Buildtest: A Software Testing Framework with Module Operations for HPC systems

Shahzeb Siddiqui (Shahzeb.Siddiqui@3ds.com)

SIMULIA R&D Run Online Operations Senior Manager - Dassault Systemes

11/18/2019

GitHub: https://github.com/HPC-buildtest/buildtest-framework
Documentation: http://buildtest.rtfd.io

whoami

- Duties: User Support Tickets, Scheduler Configuration, Software Installation, System Administration, User Training, Documentation
- Interests: Containers, Scheduler Optimization & Job Analytics, Performance Tuning and System Benchmarking, Parallel Programming, DevOps, Configuration Management
- M.S Computer Science at KAUST
- B.S Computer Engineer at Penn State University

Github: https://github.com/shahzebsiddiqui
LinkedIn: https://www.linkedin.com/in/shahzebmsiddiqui/
Email: shahzebmsiddiqui@gmail.com

GitHub: https://github.com/HPC-buildtest/buildtest-framework
Documentation: http://buildtest.rtfd.io

Background

- HPC Software Stacks are growing at an astronomical rate with up to 1000+ software (open source, commercial), many sites have adopted tools like **Easybuild** or **Spack** to automate software stack build
- HPC Support team will typically install the software and let user test the software
- What happens where there is a software bug?
- Who do you blame: User, Administrator, System, or Package Maintainer?
- HPC Support Team lack the domain expertise to test the software and often too busy with operation support & engineering projects that software testing is often neglected

GitHub: https://github.com/HPC-buildtest/buildtest-framework
Documentation: http://buildtest.rtfd.io

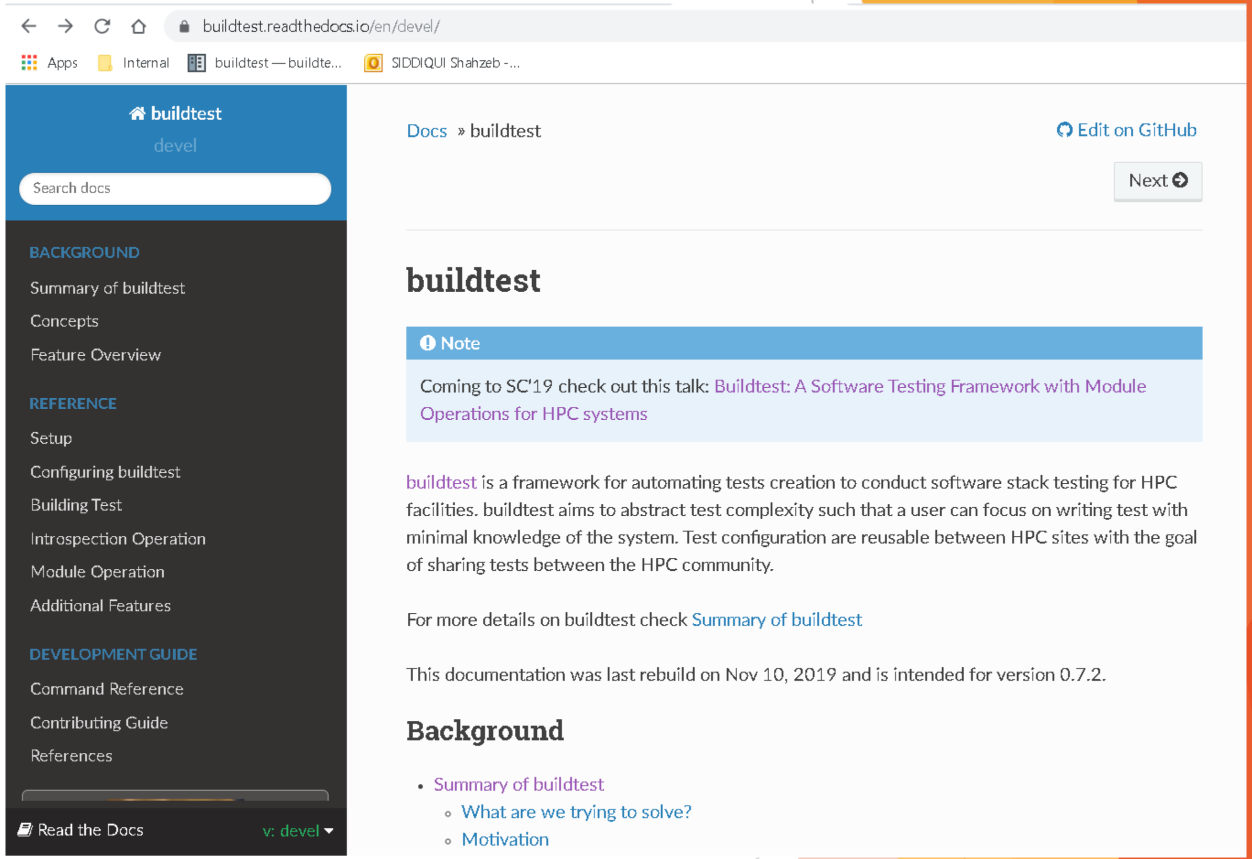
Motivation

- Currently, there is little or no collaboration in HPC community in how to conduct software stack testing
- This demands for concerted effort by HPC community to build an open-source community for software stack testing
- We need to:
  1. Build a framework to do automatic testing of installed software
  2. Build a test repository for scientific software that is community driven and reusable
- An automated test framework is a harness for automating test creation, but it requires community contribution to accumulate this repository on per-package basis

GitHub: https://github.com/HPC-buildtest/buildtest-framework
Documentation: http://buildtest.rtfd.io

What is buildtest

- Buildtest is a framework that:
  - Automates test script creation
  - Abstracts test complexity by using test configuration written in YAML
  - Allows Portable test configurations
  - Provides many software stack operations
- Buildtest comes with a repository of test configuration and source files



buildtest

Note

Coming to SC'19 check out this talk: Buildtest: A Software Testing Framework with Module Operations for HPC systems

buildtest is a framework for automating tests creation to conduct software stack testing for HPC facilities. buildtest aims to abstract test complexity such that a user can focus on writing test with minimal knowledge of the system. Test configuration are reusable between HPC sites with the goal of sharing tests between the HPC community.

