# A Framework for Benchmarking and Testing HPC Applications for the SDP

**PRACE-CERN-GÉANT-SKAO kick-off workshop on High Performance Computing**

John Taylor, Steve Brasier, John Garbutt
firstname.secondname@stackhpc.com

StackHPC

# Contents

- Context & aims
- Preliminary Study
- Proposed implementation

# Context

- Characterise performance differences for Systems Under Test
- Reduce differences if possible via tuning
- Document results
- Maintain reproducibility and repeatability across SUT
- Optimise solution space

# Preliminary Study

StackHPC

- Aims:
  - Characterise performance differences for Infiniband vs. Ethernet (RDMA)
  - Test matrix - choose a range of industry standard benchmarks
  - Reduce differences if possible via profiling and monitoring
  - Document results
- Based on previous tests by John Taylor: High Performance Ethernet for HPC – Are we there yet?

# Anticipated Problems

- Test matrix complexity
  - IB vs RoCE
  - MPI libraries
  - Number of nodes + processes etc
  - System parameters
  - Tuning parameters
- Changes to test matrix
  - *“MPI x version y has just come out, can we try that?”*
- Correctness & Repeatability
  - Was the right combination actually run?
  - System changes

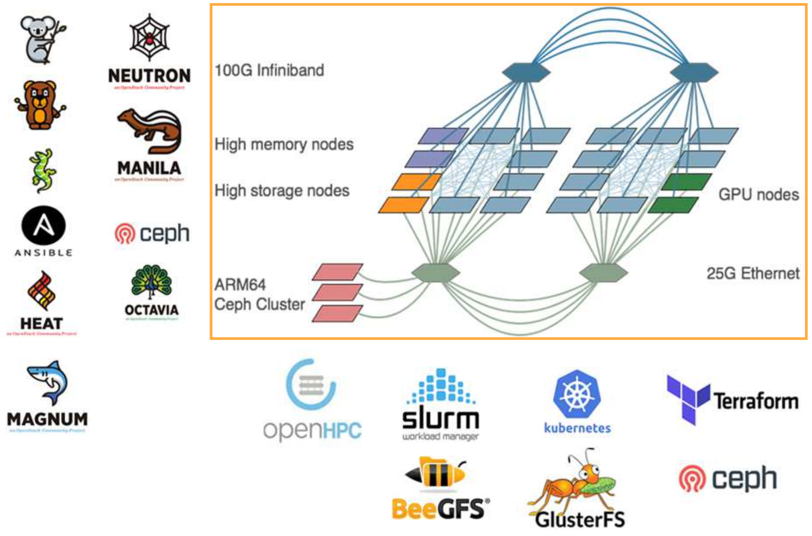
# Proposed Solution

- Testing-as-code: https://github.com/stackhpc/hpc-tests
- Propose using ReFrame - HPC regression tests
  - Tests defined in python, ReFrame handles interaction with system
  - Easy to define/integrate results extraction and processing
  - Some extension to functionality required: presently only done for slurm
  - In production use at CSCS, NERSC, OSC, responsive developers
- Automate build process
- Monitoring:
  - (Software-defined) monitoring for live view of system with context from tests

# Test Hardware

StackHPC

- Two OpenStack Bare Metal Clouds
    - AlaSKA - Baseline System
        - 25GbE RoCE
        - 100 Gbps EDR
        - Broadwell (HT-on)
        - Up to 16 nodes Slurm, K8s
        - OpenHPC Image
    - Cambridge CSD3
        - Cascade lake (HT-off)
        - 50  GbE RoCE
        - 100 Gbps - HDR100
        - Currently up to 56 nodes (Larger by end of the week)
        - Customised Image
        - Large A100 system later in the year!



