Singularity

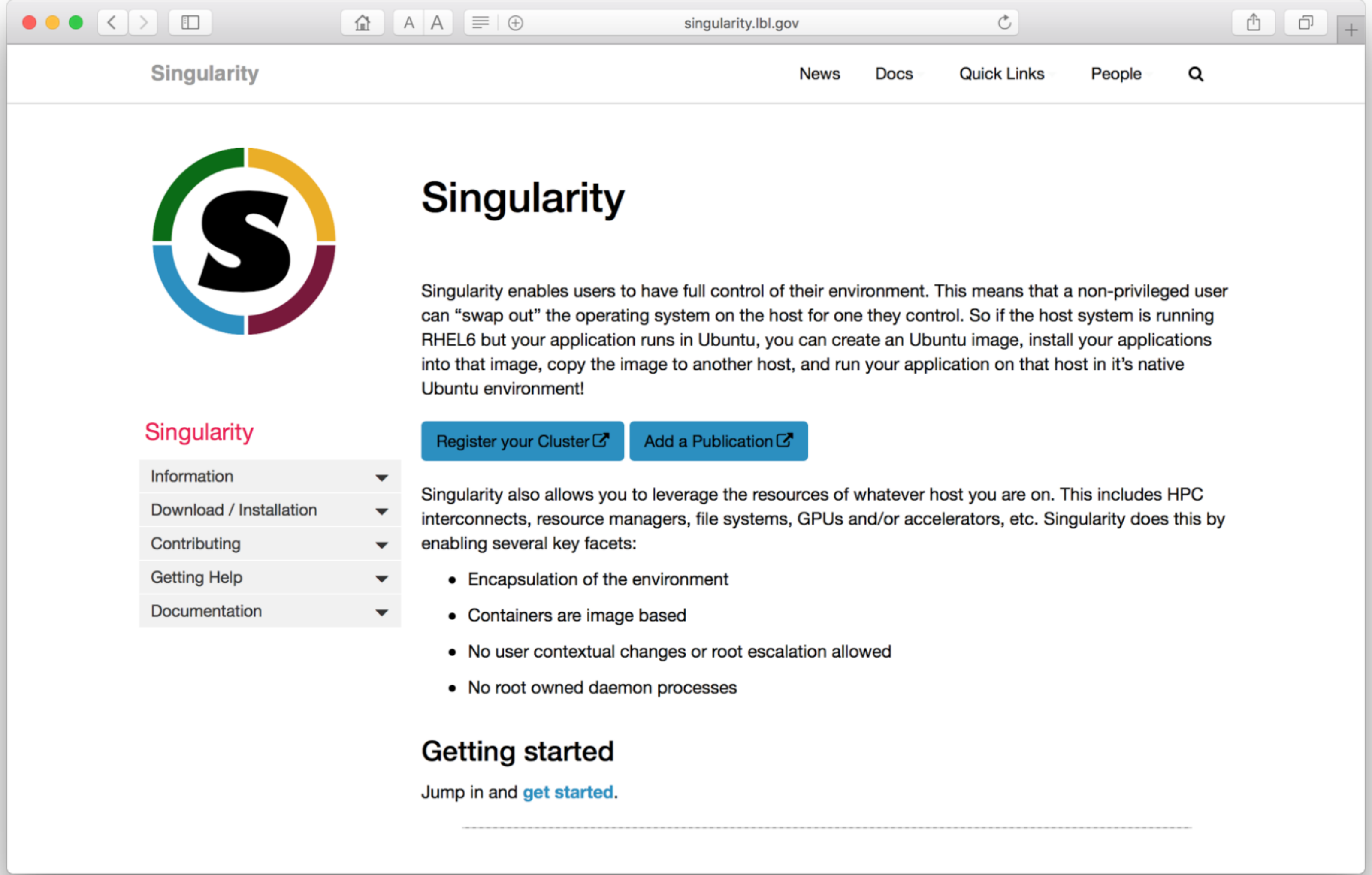
Michael Bauer

Contact Information

Michael Bauer

Staff Engineer, RStor
michael.bauer@rstor.io

@bauerm97 on GitHub
bauerm@umich.edu



Singularity

Singularity enables users to have full control of their environment. This means that a non-privileged user can "swap out" the operating system on the host for one they control. So if the host system is running RHEL6 but your application runs in Ubuntu, you can create an Ubuntu image, install your applications into that image, copy the image to another host, and run your application on that host in it's native Ubuntu environment!