For more details on buildtest check Summary of buildtest

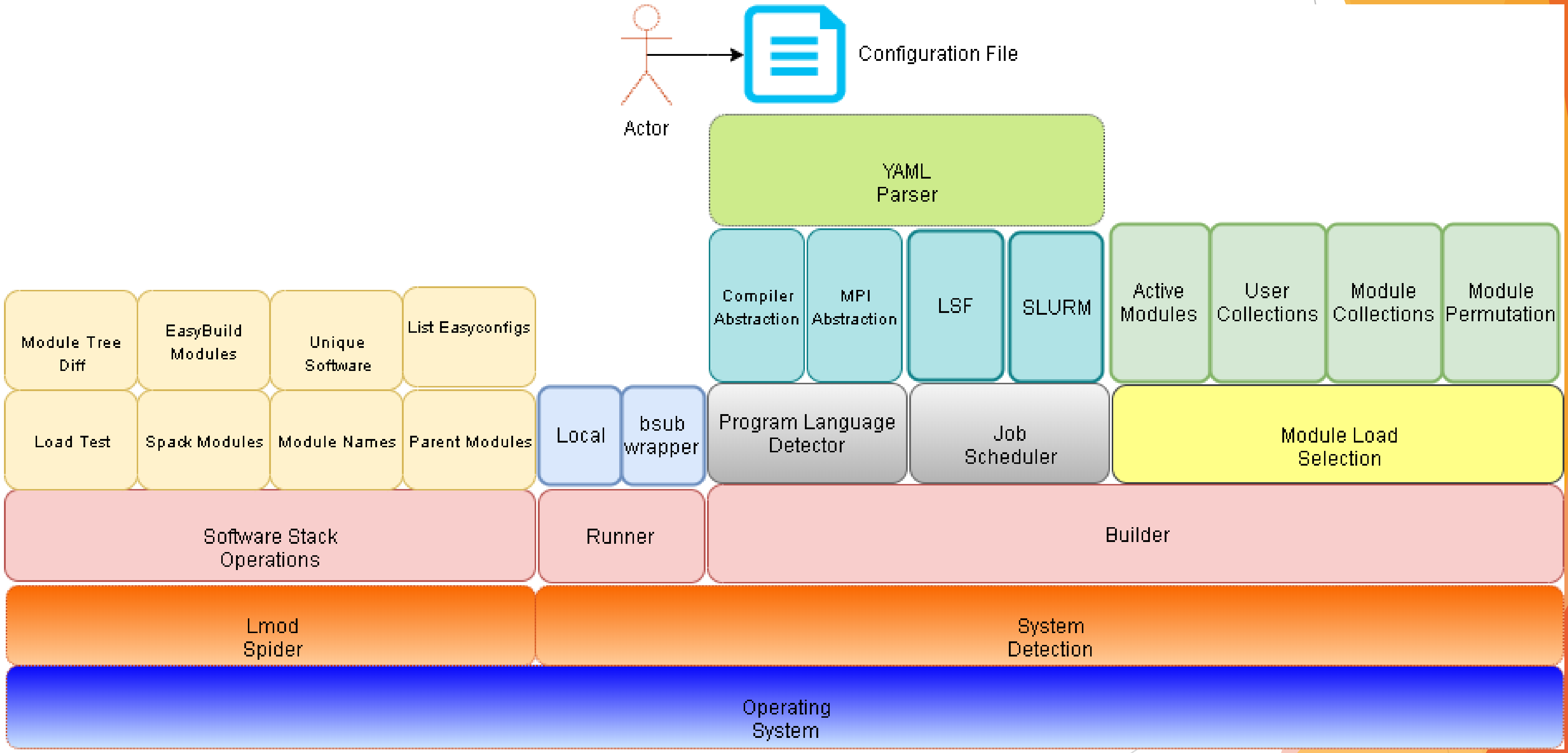
This documentation was last rebuild on Nov 10, 2019 and is intended for version 0.7.2.

Background

- Summary of buildtest
  - What are we trying to solve?
  - Motivation

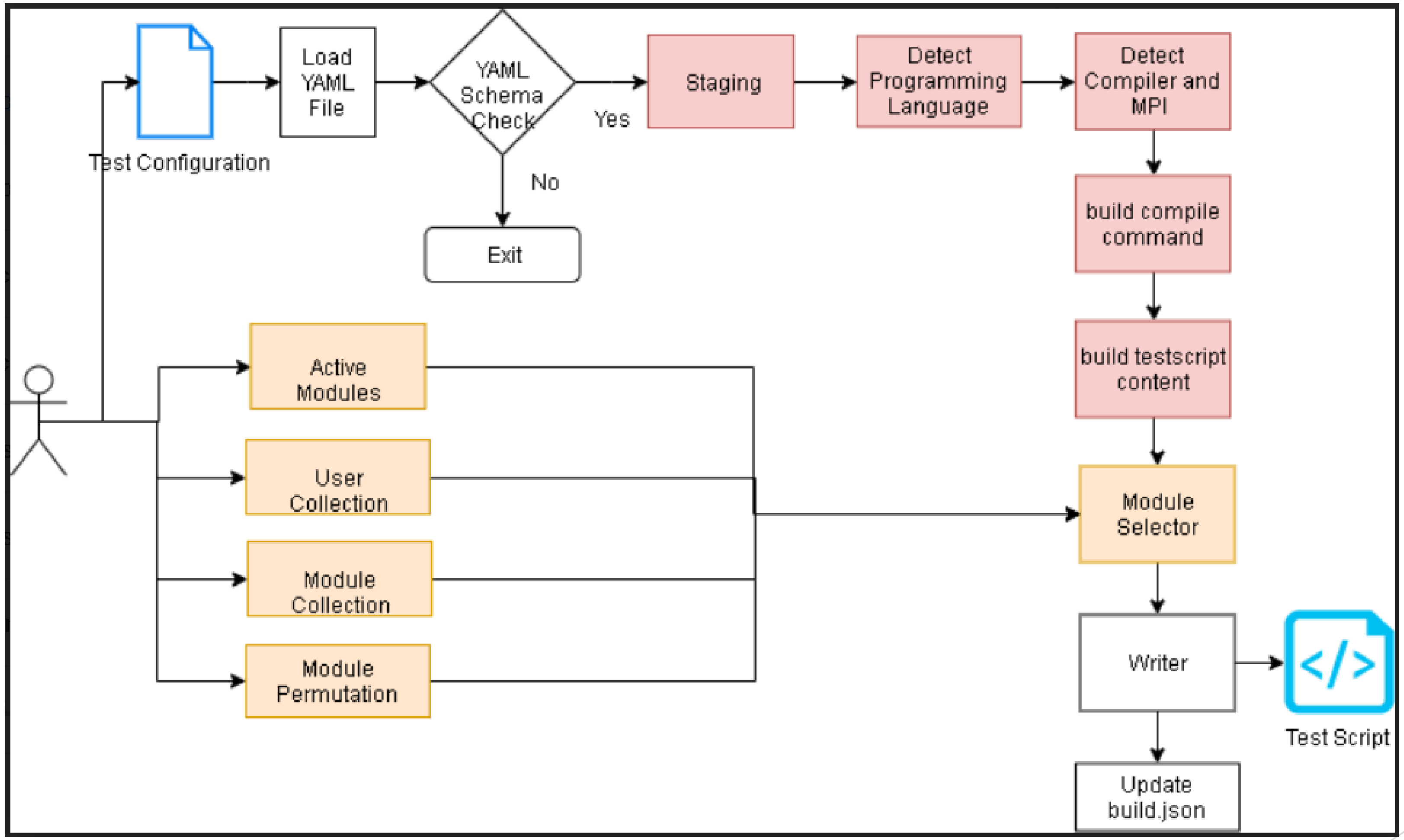
GitHub: https://github.com/HPC-buildtest/buildtest-framework
Documentation: http://buildtest.rtfd.io

Buildtest Architecture



GitHub: https://github.com/HPC-buildtest/buildtest-framework
Documentation: http://buildtest.rtfd.io

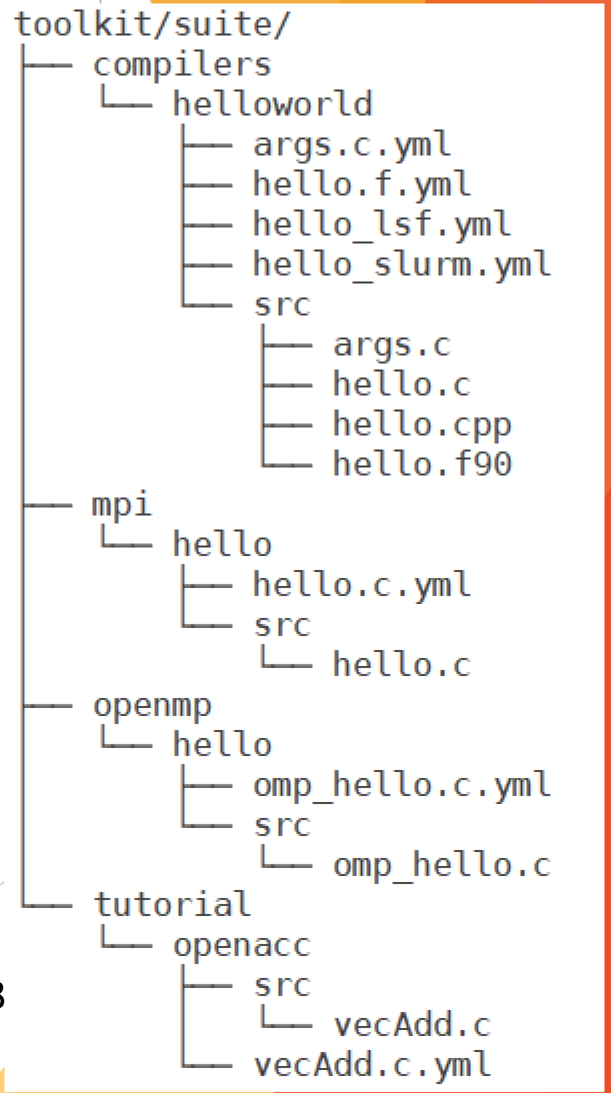
Build Pipeline



GitHub: https://github.com/HPC-buildtest/buildtest-framework
Documentation: http://buildtest.rtfd.io