# ReFrame

Three key aspects:

- System configuration broken down into:
  - Systems
  - Partitions - logical divisions
  - Environments - software configuration
- Tests don't need to know about any configuration
- Outputs (from ReFrame's PoV) are:
  - **Test outputs**: e.g. stdout/stderr/files
  - **performance variable logs**
- All under source control

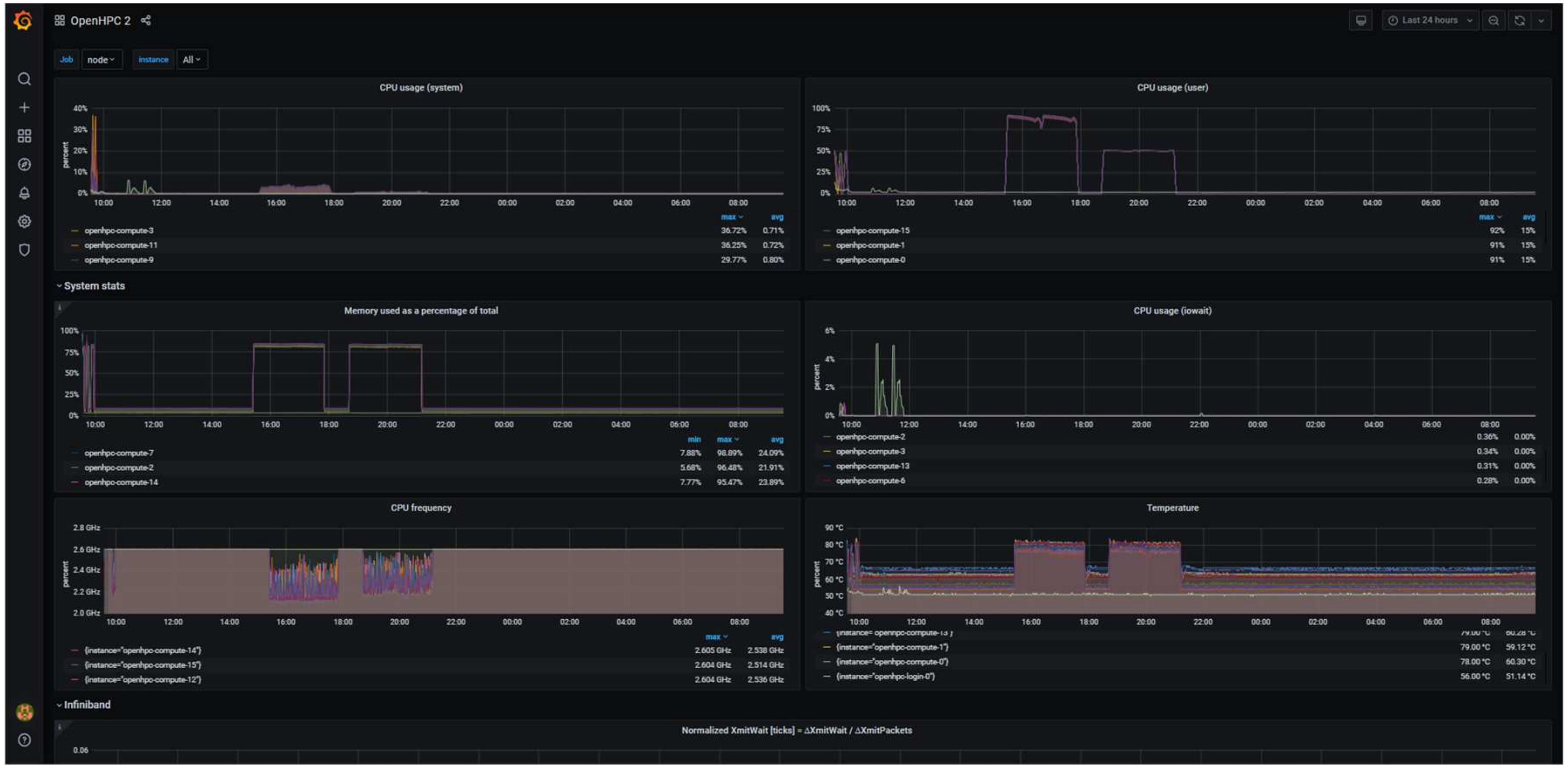
# Results Processing

Plots/tables/reports via jupyter notebooks:

