Experiences with Extreme-Scale Scientific Software Stack (E4S) and Spack on SDSC systems

E4S Forum Workshop
Albuquerque, Sept 23, 2019
Mahidhar Tatineni, SDSC

SDSC SAN DIEGO SUPERCOMPUTER CENTER

UC San Diego

Overview

- SDSC Overview – Background and info on HPC systems
- Motivation for using E4S and Spack
- Spack install, testing, and results, including application benchmarks
- Background on Singularity usage at SDSC
- Preliminary tests using E4S stack
  - Spack installing components
  - Using E4S Singularity image.

SDSC is one of the original NSF-funded Supercomputer Centers

- Established 1985 as one of original NSF-funded supercomputer centers
- Transitioned from General Atomics to UCSD in 1997
- ~$27M in annual external revenues, 225 staff
- >$1B cumulative revenues
- Worldwide brand and credentials in high performance computing, big data, and scientific data management

SDSC Overview

- 76,000 square feet of office/meeting/conference space
- 12,000 sq. ft. main data center
- 5,000 sq. ft. secure data center (FISMA Moderate security level)
- 4 major supercomputers (Comet, GordonS, Popeye, Triton Cluster)
- ~15 petabytes capacity high performance storage systems
- High bandwidth network connectivity (Internet2, California CENIC, DOE ESNet, Amazon Cloud routing)
- Dark fiber terminations for connectivity to industry/government research facilities in San Diego

Recent NSF funded SDSC HPC systems

- 2010: Awarded $20M+ for "**Flash Gordon**" the first large HPC system to feature widespread use of NVM
- 2011: Deployed **Dash**, a prototype system for Gordon that allowed the exploration of features planned for Gordon
- 2011: Deployed **Trestles**, which featured SSDs on all nodes
- 2012: **Gordon** entered production
- 2015: **Comet** enters production. Comet is designed to support the "long tail", featuring innovations in virtualization, support for science gateways and high-performance storage systems

Comet is SDSC's most recent HPC system will be in production through March 2021

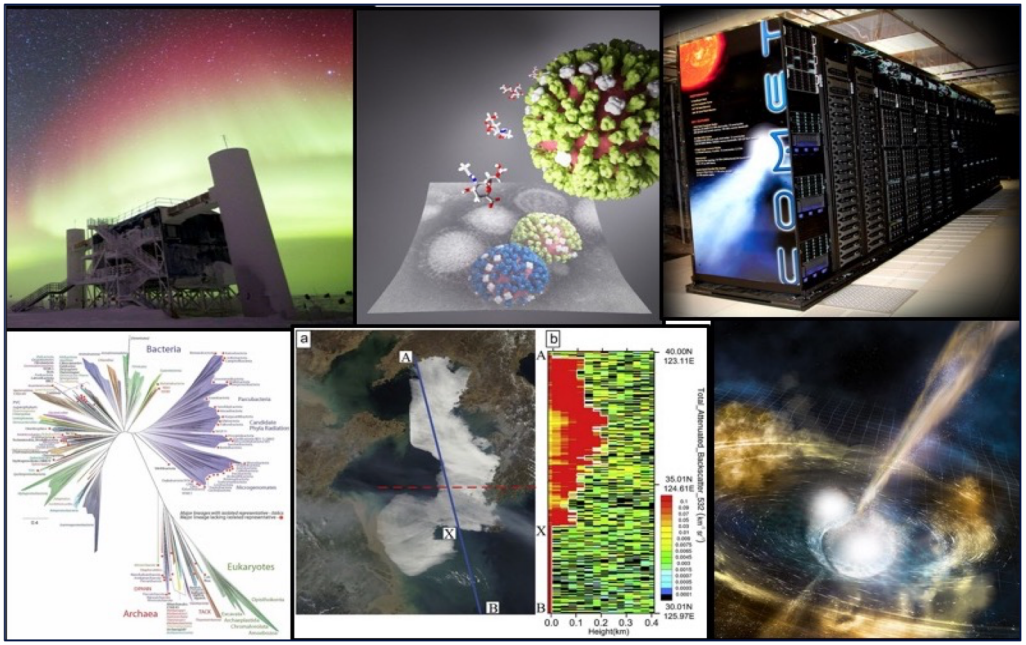
Gateways to Discovery: Cyberinfrastructure for the Long Tail of Science (NSF 1341698)