Building a Test

- To build a test script just specify a test configuration to buildtest as follows: buildtest build –c <test-configuration>
- The test configuration can be found under $BUILDTEST_ROOT/toolkit/suite
- Name of test configuration is formulated by replacing file separator (/) by a dot (.) so compilers/hellworld/args.c.yml → compilers.helloworld.args.c.yml
- Source code must be under src directory and test configuration must be named with extension .yml



```
toolkit/suite/
├── compilers
│   └── helloworld
│       ├── args.c.yml
│       ├── hello.f.yml
│       ├── hello_lsf.yml
│       ├── hello_slurm.yml
│       └── src
│           ├── args.c
│           ├── hello.c
│           ├── hello.cpp
│           └── hello.f90
├── mpi
│   └── hello
│       ├── hello.c.yml
│       └── src
│           └── hello.c
├── openmp
│   └── hello
│       ├── omp_hello.c.yml
│       └── src
│           └── omp_hello.c
└── tutorial
    └── openacc
        ├── src
        │   └── vecAdd.c
        └── vecAdd.c.yml
```

GitHub: https://github.com/HPC-buildtest/buildtest-framework
Documentation: http://buildtest.rtfd.io

Test Configuration

```
testtype: singlesource
description: "C program that prints arguments passed to executable."
scheduler: local


program:
  compiler: gnu
  source: args.c
  env:
    FOO: BAR
    X: 1
  pre_build: gcc --version
  cflags: -Wall -g
  post_build: gcc -v
  pre_run: echo $SRCDIR $TESTDIR
  exec_opts: hello world!
  post_run: echo post_run

maintainer:
- shahzeb siddiqui shahzebmsiddiqui@gmail.com
```

Informs buildtest this is a Single Source Compilation. Implemented as a Python Class

Description of text. Limited to 80 chars

Run Test Locally

Start of Test Declaration

Specify Compiler Name

Source File to be compiled

Start of Environment Variable Declaration

Commands to run before and after compilation.

Passing flags to C compiler by setting CFLAGS variable

Commands to run before and after execution.

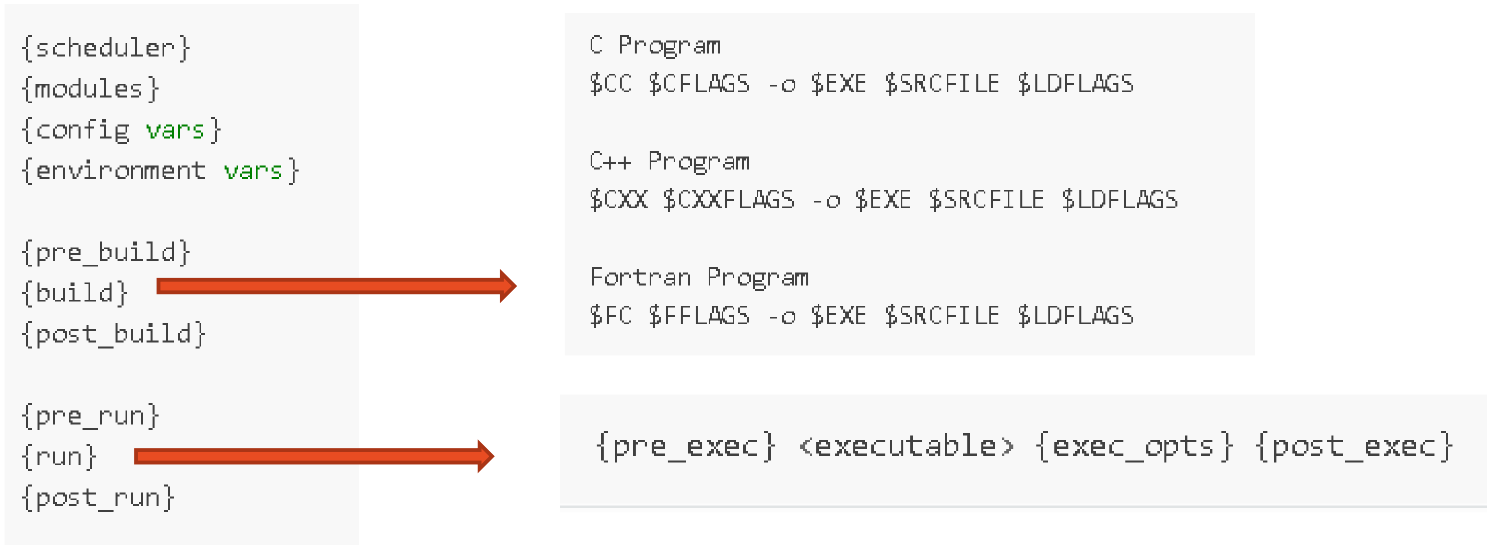
Passing Arguments to the Execution

List of Maintainers

GitHub: https://github.com/HPC-buildtest/buildtest-framework
Documentation: http://buildtest.rtfd.io

Test Configuration

- For Single Source compilation (C, C++, Fortran, CUDA) and MPI code the general structure of the test will be as follows:



```
{scheduler}
{modules}
{config vars}
{environment vars}

{pre_build}
{build}
{post_build}

{pre_run}
{run}
{post_run}
```

```
C Program
$CC $CFLAGS -o $EXE $SRCFILE $LDFLAGS

C++ Program
$CXX $CXXFLAGS -o $EXE $SRCFILE $LDFLAGS

Fortran Program
$FC $FFLAGS -o $EXE $SRCFILE $LDFLAGS
```

