Solving Problems in HPC with Singularity

Michael Bauer - CernVM Workshop 2019
Michael Bauer
Senior Software Engineer
bauerm@sylabs.io
@bauerm97 on GitHub
Singularity
What is Singularity?

- Container runtime optimized for compute
- Unique security model
- Reproducible, cryptographically verifiable
- Massive adoption across many fields: HEP, bioinformatics, climate research, financial analysis, medical, etc…
Simplicity

Singularity is designed to be simple. From its build syntax, to its image format, the runtime, and compatibility with OCI and Docker, Singularity facilitates easy and intuitive onboarding.

SlF can squashfs, encapsulate and sign Docker and OCI image formats (encryption coming soon)
Singularity container images are immutable, cryptographically signed, and verifiable, ensuring absolute trust and bit for bit reproducibility of the container environment.

Note: Singularity uses no tarballs and thus no unsigned intermediate data.
Sylabs provides licensing, enterprise level support, professional services, cloud functionality, and value-added plugins for the Singularity container platform. Sylabs also helps to drive the development efforts of the Open Source Singularity Project.
From 4 to 35+ Full Time Employees
Singularity & CVMFS
CVMFS - The Problem

- Some sites unable to install CVMFS on host
- Need to run software distributed via CVMFS
- Have to distribute containers built from CVMFS software but which can’t access CVMFS
- Impossible to run fusermount inside container, so can’t bring CVMFS w/ container
CVMFS - Singularity

- Implement new functionality to enable --fusecmd plugin
- Mount a FUSE inside container without first mounting on host
- Transparent access in a container to CVMFS
- *It works (almost)! Not released yet, expected for 3.4.0*

```
$ singularity run --fusecmd /bin/cvmfs my_image.sif [...]

$ ls /cvmfs [...]
```
Sylabs Container Services
Sylabs maintains tested, supported, binary packages for our SingularityPRO offering

$ sudo yum install http://repo.sylabs.io/.../repo-rhel7.rpm
$ sudo yum install singularitypro31
Singularity Container Services Platform

- A series of value added container services:
  - Remote Build Service
  - Container Library
  - Key Management Service (KeyStore)

- Available now as a Cloud service
  - [https://cloud.sylabs.io](https://cloud.sylabs.io)

*Now available for OnPrem licensing*
Policy Management

- Enforce container signature policies - ECL
- Administrator curated containers or BYOC
- Container certificates via PKI, revoke certificate validity
Container Services

Container Lifecycle

- Build containers using build service on different architectures (x86, ARM, etc.)
- Scan containers for security vulnerabilities (aqua, clair, etc.)
- CI/CD pipelines for container lifecycle (build -> test -> run)
- Integrated with auth services (LDAP, etc.)
Singularity and Kubernetes
Running Singularity with K8s

- Singularity-CRI now in v1.0.0-alpha.3 release
  - Please try and provide feedback!
- Run all workloads (K8s microservices, HPC, EPC, etc…) with one unifying runtime
- Native support for GPUs, Infiniband/RDMA (coming soon), other HPC-oriented hardware via Container Device Plugin
Singularity, Kubernetes, SLURM
Multi-Cluster Scheduling

Kubernetes - scheduling to multiple HPC clusters - using Singularity containers to distribute the workloads.

All as unprivileged user.
Multi-Cluster Scheduling to SLURM via K8s

- Submit SLURM job request to Kubernetes server using new job kind “SlurmJob”
- Custom K8s Resource Daemon lives on each SLURM master node, lets K8s understand what resources exist
- Use default or custom K8s scheduling algorithm to dispatch jobs to different clusters
Singularity, SLURM, K8s

Using Kubernetes API to set extended resources and node labels.

Diagram:
- Kubernetes API
  - Node 1
    - RD*
    - Job1 container*
    - Job2 container*
    - Slurm controller*
    - Slurm master 1
  - Node 2
    - RD*
    - Job1 container*
    - Slurm controller*
    - Slurm master 2
  - Node 3
    - RD*
    - Job1 container*
    - Slurm controller*
    - Slurm master 3
apiVersion: slurm.sylabs.io/v1alpha1
kind: SlurmJob
metadata:
  name: cow
spec:
  batch:
    #!/bin/sh
    ##SBATCH --nodes=1 --cpus-per-task=1
    srun singularity pull library://sylabsed/examples/lolcow
    srun singularity run lolcow_latest.sif
    srun rm lolcow_latest.sif
  nodeSelector:
    containers: singularity
Demo
Why connect Singularity, SLURM, & K8s?

- Aggregate disparate compute resources
- Facilitate cross-pollination of HPC and Enterprise/AI workloads, tools, and resources
- World domination
1. Parallel training
2. Distribution of trained models
3. Real time AI / compute
4. Data streaming
5. Complete validation and trust
6. Supporting ecosystem
7. HPC as a Service

Singularity is the unifying substrate for all computing needs
Singularity Roadmap
Singularity Roadmap

- **v3.3.0 - Soonish**
  - Unprivileged fakeroot - Take advantage of new kernel multi user mapping functionality
  - Massive testing improvements (looking for help testing more complex workflows *e.g.* MPI)

- **v3.4.0 - Augustish**
  - Encrypted Containers

- **v3.5.0+ - Not so soon**
  - Partner/Vendor plugin ecosystem (*looking for collaborators*)
  - Rootless (non-SUID) support for SIF - Kernel work or SquashFS FUSE
Thank You!

Questions? Angry Comments?