Singularity also allows you to leverage the resources of whatever host you are on. This includes HPC interconnects, resource managers, file systems, GPUs and/or accelerators, etc. Singularity does this by enabling several key facets:

- Encapsulation of the environment
- Containers are image based
- No user contextual changes or root escalation allowed
- No root owned daemon processes

Getting started

Jump in and get started.

Containers for Scientific Computing

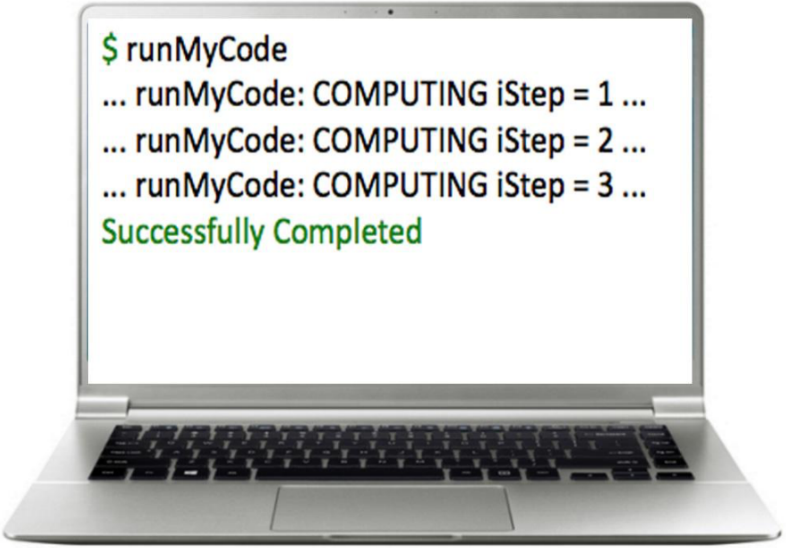
Why do we want containers in HPC?

Escape “dependency hell”

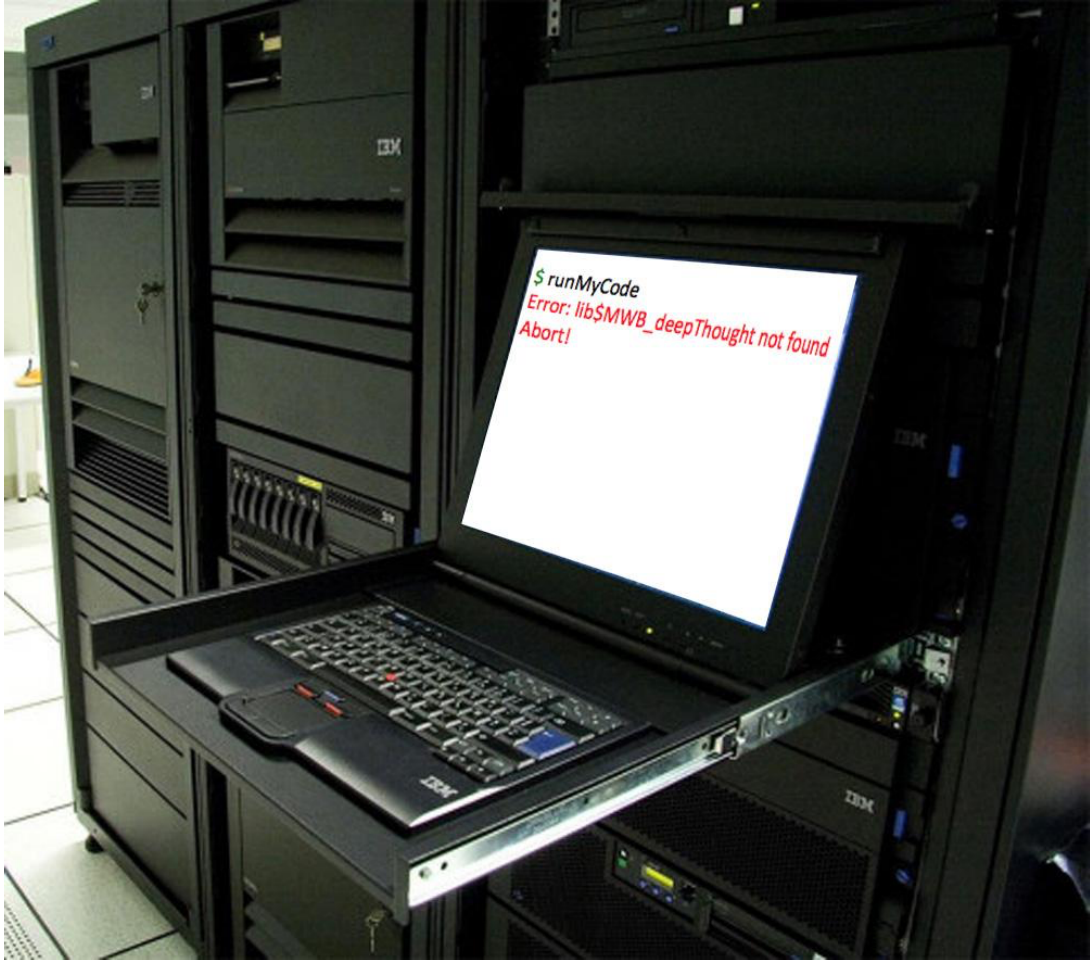
Local and remote code works identically every time