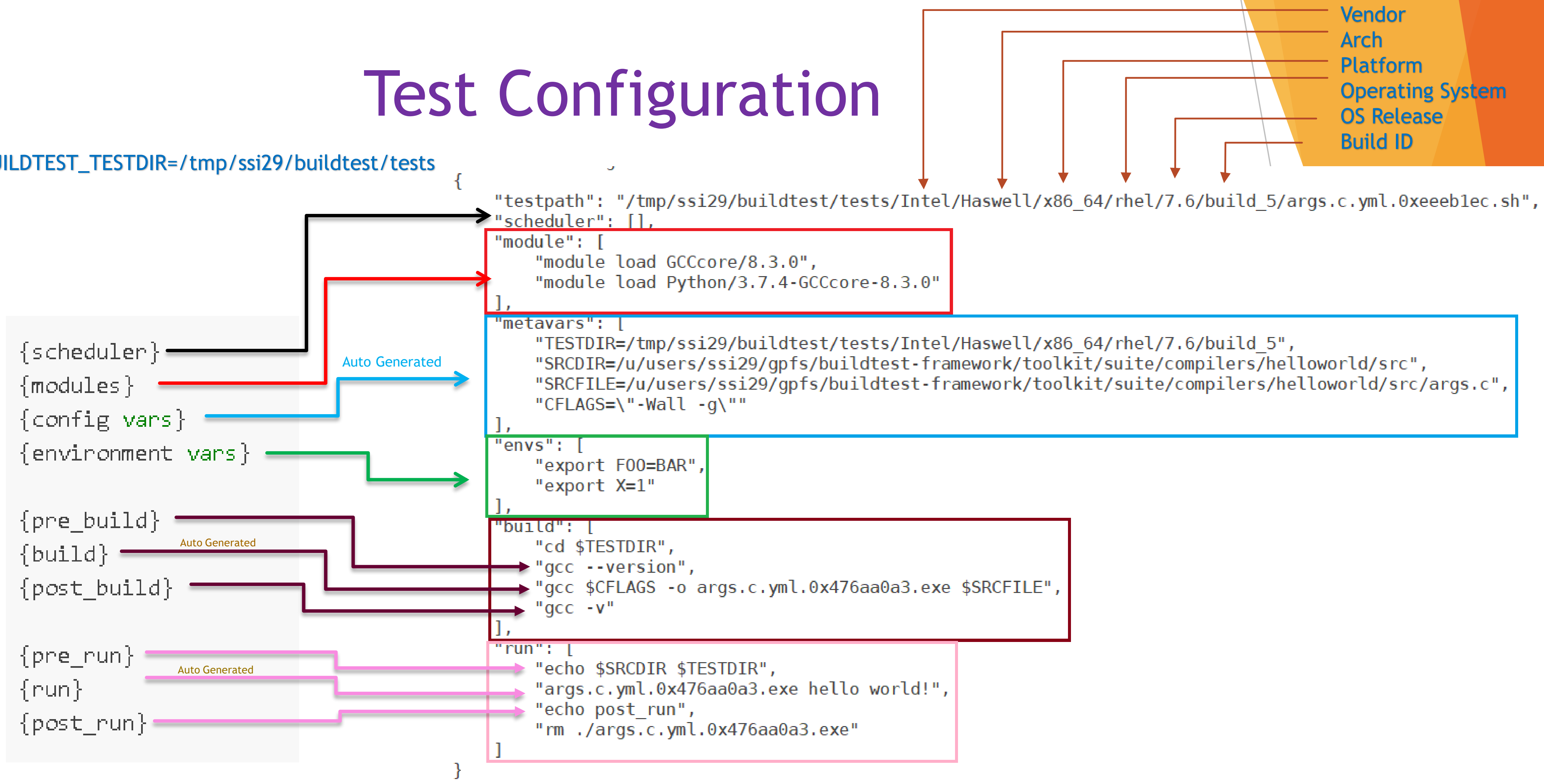
```
{pre_exec} <executable> {exec_opts} {post_exec}
```

- Buildtest will auto create the following sections: {config vars} {build} and {run}
- {pre_build}, {post_build}, {pre_run}, {post_run} are sections where shell commands can be injected into test script
- {module} section is used for loading modules that can be one of the following: active modules, user collection, buildtest module collection, or module permutation.
- {scheduler} section will be generated only if **scheduler: LSF** or **scheduler: SLURM** is set in configuration file.

GitHub: https://github.com/HPC-buildtest/buildtest-framework
Documentation: http://buildtest.rtfd.io

Test Configuration

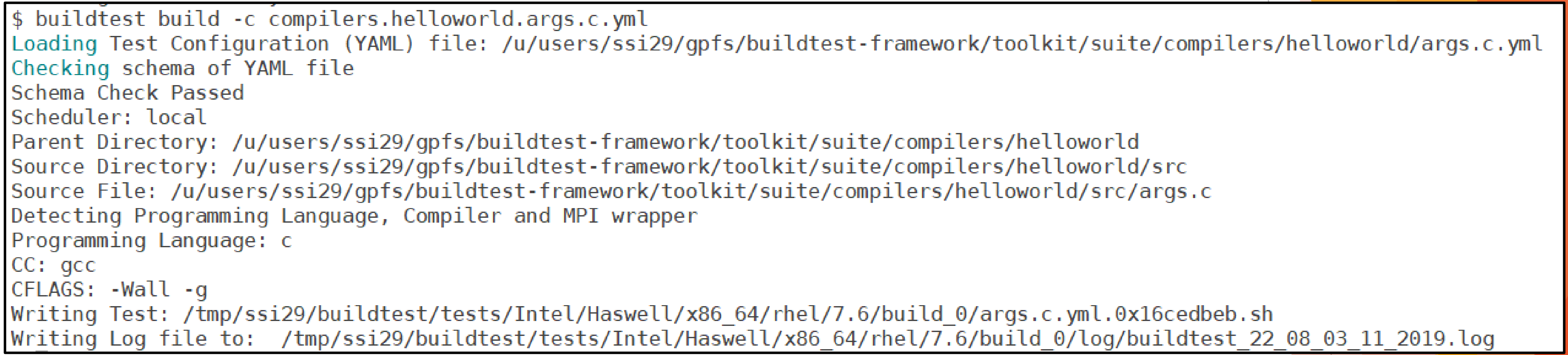
BUILDTEST_TESTDIR=/tmp/ssi29/buildtest/tests



```
{
    "testpath": "/tmp/ssi29/buildtest/tests/Intel/Haswell/x86_64/rhel/7.6/build_5/args.c.yml.0xeeeb1ec.sh",
    "scheduler": [],
    "module": [
        "module load GCCcore/8.3.0",
        "module load Python/3.7.4-GCCcore-8.3.0"
    ],
    "metavars": [
        "TESTDIR=/tmp/ssi29/buildtest/tests/Intel/Haswell/x86_64/rhel/7.6/build_5",
        "SRCDIR=/u/users/ssi29/gpfs/buildtest-framework/toolkit/suite/compilers/helloworld/src",
        "SRCFILE=/u/users/ssi29/gpfs/buildtest-framework/toolkit/suite/compilers/helloworld/src/args.c",
        "CFLAGS=\"-Wall -g\""
    ],
    "envs": [
        "export FOO=BAR",
        "export X=1"
    ],
    "build": [
        "cd $TESTDIR",
        "gcc --version",
        "gcc $CFLAGS -o args.c.yml.0x476aa0a3.exe $SRCFILE",
        "gcc -v"
    ],
    "run": [
        "echo $SRCDIR $TESTDIR",
        "args.c.yml.0x476aa0a3.exe hello world!",
        "echo post_run",
        "rm ./args.c.yml.0x476aa0a3.exe"
    ]
}
```

GitHub: https://github.com/HPC-buildtest/buildtest-framework
Documentation: http://buildtest.rtfd.io