- *First HPC system to deploy high performance virtualization*
- *Hybrid fat tree FDR interconnect designed to favor modest-scale jobs*
- *Designed to support science gateways*
- *Allocation and management policies for rapid turnaround*
- *~ 650 TB of SSD*

Comet: System Characteristics

- Total peak flops ~2.76 PF
- Dell primary integrator
  - *Intel Haswell processors w/ AVX2*
  - *Mellanox FDR InfiniBand*
- 1,944 standard compute nodes (46,656 cores)
  - *Dual CPUs, each 12-core, 2.5 GHz*
  - *128 GB DDR4 2133 MHz DRAM*
  - *2*160GB GB SSDs (local disk)*
- 72 GPU nodes
  - *36 nodes with two NVIDIA K80 cards, each with dual Kepler3 GPUs*
  - *36 nodes with 4 P100 GPUs each*
- 4 large-memory nodes
  - *1.5 TB DDR4 1866 MHz DRAM*
  - *Four Haswell processors/node*
  - *64 cores/node*
- Hybrid fat-tree topology
  - *FDR (56 Gbps) InfiniBand*
  - *Rack-level (72 nodes, 1,728 cores) full bisection bandwidth*
  - *4:1 oversubscription cross-rack*
- Performance Storage (Aeon)
  - *7.6 PB, 200 GB/s; Lustre*
  - *Scratch & Persistent Storage segments*
- Durable Storage (Aeon)
  - *6 PB, 100 GB/s; Lustre*
  - *Automatic backups of critical data*
- Home directory storage
- Gateway hosting nodes
- Virtual image repository
- 100 Gbps external connectivity to Internet2 & ESNet

Comet advances science and engineering discovery for a broad user base



Clockwise from upper left: IceCube Neutrino Detection; Battling Influenza; Comet Surpasses 40,000 Users; Detecting Gravitational Waves; Predicting Sea Fog; Defining a New Tree of Life

In just over 4 years:

- 40,000+ Unique Users
- 1,200+ Publications
- ~2,000 Research, education and startup allocations
- 400+ Institutions
- Scientific discoveries and breakthroughs

TSCC System Overview

- **Total "Condo" nodes: 254 (37 groups)**
- **Total "Hotel" nodes: 52**
- **GPU nodes: 50 in Condo, 5 in Hotel (included in counts above)**
- **General compute nodes:**
  - Mix of dual-socket Sandy Bridge, Haswell, Broadwell, Cascade Lake
  - 16-28 cores/node, mix of 64GB and 128GB memory
  - Mixed 10GbE and QDR Infiniband interconnect
- **GPU nodes:**
  - Mix of NVIDIA GTX 680, 780, 980, 1080, 2080 and Titan-X

FUTURE SYSTEM: EXPANSE (Oct 2020)

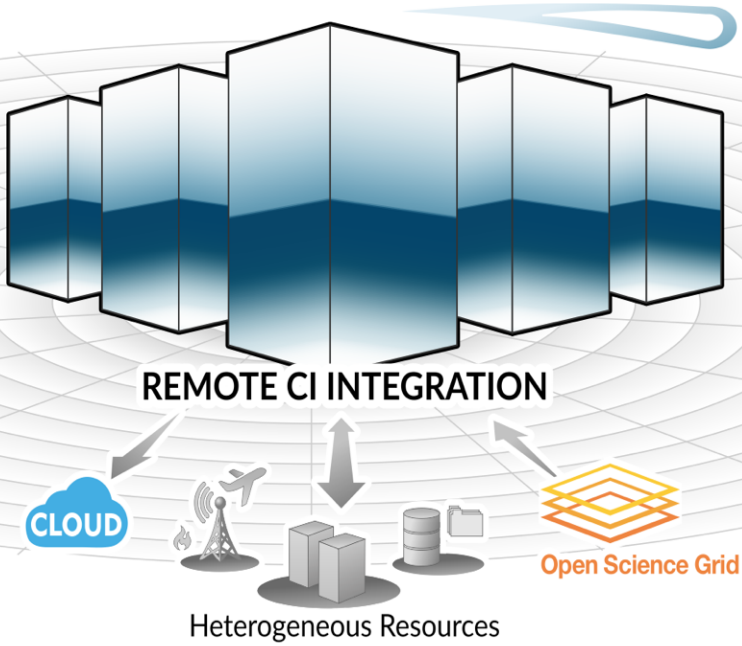
Computing Without Boundaries: Cyberinfrastructure for the Long Tail of Science
NSF Solicitation 19-534: Advanced Computing Systems & Services: Adapting to the Rapid Evolution of Science and Engineering Research
Category 1: Capacity System, NSF Award # 1928224
PIs: Mike Norman (PI), Ilkay Altintas, Amit Majumdar, Mahidhar Tatineni, Shawn Strande