- Web-based interactive python notebook
- Pages rendered in github

Current demo:

- System info
- Includes automated setup of self-signed key for https:// server
- Separate notebook per-test:
  - IMB - plot of raw results, performance variable history
  - Gromacs - performance variable scaling w/ nodes & history
- Shared code between tests e.g. basic plots of performance variable history



CPU usage (system)

| | max | avg |
|---|---|---|
| openhpc-compute-3 | 36.72% | 0.71% |
| openhpc-compute-11 | 36.25% | 0.72% |
| openhpc-compute-9 | 29.77% | 0.80% |

CPU usage (user)

| | max | avg |
|---|---|---|
| openhpc-compute-15 | 92% | 15% |
| openhpc-compute-1 | 91% | 15% |
| openhpc-compute-0 | 91% | 15% |

Memory used as a percentage of total

| | min | max | avg |
|---|---|---|---|
| openhpc-compute-7 | 7.88% | 98.89% | 24.09% |
| openhpc-compute-2 | 5.68% | 96.48% | 21.91% |
| openhpc-compute-14 | 7.77% | 95.47% | 23.89% |

CPU usage (iowait)

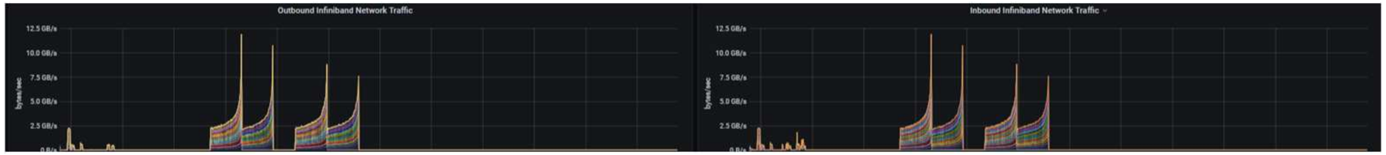
| | | |
|---|---|---|
| openhpc-compute-2 | 0.36% | 0.00% |
| openhpc-compute-3 | 0.34% | 0.00% |
| openhpc-compute-13 | 0.31% | 0.00% |
| openhpc-compute-6 | 0.28% | 0.00% |

CPU frequency

| | max | avg |
|---|---|---|
| {instance="openhpc-compute-14"} | 2.605 GHz | 2.538 GHz |
| {instance="openhpc-compute-15"} | 2.604 GHz | 2.514 GHz |
| {instance="openhpc-compute-12"} | 2.604 GHz | 2.536 GHz |