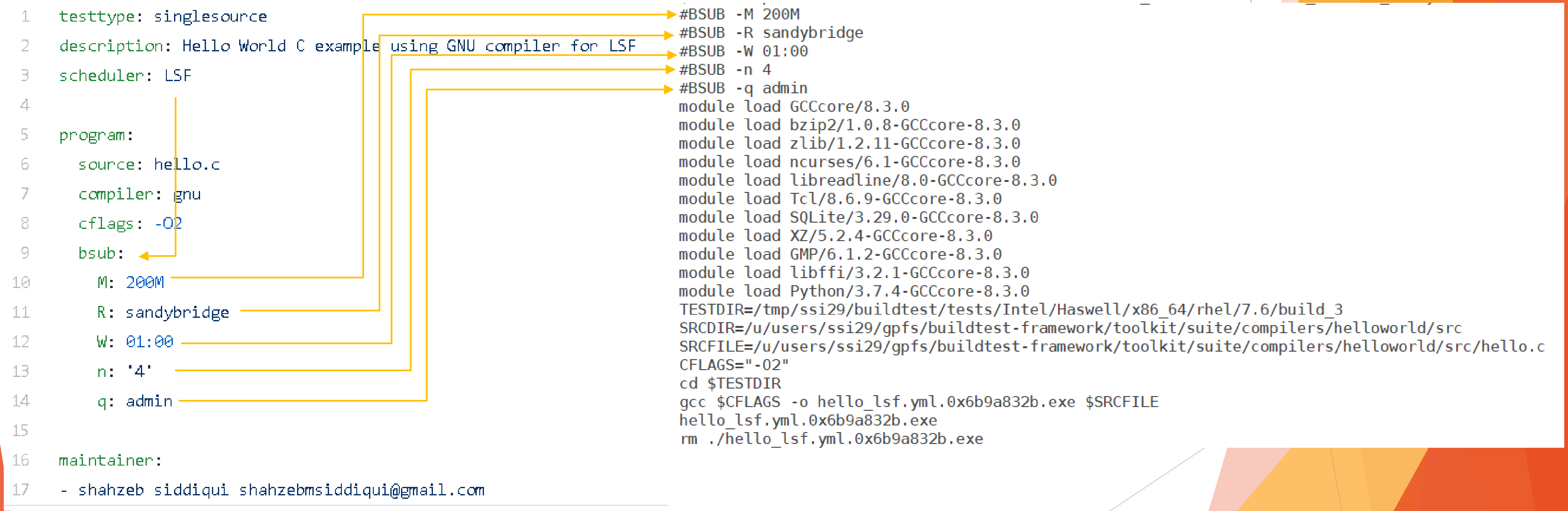
Example Build



```
$ buildtest build -c compilers.helloworld.args.c.yml
Loading Test Configuration (YAML) file: /u/users/ssi29/gpfs/buildtest-framework/toolkit/suite/compilers/helloworld/args.c.yml
Checking schema of YAML file
Schema Check Passed
Scheduler: local
Parent Directory: /u/users/ssi29/gpfs/buildtest-framework/toolkit/suite/compilers/helloworld
Source Directory: /u/users/ssi29/gpfs/buildtest-framework/toolkit/suite/compilers/helloworld/src
Source File: /u/users/ssi29/gpfs/buildtest-framework/toolkit/suite/compilers/helloworld/src/args.c
Detecting Programming Language, Compiler and MPI wrapper
Programming Language: c
CC: gcc
CFLAGS: -Wall -g
Writing Test: /tmp/ssi29/buildtest/tests/Intel/Haswell/x86_64/rhel/7.6/build_0/args.c.yml.0x16cedbeb.sh
Writing Log file to:  /tmp/ssi29/buildtest/tests/Intel/Haswell/x86_64/rhel/7.6/build_0/log/buildtest_22_08_03_11_2019.log
```

GitHub: https://github.com/HPC-buildtest/buildtest-framework
Documentation: http://buildtest.rtfd.io

LSF Test

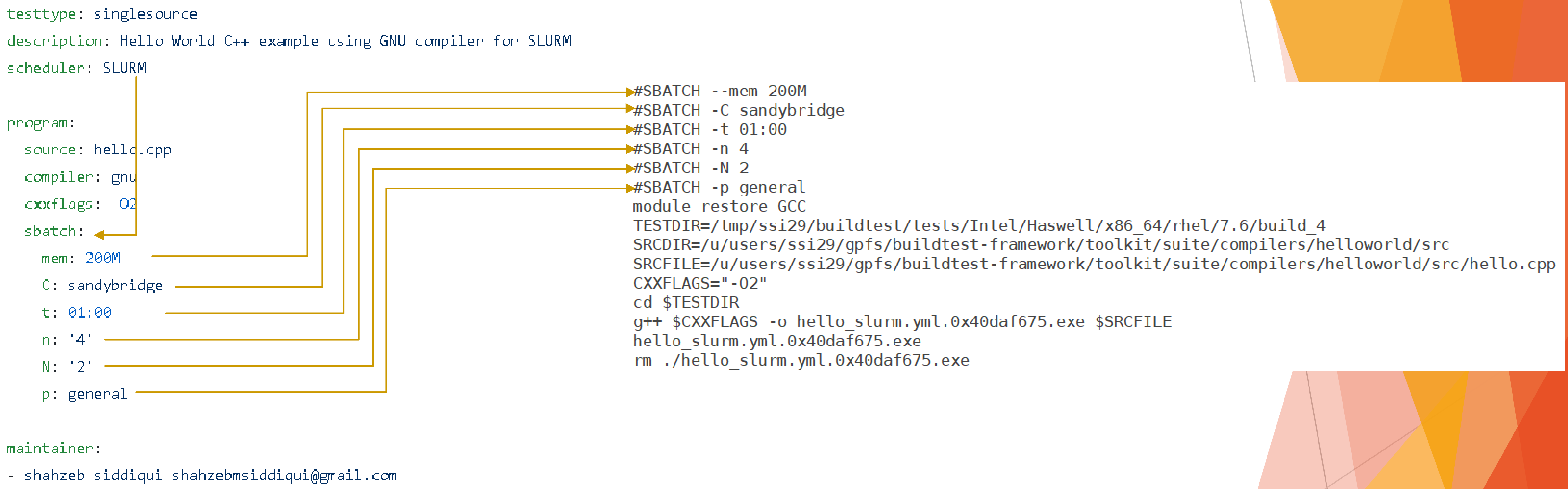


```
testtype: singlesource
description: Hello World C example using GNU compiler for LSF
scheduler: LSF

program:
  source: hello.c
  compiler: gnu
  cflags: -O2
  bsub:
    M: 200M
    R: sandybridge
    W: 01:00
    n: '4'
    q: admin

maintainer:
- shahzeb siddiqui shahzebmsiddiqui@gmail.com
```

```
#BSUB -M 200M
#BSUB -R sandybridge
#BSUB -W 01:00
#BSUB -n 4
#BSUB -q admin
module load GCCcore/8.3.0
module load bzip2/1.0.8-GCCcore-8.3.0
module load zlib/1.2.11-GCCcore-8.3.0
module load ncurses/6.1-GCCcore-8.3.0
module load libreadline/8.0-GCCcore-8.3.0
module load Tcl/8.6.9-GCCcore-8.3.0
module load SQLite/3.29.0-GCCcore-8.3.0
module load XZ/5.2.4-GCCcore-8.3.0
module load GMP/6.1.2-GCCcore-8.3.0
module load libffi/3.2.1-GCCcore-8.3.0
module load Python/3.7.4-GCCcore-8.3.0
TESTDIR=/tmp/ssi29/buildtest/tests/Intel/Haswell/x86_64/rhel/7.6/build_3
SRCDIR=/u/users/ssi29/gpfs/buildtest-framework/toolkit/suite/compilers/helloworld/src
SRCFILE=/u/users/ssi29/gpfs/buildtest-framework/toolkit/suite/compilers/helloworld/src/hello.c
CFLAGS="-O2"
cd $TESTDIR
gcc $CFLAGS -o hello_lsf.yml.0x6b9a832b.exe $SRCFILE
hello_lsf.yml.0x6b9a832b.exe
rm ./hello_lsf.yml.0x6b9a832b.exe
```

GitHub: https://github.com/HPC-buildtest/buildtest-framework
Documentation: http://buildtest.rtfd.io