EXPANSE

COMPUTING WITHOUT BOUNDARIES
5 PETAFLOP/S HPC and DATA RESOURCE

HPC RESOURCE
13 Scalable Compute Units
728 Standard Compute Nodes
52 GPU Nodes: 208 GPUs
4 Large Memory Nodes

DATA CENTRIC ARCHITECTURE
12PB Perf. Storage: 140GB/s, 200k IOPS
Fast I/O Node-Local NVMe Storage
7PB Ceph Object Storage
High-Performance R&E Networking



LONG-TAIL SCIENCE
Multi-Messenger Astronomy
Genomics
Earth Science
Social Science

INNOVATIVE OPERATIONS
Composable Systems
High-Throughput Computing
Science Gateways
Interactive Computing
Containerized Computing
Cloud Bursting

Disclaimer:
The system has not yet been procured and technical details are subject to change.

Libraries and Applications Software Current Approach on SDSC systems

- Users can manage environment via modules.
- Applications packaged into “Rocks Rolls” that can built and deployed on any of the SDSC systems. Benefits wider community deploying software on their Rocks clusters.
- Efficient system administration pooling software install/testing efforts from different projects/machines – Comet benefits from work done for Trestles, Gordon, and Triton Shared Computing Cluster (TSCC).
- Users benefit from a familiar applications environment across SDSC systems => can easily transition to Comet, TSCC from older systems.
- Rolls available for Rocks community (**https://github.com/sdsc**)

Motivation for Spack based approach

- SDSC supports many systems, with thousands of users, and a broad software stack with a small user support team. The motivation is to support a large, diverse software environment as efficiently as possible.
- Leverage work of the wider Spack community for installs.
- SDSC clusters feature a broad range of CPU and GPU architectures. Helps to have ability to have multiple installs – customizing and optimizing for specific targets.
- Easier for users to do customizations – chained Spack installs, environments.
- Future systems (like Expanse) feature cloud integration, composable options. Spack based approach can help simplify the software management.

Applicability of Extreme-Scale Scientific Software Stack (E4S)

- Current Comet software stack already features a big fraction of the E4S stack. Examples:
  - TAU, PAPI, PDT, Darshan
  - ParaView, Gasnet, OpenMPI, Parallel netCDF, Jupyter
  - SuperLU, Sundials, Trilinos, PETSc, Magma, Hypre

- Leveraging a performant, interoperable software stack will be very useful and save staff time.

Overview of Spack Testing

- *Work started after tutorials at Spack workshop in July.*
- Goal was to replicate as much of the Comet stack as possible using Spack. A quick review of package list suggests we can get 90% done.
- Preliminary installs worked well with relatively minor changes in the configuration. Examples:
  - py-scipy and py-numpy versions in AMBER build
  - AVX512 flags for GROMACS install
- Preliminary benchmark results for two applications – NEURON and RAxML

