?

• Management framework for Linux Containers
• **Easy to use**
• “Stackable”
  – Can build new containers on top of old ones
  – Rapid prototyping and development
• Simple sharing of containers
  – http://hub.docker.com
• 1 process per container -philosophy
• Grown into a complete ecosystem
  – docker-swarm, docker-machine, docker-compose...
What is Docker?
Docker Adoption is Rapid

<table>
<thead>
<tr>
<th></th>
<th>June 2014</th>
<th>June 2015</th>
<th>Growth</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Contributors</strong></td>
<td>460</td>
<td>1,300</td>
<td>183%</td>
</tr>
<tr>
<td>Projects on GitHub</td>
<td>6,500</td>
<td>40,000</td>
<td>515%</td>
</tr>
<tr>
<td><strong>Docker Job Openings (Indeed)</strong></td>
<td>2,500</td>
<td>43,000</td>
<td>1720%</td>
</tr>
<tr>
<td>Dockerized Applications</td>
<td>14,500</td>
<td>150,000</td>
<td>934%</td>
</tr>
<tr>
<td><strong>Boot2Docker Downloads</strong></td>
<td>225,000</td>
<td>3,500,000</td>
<td>1,456%</td>
</tr>
<tr>
<td>Container Downloads</td>
<td>2,750,000</td>
<td>500,000,000</td>
<td>18,082%</td>
</tr>
</tbody>
</table>
Interest in HPC is growing


Container vs. cloud HPC

- Containers are simple to run under batch job queuing systems
- Clouds typically have their own management system (i.e. OpenStack) with no queuing
Current Choices for HPC Workloads

Cloud HPC
- Hosting
  - Web Servers
  - Windows
  - VM image app
  - Secure access
- Non-SLURM batch queue system
  - Linux distro X needed
  - “I need/want Root”
  - Preservation of SW stack
  - Complex stack

Bare-metal HPC
- Site-supplied apps
- User-supplied apps
Future Choices for HPC Workloads

Cloud HPC
- Hosting
- Web Servers
- Windows
- VM image app
- Secure access

Container HPC
- Non-SLURM batch queue system
- Linux distro X needed
- “I need/want Root”
- Containerized app
- Preservation of SW stack
- Complex stack

Bare-metal HPC
- Site-supplied apps
- User-supplied apps
Example Case

“My HPC application needs Distro X”

• Alternative 1: Adapt the application
  – Takes time and work
  – May not be possible with ISV codes

• Alternative 2: Run it in the cloud
  – You may get to play cluster admin!
  – Scheduling is limited in OpenStack (no backfill etc.)
  – Running a short job has large initialization overhead
  – Performance penalties

• Alternative 3: Run it in a container
  – No need to touch the application
  – Nearly as easy to run as a normal job
  – Very little overhead
  – Can use the normal batch job scheduler
Challenges for Docker HPC Use

• Security model is problematic
  – Initially designed for server environments
  – Containers launched as root
  – Access to bare metal, filesystems & device drivers

• Requires special kernel features
  – Relatively new Linux OS version needed

• Daemon must run on every compute node
Shifter (User Defined Images)

- Docker containers for HPC
  - “Cracks open” the container
    - Flattens contents into a chroot:able filesystem
    - No need for container support in OS
  - Integrates with batch job queue systems
    - No need to run a daemon on compute nodes
- Local repository: “Image gateway”
- Developed by NERSC for Cori (Cray XC40)
  - Cray released 1.0 support in March
  - Should also work for clusters

https://www.nersc.gov/research-and-development/user-defined-images/
https://bitbucket.org/berkeleylab/shifter
Some results

From NERSC (http://www.nersc.gov/research-and-development/user-defined-images/)
Singularity

• Containerize **only** the workload and it’s dependencies
  – Binaries, libraries, IO files
• Configuration stored in a `.sspec` file
  – Can be generated based on a run (libs, inputs etc.)
• A `.sapp` file is generated from the `.sspec`
  – A binary that can be run anywhere
• Supports MPI out of the box
• Developed by Lawrence Berkeley (1.0 just out)

http://singularity.lbl.gov/
Spec example

Name: services

%files
/bin/grep
/etc/services

%runscript
for i in $@; do
grep "$i" /etc/services
done

%test
if grep -q "smtp" /etc/services; then
    exit 0
else
    exit 1
fi

http://singularity.lbl.gov/docs_specfile.html
Quick Summary

• Docker
  – Rapid prototyping and development

• Shifter
  – Works directly with Docker containers
  – No need for container support

• Singularity
  – Packaging a single application or a workflow
  – No need for a big base image
  – Need to ensure all dependencies are there
Containers are not a magic bullet

• You can still build the app very badly
  – Maintaining applications/libraries up-to-date
  – Avoiding divergence (10s of base images etc.)
  – Testing each new release
  – Keeping the containerized stack optimized
  – Multiplatform deployments and sharing
    • scripts, sspecs, dockerfiles, playbooks oh my...

• Should app configuration be decoupled?
Enter EasyBuild

- Designed for managing HPC software stacks
- Single config for multiple targets
  - Bare metal, VMs, Singularity, Docker etc.
- Creation of well-built base containers
  - Libraries, compilers etc.
  - Give these to users, standardize between sites...
- Which way to do it?
  - Using EasyBuild in a container or
  - Building containers with EasyBuild?
Future challenges

• Encouraging users to do the right thing
  – Using containers only when needed
  – Promoting well-maintained containers
  – Avoiding insecure containers
• Deciding which container system(s) to support
  – The scene is constantly evolving
• Developing/sharing good base containers
• Looking at managing driver issues
  – Incompatible drivers w/ stack (OFED, CUDA etc.)
• Containers for system management / services
  – Microservice –based service nodes, containerized parallel filesystems?
Thank you!

• Questions?

@ople
https://github.com/CSC-IT-Center-for-Science