SLURM Test



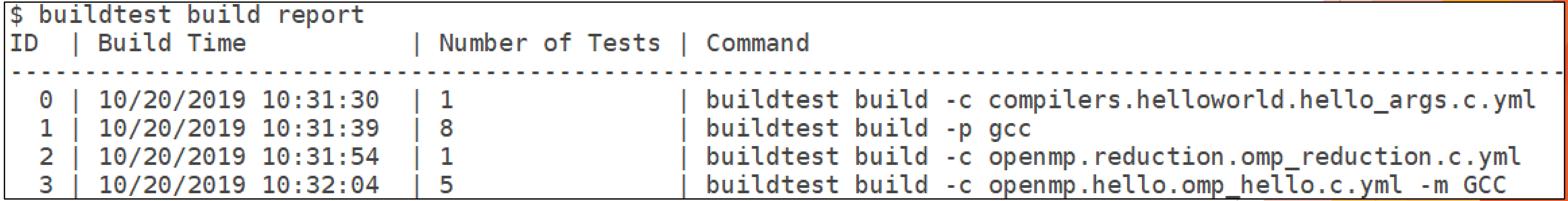
```
testtype: singlesource
description: Hello World C++ example using GNU compiler for SLURM
scheduler: SLURM

program:
  source: hello.cpp
  compiler: gnu
  cxxflags: -O2
  sbatch:
    mem: 200M
    C: sandybridge
    t: 01:00
    n: '4'
    N: '2'
    p: general

maintainer:
- shahzeb siddiqui shahzebmsiddiqui@gmail.com
```

```
#SBATCH --mem 200M
#SBATCH -C sandybridge
#SBATCH -t 01:00
#SBATCH -n 4
#SBATCH -N 2
#SBATCH -p general
module restore GCC
TESTDIR=/tmp/ssi29/buildtest/tests/Intel/Haswell/x86_64/rhel/7.6/build_4
SRCDIR=/u/users/ssi29/gpfs/buildtest-framework/toolkit/suite/compilers/helloworld/src
SRCFILE=/u/users/ssi29/gpfs/buildtest-framework/toolkit/suite/compilers/helloworld/src/hello.cpp
CXXFLAGS="-O2"
cd $TESTDIR
g++ $CXXFLAGS -o hello_slurm.yml.0x40daf675.exe $SRCFILE
hello_slurm.yml.0x40daf675.exe
rm ./hello_slurm.yml.0x40daf675.exe
```

GitHub: https://github.com/HPC-buildtest/buildtest-framework
Documentation: http://buildtest.rtfd.io

Build History

- Buildtest keeps track of every build in a json file (build.json). The **build ID** that can be used to retrieve tests, logs, and run tests
- To retrieve a report of all builds: **buildtest build report**
- Retrieve Logs for build ID 3: **buildtest build log 3**
- Retrieve test scripts for build ID 3: **buildtest build tests 3**
- Run tests for build ID 3: **buildtest build run 3**

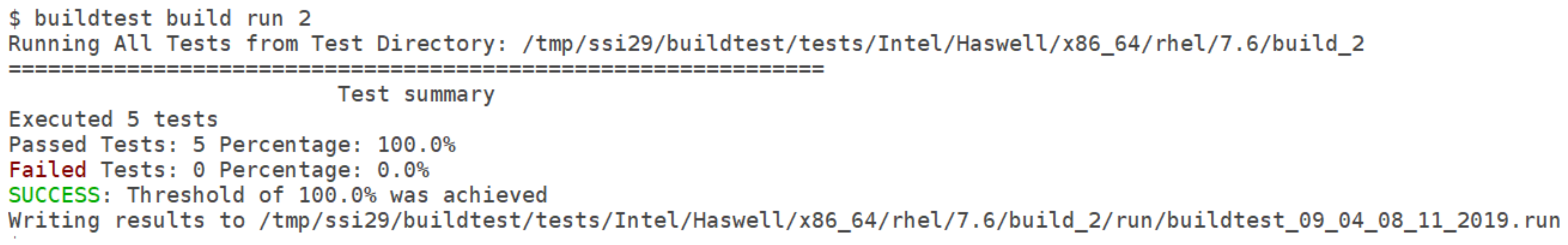


```
$ buildtest build report
ID   | Build Time            | Number of Tests | Command
-----------------------------------------------------------------------------------------------------------
   0 | 10/20/2019 10:31:30   | 1               | buildtest build -c compilers.helloworld.hello_args.c.yml
   1 | 10/20/2019 10:31:39   | 8               | buildtest build -p gcc
   2 | 10/20/2019 10:31:54   | 1               | buildtest build -c openmp.reduction.omp_reduction.c.yml
   3 | 10/20/2019 10:32:04   | 5               | buildtest build -c openmp.hello.omp_hello.c.yml -m GCC
```

GitHub: https://github.com/HPC-buildtest/buildtest-framework
Documentation: http://buildtest.rtfd.io

Running Test Locally

- Running **buildtest build run <ID>** will run all testscripts that corresponds to the build ID.
- Buildtest will write a .run file that contains output of all tests
- A zero exit status will be a PASSED test and non-zero will be a FAILED test

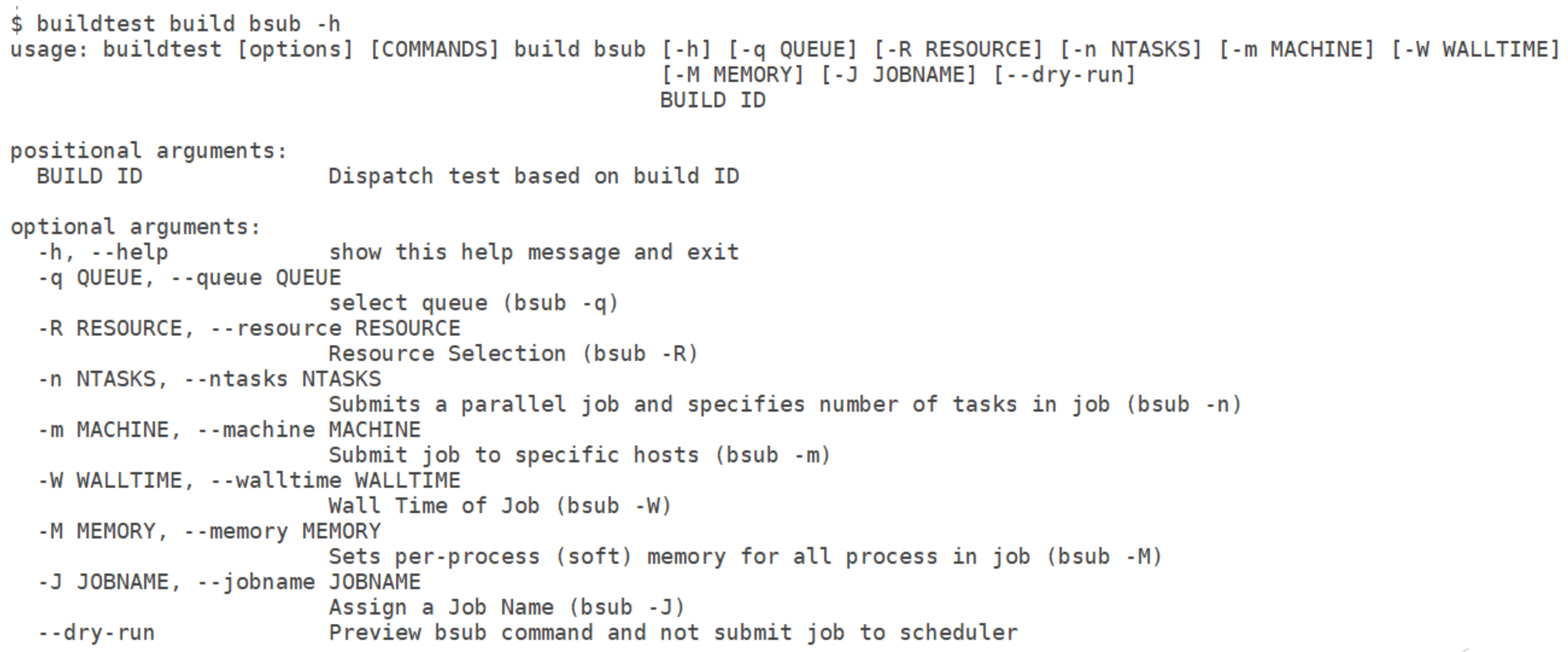


```
$ buildtest build run 2
Running All Tests from Test Directory: /tmp/ssi29/buildtest/tests/Intel/Haswell/x86_64/rhel/7.6/build_2
==============================================================
                         Test summary
Executed 5 tests
Passed Tests: 5 Percentage: 100.0%
Failed Tests: 0 Percentage: 0.0%
SUCCESS: Threshold of 100.0% was achieved
Writing results to /tmp/ssi29/buildtest/tests/Intel/Haswell/x86_64/rhel/7.6/build_2/run/buildtest_09_04_08_11_2019.run
```