Snapshot of packages in test install

```
[-bash-4.2$ spack find
==> 184 installed packages
-- linux-centos7-x86_64 / gcc@4.8.5 -----------------------------
apr@1.6.2                intel-mkl@2019.3.199          perl-extutils-makemaker@7.24
apr-util@1.6.0           intel-tbb@2019.4              perl-extutils-pkgconfig@1.16
autoconf@2.69            libbeagle@3.1.2               perl-font-ttf@1.06
automake@1.16.1          libbsd@0.9.1                  perl-gd@2.53
bamtools@2.5.1           libffi@3.2.1                  perl-gd-graph@1.4308
bdftopcf@1.0.5           libfontenc@1.1.3              perl-gd-text@0.86
beast1@1.10.4            libgd@2.2.4                   perl-module-build@0.4224
beast2@2.5.2             libiconv@1.15                 perl-padwalker@2.2
binutils@2.32            libjpeg-turbo@2.0.2           perl-pdf-api2@2.033
bismark@0.19.0           libpciaccess@0.13.5           perl-sub-uplevel@0.2800
blat@35                  libpng@1.6.34                 perl-test-exception@0.43
bowtie@1.2.3             libsigsegv@2.11               perl-test-memory-cycle@1.06
bowtie2@2.3.5.1          libsvm@322                    pkgconf@1.6.1
bwa@0.7.17               libtiff@4.0.10                py-setuptools@41.0.1
bzip2@1.0.8              libtool@2.4.6                 python@2.7.16
charmpp@6.9.0            libxfont@1.5.2                randfold@2.0.1
cmake@3.15.3             libxml2@2.9.9                 raxml@8.2.11
curl@7.63.0              lmod@8.1.5                    readline@7.0
diffutils@3.7            lua@5.3.5                     samtools@1.9
expat@2.2.5              lua-luafilesystem@1_7_0_2     scons@3.1.0
fastq-screen@0.11.2      lua-luaposix@33.4.0           serf@1.3.9
fastqc@0.11.7            m4@1.4.18                     soapdenovo2@240
fftw@3.3.8               mkfontdir@1.0.7               sparsehash@2.0.3
fftw@3.3.8               mkfontscale@1.1.2             sqlite@3.29.0
findutils@4.6.0          mpich@3.2.1                   squid@1.9g
font-util@1.3.2          mrbayes@2017-11-22            subversion@1.9.7
fontconfig@2.12.3        nasm@2.14.02                  tar@1.31
fontsproto@2.1.3         ncurses@6.1                   tcl@8.6.8
freetype@2.9.1           numactl@2.0.12                texinfo@6.5
gdbm@1.18.1              openjdk@11.0.2                unzip@6.0
gettext@0.19.8.1         openmpi@3.1.4                 util-macros@1.19.1
git@2.21.0               openssl@1.1.1c                velvet@1.2.10
gperf@3.0.4              pcre@8.42                     xproto@7.0.31
gsl@2.5                  perl@5.26.2                   xtrans@1.3.5
htslib@1.9               perl-capture-tiny@0.46        xz@5.2.4
hwloc@1.11.11            perl-devel-cycle@1.12         zlib@1.2.11
```

```
-- linux-centos7-x86_64 / intel@19.0.3.199 ----------------------
autoconf@2.69      libffi@3.2.1           perl@5.26.2
autoconf@2.69      libiconv@1.15          perl-devel-cycle@1.12
automake@1.16.1    libiconv@1.15          perl-font-ttf@1.06
automake@1.16.1    libpciaccess@0.13.5    perl-padwalker@2.2
bison@3.0.5        libpciaccess@0.13.5    perl-pdf-api2@2.033
bowtie@1.2.3       libsigsegv@2.11        perl-sub-uplevel@0.2800
bzip2@1.0.8        libsigsegv@2.11        perl-test-exception@0.43
cmake@3.9.4        libtool@2.4.6          perl-test-memory-cycle@1.06
cmake@3.15.3       libtool@2.4.6          pkgconf@1.6.1
cmake@3.15.3       libxml2@2.9.9          pkgconf@1.6.1
cuda@10.1.243      libxml2@2.9.9          python@2.7.16
diffutils@3.7      m4@1.4.18              readline@7.0
expat@2.2.5        m4@1.4.18              readline@7.0
fftw@3.3.8         migrate@3.6.11         readline@7.0
flex@2.6.4         ncurses@6.1            sparsehash@2.0.3
gdbm@1.18.1        ncurses@6.1            sqlite@3.29.0
gdbm@1.18.1        netcdf@4.7.0           squid@1.9g
gdbm@1.18.1        netcdf-fortran@4.4.5   tar@1.31
gettext@0.19.8.1   neuron@7.5             util-macros@1.19.1
gromacs@2019.2     numactl@2.0.12         util-macros@1.19.1
hdf5@1.10.5        numactl@2.0.12         xz@5.2.4
hdf5@1.10.5        openmpi@3.1.4          xz@5.2.4
help2man@1.47.8    openmpi@3.1.4          zlib@1.2.11
hwloc@1.11.11      openssl@1.1.1c         zlib@1.2.11
hwloc@1.11.11      openssl@1.1.1c
libbsd@0.9.1       perl@5.26.2
[-bash-4.2$
```

Overview of Codes Benchmarked

- **NEURON** simulates individual neurons or networks of neurons. The code is parallelized with MPI. The benchmark problem, which is typical of those run on *Comet* via the Neuroscience Gateway, is simulation of a large-scale model of the olfactory bulb.