One file contains everything and can be moved anywhere

Environment Matters



```
$ runMyCode
... runMyCode: COMPUTING iStep = 1 ...
... runMyCode: COMPUTING iStep = 2 ...
... runMyCode: COMPUTING iStep = 3 ...
Successfully Completed
```



```
$ runMyCode
Error: lib$MWB_deepThought not found
Abort!
```

HPC container software can never touch root

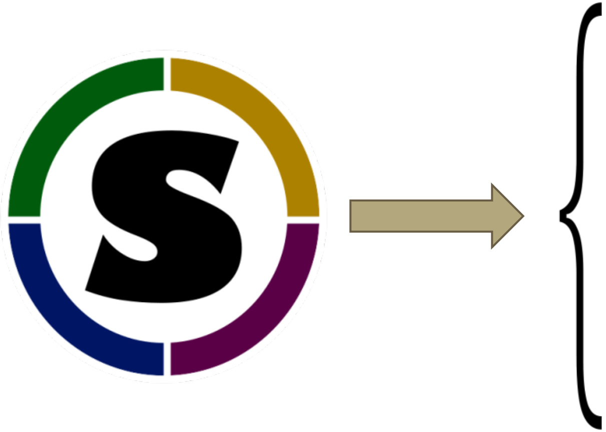
Singularity

Needs for HPC containers

- Any user can run containers without special privileges (root) ✓
- Integrate seamlessly into existing infrastructure ✓
- Portability between many systems ✓
- Users created and provided containers (no administrative oversight)

☒ This image cannot currently be displayed.