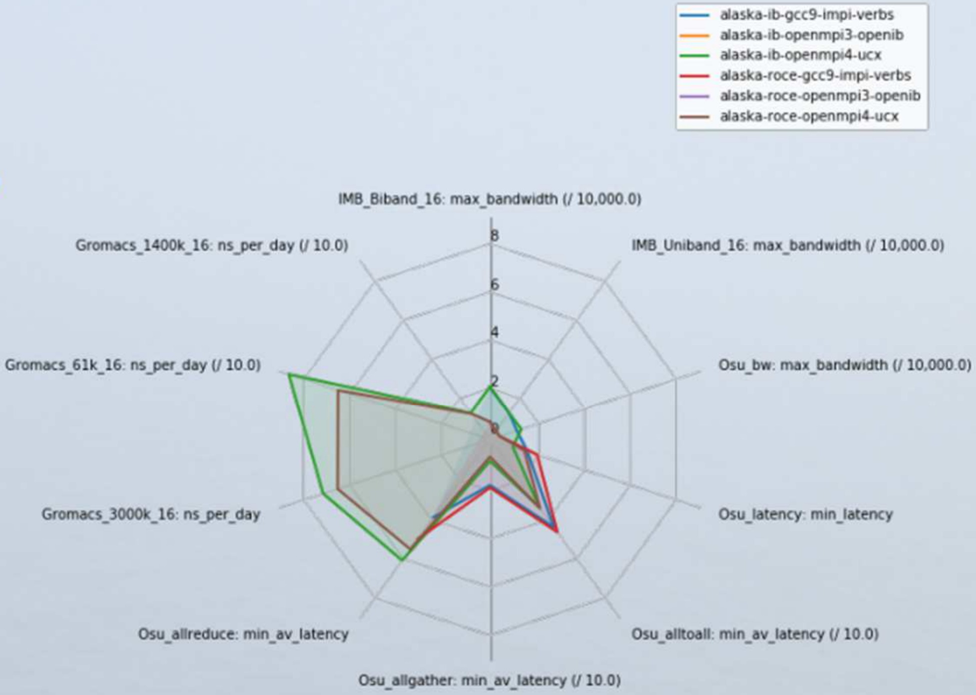
Temperature

| | | |
|---|---|---|
| {instance="openhpc-compute-13"} | 79.00 °C | 60.28 °C |
| {instance="openhpc-compute-1"} | 79.00 °C | 59.12 °C |
| {instance="openhpc-compute-0"} | 78.00 °C | 60.30 °C |
| {instance="openhpc-login-0"} | 56.00 °C | 51.14 °C |



# Optimisation for Key Workloads

- Ensure that infrastructure meets required levels of application performance
- Ensure that performance levels do not regress after reconfiguration
- Find optimal combinations of application and infrastructure configuration



Thanks to
Steve Brasier, John Garbutt,
UIS at the University of Cambridge

# Back Up Slides

# Application Install

- Some packages available via *openhpc* repos.
- Also using *spack*: source-based package manager - no root required
- (Multiple) installs define version, build options, compiler, dependencies (mpi)

E.g.:

```
spack install gromacs@2016.4 ^openmpi@4: fabrics=ucx
schedulers=auto
```

- Show `spack info gromacs`
- Integrates with `lmod` and therefore with openhpc & ReFrame (docs for this somewhat lacking)

# Tests & Benchmarks

Synthetics:

- MPI OSU (latency, bandwidth, alltoall, allgather, allreduce):
  - various options here
- IMB: uniband, biband
- HPL
- HPCG

# Tests & Benchmarks

Applications

- GROMACS, NAMD (molecular dynamics): HecBioSim 61k/1.5M?/ 3M atoms:
  - One or both codes?
  - Gromacs 1.4M run so far (also used for Archer “small” benchmark)
  - Can only use 2018.x Gromacs
- LS-Dyna (dynamic FEA): Neon (*neon_refined_revised?*), car2car, ODB-10M.
  - Licences? LSTC licence server too.
- Star-CCM+ (CFD): LeMans_100M, TurboCharger, Civil 20M
  - Licences? Flex-LM licence server too
- WRF (CONUS2.5, 12.5 and customer dataset)
  - CONUS require <= WRFV3.8.1. Difficulty getting convergence in previous tests
  - Customer dataset?
- Tensorflow: ResNet50

Potentially also relevant from Archer benchmarks: CASTEP, OpenFOAM

# Test Matrix

- Network
  - IB (100GB?)
  - RoCE (25GB)?
- MPI libraries:
  - OpenMPI4 using UCX
  - Intel MPI (using UCX?): Up to 2019.6 available via yum, .7 in release notes? Early 2019.x known to be problematic.
- Launcher: Only use slurm's `srun` for openmpi (via pmix), impi (via pmi2) at least?
- Number of nodes/cpus/gpus + number of jobs + possibly placement/pinning?
- Number of OpenMP threads (where supported)
- Other MPI tuning parameters
- Any application tuning parameters