- **RAxML** performs phylogenetic tree inference using a maximum likelihood approach. It has hybrid MPI/Pthreads parallelization. The benchmark problem, which is typical of those run on *Comet* via the CIPRES science gateway, is an all-in-one analysis. That consists of 20 regular searches and 100 bootstrap searches, followed by mapping of the bootstrap support to the best tree from the regular searches.

NEURON: Model of olfactory bulb:10,500 cells, 40K timesteps

MPI Tasks	Baseline (s)	Spack install version (s)
96	496	485
192	236	234
384	107	98
768	46	49

RAxML : 218 taxa, 2294 DNA, 1846 patterns, 100 bootstraps

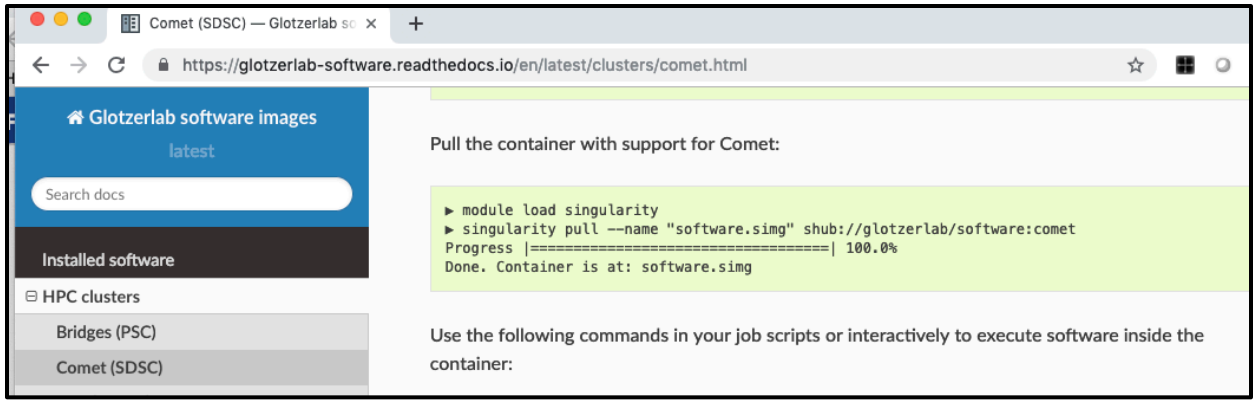
Tasks	Baseline (s)	Spack install version (s)
10	594	670
20	372	415
30	292	296
40	261	245

Singularity on SDSC Resources

Singularity: Provides Flexibility and Portability

- Singularity has been available on Comet since 2016 and its become very popular on Comet.
- Singularity runs in user space, and requires very little user support. A lot of our users run on multiple systems and some use the Singularity approach to simplify porting.
- Singularity allows groups to easily migrate complex software stacks and workflows to Comet.

Glotzer Lab
Univ. Michigan
HOOMD-Blue and other packages



Singularity Use Cases

- Applications with newer library OS requirements than available on the HPC system – e.g. TensorFlow, PyTorch, Caffe2 (SDSC staff maintain optimal versions for Comet).
- Commercial application binaries with specific OS requirements.
- Importing singularity and docker images to enable use in a shared HPC environment. Usually this is entire workflows with a large set of tools bundled in one image.
- Training – encapsulate all the requirements in an image for workshops and SDSC summer institute. Also makes it easy for users to try out outside of the training accounts on Comet.

Expanse: Integration with public cloud supports projects that share data, need access to novel technologies, and integrate cloud resources into workflows

- Slurm + in-house developed software + Terraform (Hashicorp)
- Early work funded internally and via NSF E-CAS/Internet2 project for CIPRES (Exploring Cloud for the Acceleration of Science, Award #1904444).
- Approach is cloud-agnostic and will support the major cloud providers
- Users submit directly via the Slurm, or as part of a composed system
- Options for data movement: data in the cloud; remote mounting of file systems; cached filesystems (e.g., StashCache), and data transfer during the job.
- New CC* Award (1925558), Triton Stratus, will integrate campus cluster with commercial cloud

* **Funding for user cloud resources is not part of the Expanse award. Researcher must have access to these via other NSF awards and funding.**