GitHub: https://github.com/HPC-buildtest/buildtest-framework
Documentation: http://buildtest.rtfd.io

Submit Jobs via bsub (Experimental Feature)

- Buildtest provides CLI to run any build ID via bsub wrapper regardless if you have specified any bsub parameters in the test configuration.
- Currently, the following options are available for bsub

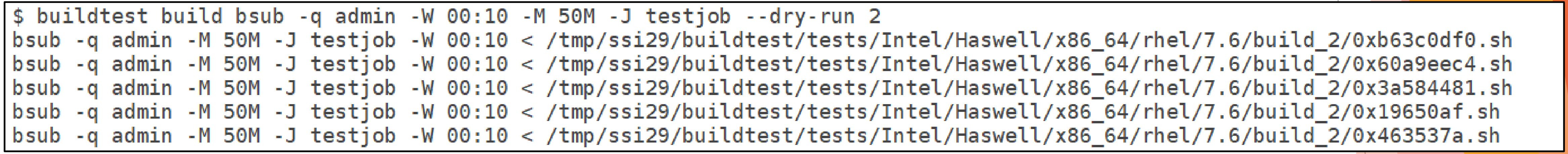


```
$ buildtest build bsub -h
usage: buildtest [options] [COMMANDS] build bsub [-h] [-q QUEUE] [-R RESOURCE] [-n NTASKS] [-m MACHINE] [-W WALLTIME]
                                                 [-M MEMORY] [-J JOBNAME] [--dry-run]
                                                 BUILD ID

positional arguments:
  BUILD ID              Dispatch test based on build ID

optional arguments:
  -h, --help            show this help message and exit
  -q QUEUE, --queue QUEUE
                        select queue (bsub -q)
  -R RESOURCE, --resource RESOURCE
                        Resource Selection (bsub -R)
  -n NTASKS, --ntasks NTASKS
                        Submits a parallel job and specifies number of tasks in job (bsub -n)
  -m MACHINE, --machine MACHINE
                        Submit job to specific hosts (bsub -m)
  -W WALLTIME, --walltime WALLTIME
                        Wall Time of Job (bsub -W)
  -M MEMORY, --memory MEMORY
                        Sets per-process (soft) memory for all process in job (bsub -M)
  -J JOBNAME, --jobname JOBNAME
                        Assign a Job Name (bsub -J)
  --dry-run             Preview bsub command and not submit job to scheduler
```

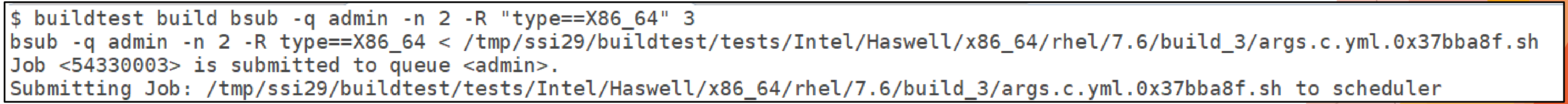
GitHub: https://github.com/HPC-buildtest/buildtest-framework
Documentation: http://buildtest.rtfd.io

Submitting Jobs via bsub

- The --dry-run option will let you see the bsub command without actually running the command.
- All bsub options are processed as string types in order for bsub command to handle complex commands



```
$ buildtest build bsub -q admin -W 00:10 -M 50M -J testjob --dry-run 2
bsub -q admin -M 50M -J testjob -W 00:10 < /tmp/ssi29/buildtest/tests/Intel/Haswell/x86_64/rhel/7.6/build_2/0xb63c0df0.sh
bsub -q admin -M 50M -J testjob -W 00:10 < /tmp/ssi29/buildtest/tests/Intel/Haswell/x86_64/rhel/7.6/build_2/0x60a9eec4.sh
bsub -q admin -M 50M -J testjob -W 00:10 < /tmp/ssi29/buildtest/tests/Intel/Haswell/x86_64/rhel/7.6/build_2/0x3a584481.sh
bsub -q admin -M 50M -J testjob -W 00:10 < /tmp/ssi29/buildtest/tests/Intel/Haswell/x86_64/rhel/7.6/build_2/0x19650af.sh
bsub -q admin -M 50M -J testjob -W 00:10 < /tmp/ssi29/buildtest/tests/Intel/Haswell/x86_64/rhel/7.6/build_2/0x463537a.sh
```



```
$ buildtest build bsub -q admin -n 2 -R "type==X86_64" 3
bsub -q admin -n 2 -R type==X86_64 < /tmp/ssi29/buildtest/tests/Intel/Haswell/x86_64/rhel/7.6/build_3/args.c.yml.0x37bba8f.sh
Job <54330003> is submitted to queue <admin>.
Submitting Job: /tmp/ssi29/buildtest/tests/Intel/Haswell/x86_64/rhel/7.6/build_3/args.c.yml.0x37bba8f.sh to scheduler
```

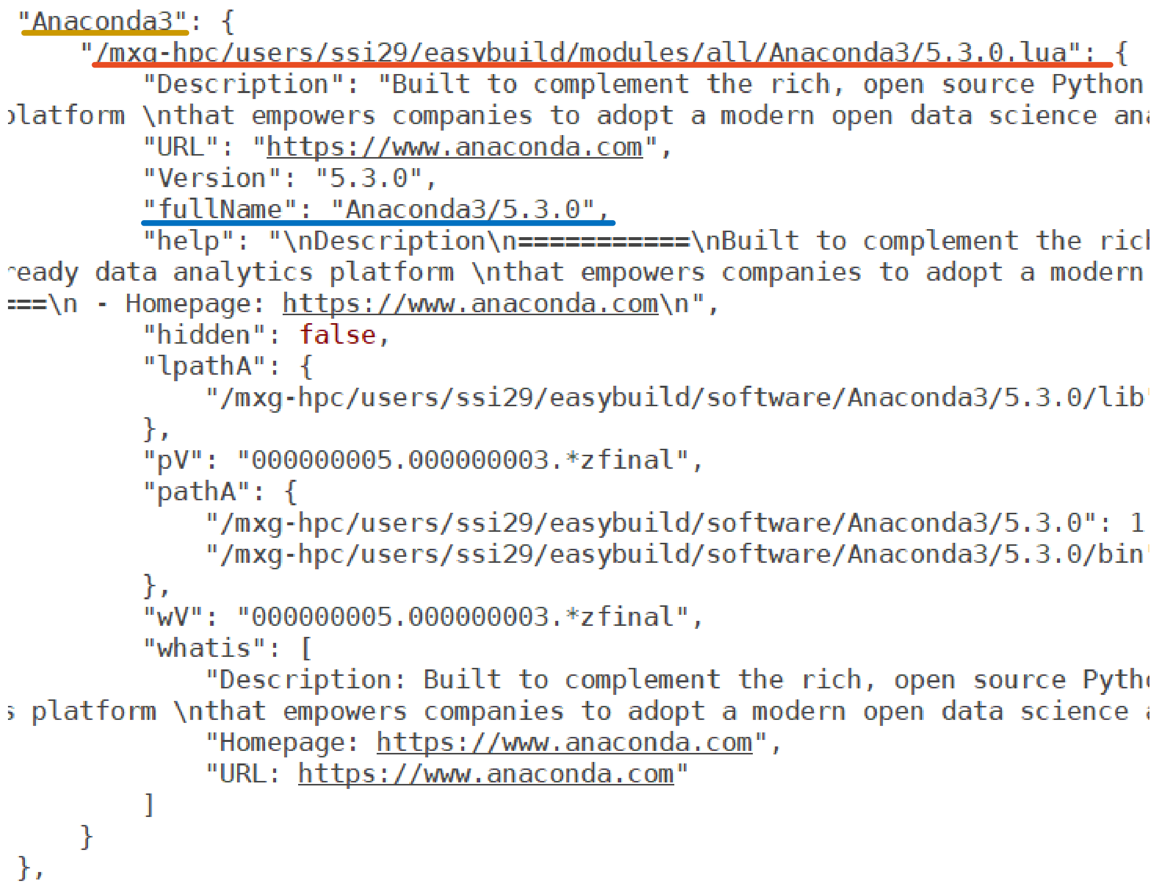
GitHub: https://github.com/HPC-buildtest/buildtest-framework
Documentation: http://buildtest.rtfd.io