Singularity

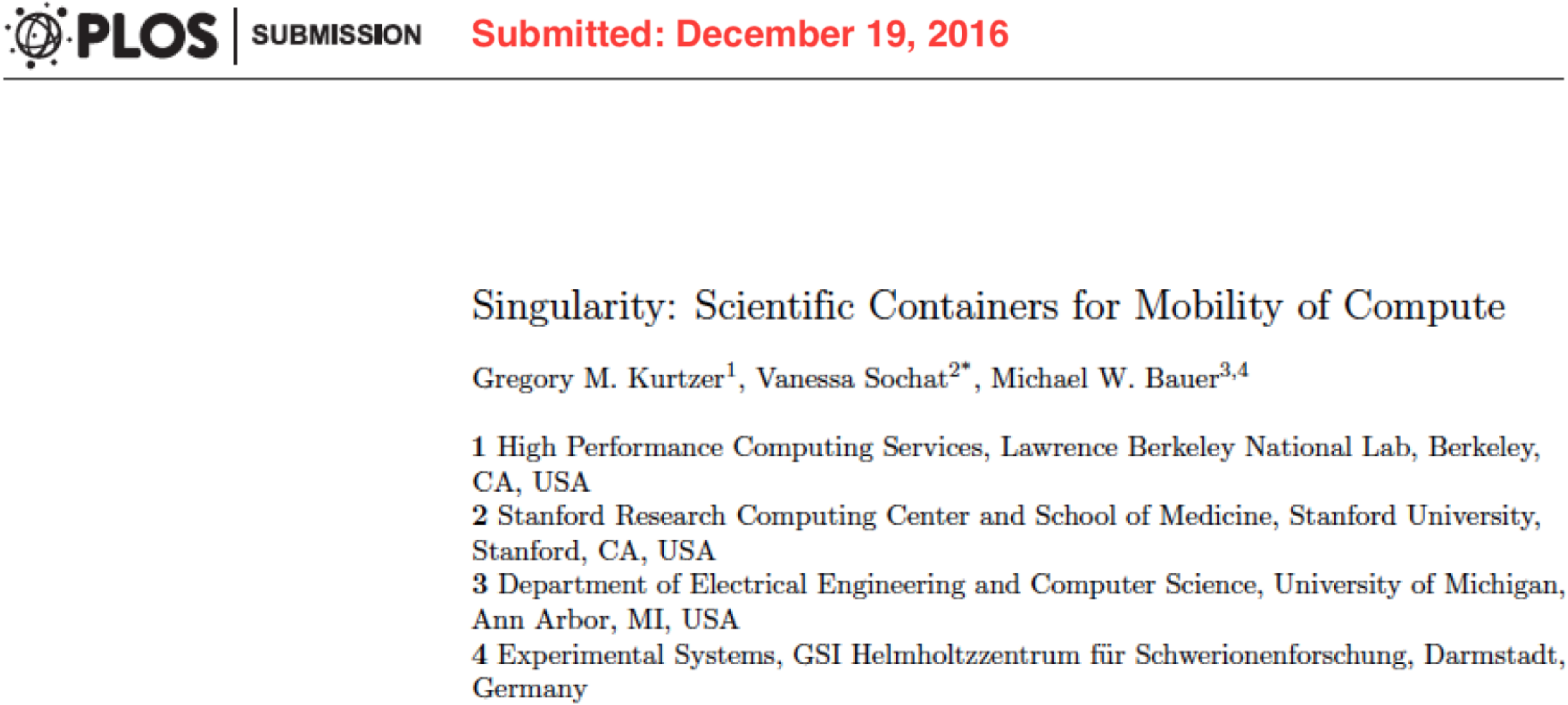


Any container can be run by any user - same user inside container and on host

No workflow changes necessary to use

Single .img file contains everything necessary

Safe to run any container without screening its contents



PLOS | SUBMISSION

Submitted: December 19, 2016

Singularity: Scientific Containers for Mobility of Compute

Gregory M. Kurtzer[1], Vanessa Sochat[2*], Michael W. Bauer[3,4]

1 High Performance Computing Services, Lawrence Berkeley National Lab, Berkeley, CA, USA
2 Stanford Research Computing Center and School of Medicine, Stanford University, Stanford, CA, USA
3 Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, MI, USA
4 Experimental Systems, GSI Helmholtzzentrum für Schwerionenforschung, Darmstadt, Germany

Basic Usage of Singularity

Singularity Workflow

1. Create image file

   ```
   $    sudo singularity create [image]
   ```

2. Bootstrap image

   ```
   $    sudo singularity bootstrap [image] [definition.def]
   ```

3. Run image

   ```
   $    singularity shell [image]
   $    singularity exec [image] [/path/to/executable]
   $    singularity run [image]
   $    ./image
   ```

Format	**Description**
directory	Standard Unix directories containing a root container image
tar.gz	Zlib compressed tar archives
*tar.bz*2	Bzip2 compressed tar archives
tar	Uncompressed tar archives
cpio.gz	Zlib compressed CPIO archives
cpio	Uncompressed CPIO archives