=> Motivation for containerized solutions, simplified build processes

E4S Testing Overview

- **Two install/usage approaches**
  - Spack based install
  - E4S Singularity image
- **Functional testing**
  - Demo NPB run using Singularity image
  - HPCToolkit
  - TensorFlow example
- **User application builds using E4S packages (Planned)**
  - Trilinos, PETSc, Hypre, Metis based codes

Singularity based NPB test script

```
[-bash-4.2$ more run.hpctoolkit.sh
module load singularity
export PATH=/home/mahidhar/ECP/demo/NPB3.1/bin:$PATH
export MV2_ENABLE_AFFINITY=0
mpirun -np 256 singularity exec -B /etc/libibverbs.d:/etc/libibverbs.d -B /usr/l
ib64:/hostlib64  -B /opt:/opt -B /home/mahidhar:/scratch1 /home/mahidhar/ECP/ima
ge/ecp.simg /bin/bash -c ' . /etc/bashrc; spack unload mpich openmpi; spack load
 gcc; spack load hpctoolkit; export LD_LIBRARY_PATH=/scratch1/SPACK/spack/opt/sp
ack/linux-centos7-x86_64/gcc-4.8.5/mpich-3.2.1-kynrgt3zjvrpg5u24a3ilehfxuqw56zb/
lib:$LD_LIBRARY_PATH:/hostlib64:/hostlib64/libibverbs; hpcrun -e CYCLES ./mg.D.2
56'
[-bash-4.2$ ls
hpctoolkit-mg.D.256-measurements-38787  mg.D.256.8node.hpctk.out  run.out
lu.C.64                                 mg.D.256.8node.out        run.sh
mg.D.256                                run.hpctoolkit.sh
```

MG Class D Output Verification

```
VERIFICATION SUCCESSFUL
L2 Norm is    0.158327506042E-09
Error is      0.511151127200E-11


MG Benchmark Completed.
Class           =                        D
Size            =          1024x1024x1024
Iterations      =                       50
Time in seconds =                    33.93
Total processes =                      256
Compiled procs  =                      256
Mop/s total     =                 91773.09
Mop/s/process   =                   358.49
Operation type  =           floating point
Verification    =               SUCCESSFUL
Version         =                      3.1
Compile date    =              21 Sep 2019
```

TensorFlow Example with MNIST datset

```
Singularity ecp.simg:~/SPACK/benchmarks/MNIST> python main.py
Successfully downloaded train-images-idx3-ubyte.gz 9912422 bytes.
Extracting MNIST_data/train-images-idx3-ubyte.gz
Successfully downloaded train-labels-idx1-ubyte.gz 28881 bytes.
Extracting MNIST_data/train-labels-idx1-ubyte.gz
Successfully downloaded t10k-images-idx3-ubyte.gz 1648877 bytes.
Extracting MNIST_data/t10k-images-idx3-ubyte.gz
Successfully downloaded t10k-labels-idx1-ubyte.gz 4542 bytes.
Extracting MNIST_data/t10k-labels-idx1-ubyte.gz
WARNING:tensorflow:From main.py:38: softmax_cross_entropy_with_logits (from tensorflo
w.python.ops.nn_ops) is deprecated and will be removed in a future version.
Instructions for updating:

Future major versions of TensorFlow will allow gradients to flow
into the labels input on backprop by default.

See tf.nn.softmax_cross_entropy_with_logits_v2.

2019-09-23 09:38:55.061807: I tensorflow/core/platform/cpu_feature_guard.cc:137] Your
 CPU supports instructions that this TensorFlow binary was not compiled to use: AVX2
AVX512F FMA
Iteration 0     | Loss = 3.82887        | Accuracy = 0.078125
Iteration 100   | Loss = 0.578979       | Accuracy = 0.828125
Iteration 200   | Loss = 0.323711       | Accuracy = 0.898438
Iteration 300   | Loss = 0.27962        | Accuracy = 0.90625
Iteration 400   | Loss = 0.609797       | Accuracy = 0.8125
```

Summary

- Building up software stack using Spack to match production stack on Comet.
- Installs working well so far – only needed minor modifications.
- Preliminary benchmark results show performance is good (with arch specific flags)
- E4S components installed via Spack and also tested using ECP singularity image.
- Future work includes completing the full stack build, setting up local Spack repo, user application testing.

Thank you!

Mahidhar Tatineni

mahidhar@sdsc.edu