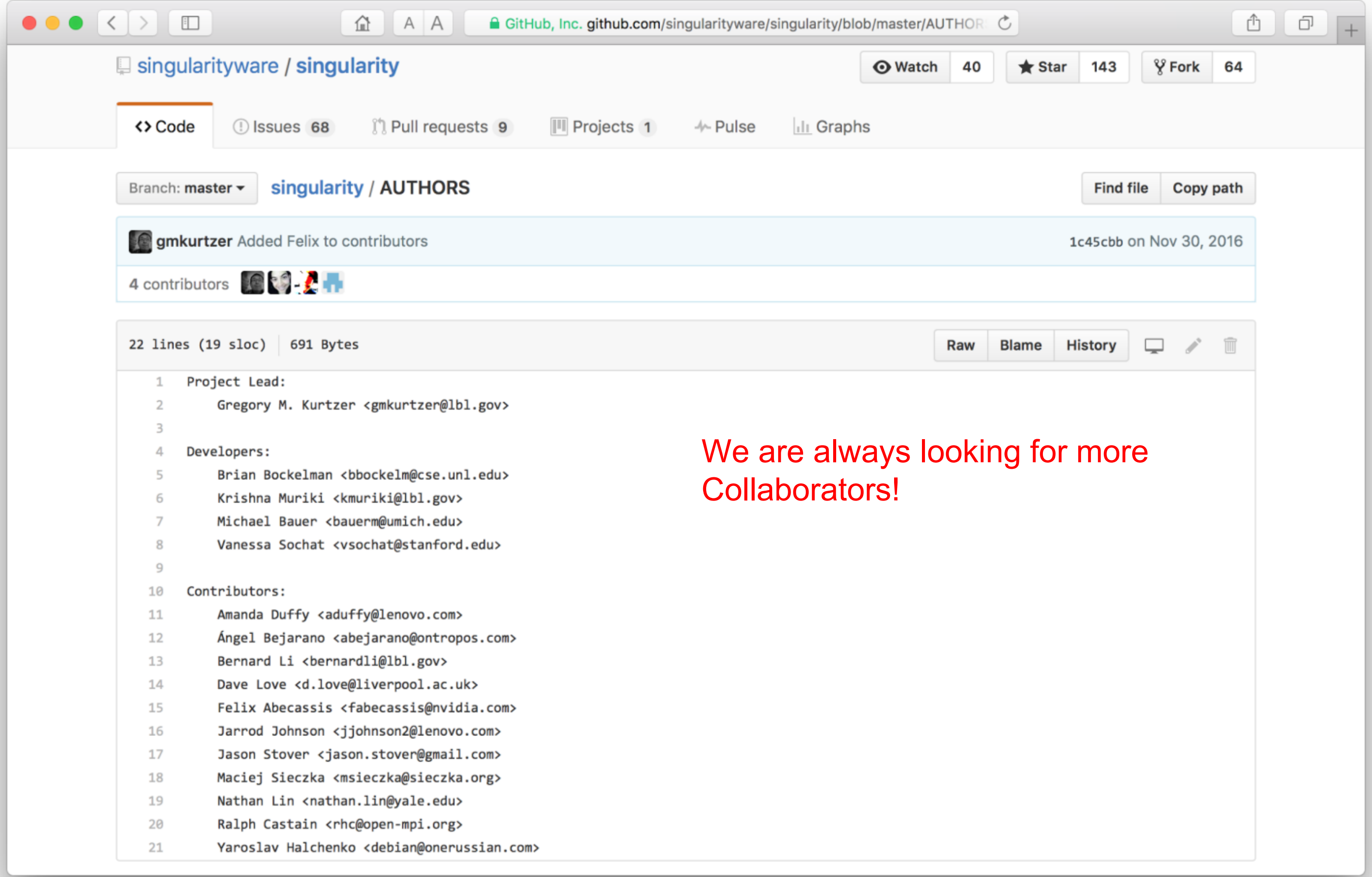
SLURM Integration

```
#!/bin/bash -l

#SBATCH --image=~/centos7/latest
#SBATCH -p debug
#SBATCH -N 64
#SBATCH -t 00:20:00
#SBATCH -J my_job
#SBATCH -L SCRATCH
#SBATCH -C haswell

srun -n 4096 ./mycode.exe      # an extra -c 1 flag is optional for fully packed pure MPI with hyperthreading
```

Global Options	
−d − −debug	Print debugging information
−h − −help	Display usage summary
−q − −quiet	Only print errors
− − version	Show application version
−v − −verbose	Increase verbosity +1
−x − −sh − debug	Print shell wrapper debugging information
General Commands	
help	Show additional help for a command
Container Usage Commands	
exec	Execute a command within container
run	Launch a runscript within container
shell	Run a Bourne shell within container
test	Execute any test code defined within container
Container Management Commands (requires root)	
bootstrap	Bootstrap a new Singularity image
copy	Copy files from your host into the container
create	Create a new container image
export	Export the contents of a container via a tar pipe
import	Import/add container contents via a tar pipe
mount	Mount a Singularity container image



singularity / AUTHORS

gmkurtzer Added Felix to contributors — 1c45cbb on Nov 30, 2016

4 contributors

22 lines (19 sloc) | 691 Bytes

```
Project Lead:
    Gregory M. Kurtzer <gmkurtzer@lbl.gov>

Developers:
    Brian Bockelman <bbockelm@cse.unl.edu>
    Krishna Muriki <kmuriki@lbl.gov>
    Michael Bauer <bauerm@umich.edu>
    Vanessa Sochat <vsochat@stanford.edu>

Contributors:
    Amanda Duffy <aduffy@lenovo.com>
    Ángel Bejarano <abejarano@ontropos.com>
    Bernard Li <bernardli@lbl.gov>
    Dave Love <d.love@liverpool.ac.uk>
    Felix Abecassis <fabecassis@nvidia.com>
    Jarrod Johnson <jjohnson2@lenovo.com>
    Jason Stover <jason.stover@gmail.com>
    Maciej Sieczka <msieczka@sieczka.org>
    Nathan Lin <nathan.lin@yale.edu>
    Ralph Castain <rhc@open-mpi.org>
    Yaroslav Halchenko <debian@onerussian.com>
```

We are always looking for more Collaborators!