Integration with Spider

- Buildtest solves the module load problem by parsing json content of the following command: `spider –o spider-json $BUILDTEST_MODULEPATH`
- Buildtest leverages spider to load modules into test.
- Spider is automatically updated when MODULEPATH changes!
- In addition, spider has allowed buildtest to create new module utilities useful for Software Stack Administrators
- For more details refer to the following links:
  https://lmod.readthedocs.io/en/latest/136_spider.html
  https://buildtest.readthedocs.io/en/devel/concepts.html

GitHub: https://github.com/HPC-buildtest/buildtest-framework
Documentation: http://buildtest.rtfd.io

Spider Content



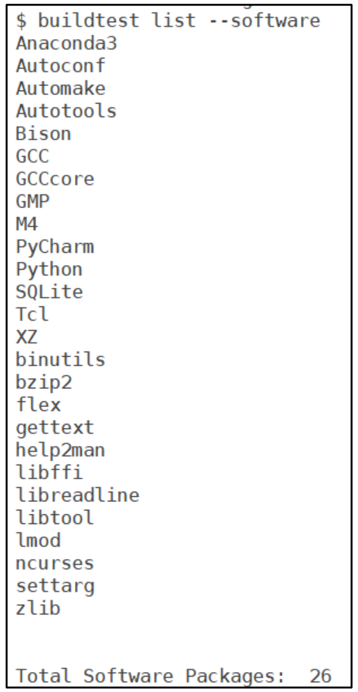
```
 "Anaconda3": {
     "/mxg-hpc/users/ssi29/easybuild/modules/all/Anaconda3/5.3.0.lua": {
         "Description": "Built to complement the rich, open source Python
latform \nthat empowers companies to adopt a modern open data science ana
         "URL": "https://www.anaconda.com",
         "Version": "5.3.0",
         "fullName": "Anaconda3/5.3.0",
         "help": "\nDescription\n===========\nBuilt to complement the ricl
eady data analytics platform \nthat empowers companies to adopt a modern
===\n - Homepage: https://www.anaconda.com\n",
         "hidden": false,
         "lpathA": {
             "/mxg-hpc/users/ssi29/easybuild/software/Anaconda3/5.3.0/lib
         },
         "pV": "000000005.000000003.*zfinal",
         "pathA": {
             "/mxg-hpc/users/ssi29/easybuild/software/Anaconda3/5.3.0": 1
             "/mxg-hpc/users/ssi29/easybuild/software/Anaconda3/5.3.0/bin
         },
         "wV": "000000005.000000003.*zfinal",
         "whatis": [
             "Description: Built to complement the rich, open source Pyth
s platform \nthat empowers companies to adopt a modern open data science
             "Homepage: https://www.anaconda.com",
             "URL: https://www.anaconda.com"
         ]
     }
 },
```

GitHub: https://github.com/HPC-buildtest/buildtest-framework
Documentation: http://buildtest.rtfd.io

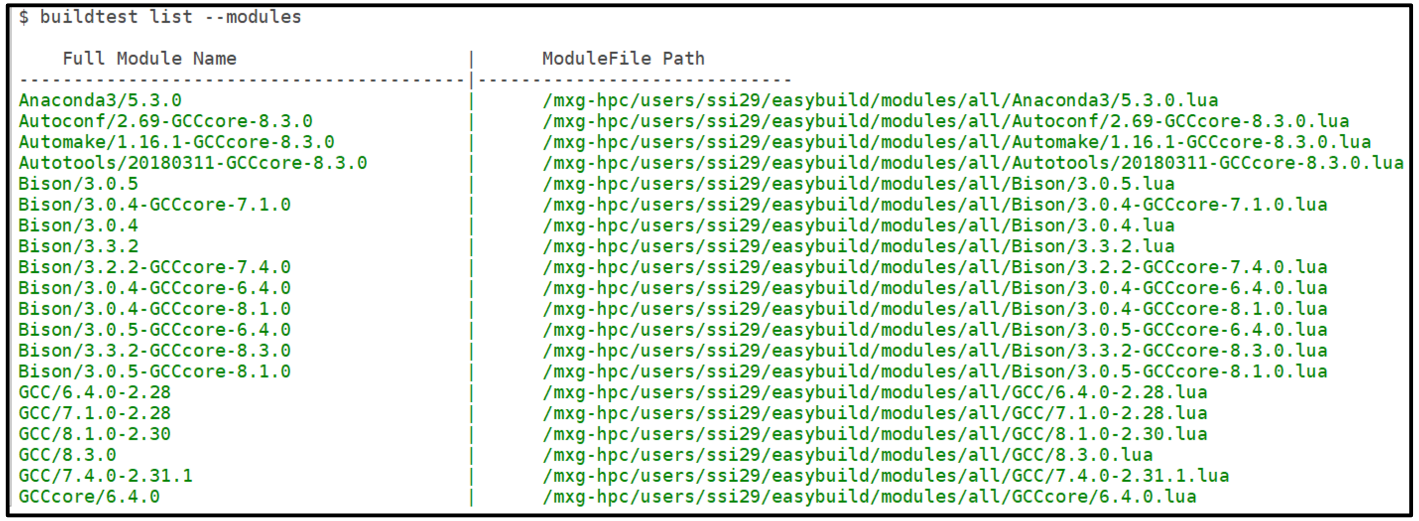
Software Stack Operations

List of Installed Software and Modules

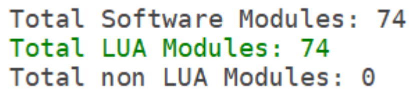
- Problem: Retrieve a list of unique software and modules installed in the cluster
- Use Case: Automatically keep an updated list of software as part of end-user documentation.



```
$ buildtest list --software
Anaconda3
Autoconf
Automake
Autotools
Bison
GCC
GCCcore
GMP
M4
PyCharm
Python
SQLite
Tcl
XZ
binutils
bzip2
flex
gettext
help2man
libffi
libreadline
libtool
lmod
ncurses
settarg
zlib


Total Software Packages:  26
```



```
$ buildtest list --modules

    Full Module Name                        |      ModuleFile Path
-----------------------------------------|-----------------------------
Anaconda3/5.3.0                           |      /mxg-hpc/users/ssi29/easybuild/modules/all/Anaconda3/5.3.0.lua
Autoconf/2.69-GCCcore-8.3.0               |      /mxg-hpc/users/ssi29/easybuild/modules/all/Autoconf/2.69-GCCcore-8.3.0.lua
Automake/1.16.1-GCCcore-8.3.0             |      /mxg-hpc/users/ssi29/easybuild/modules/all/Automake/1.16.1-GCCcore-8.3.0.lua
Autotools/20180311-GCCcore-8.3.0          |      /mxg-hpc/users/ssi29/easybuild/modules/all/Autotools/20180311-GCCcore-8.3.0.lua
Bison/3.0.5                               |      /mxg-hpc/users/ssi29/easybuild/modules/all/Bison/3.0.5.lua
Bison/3.0.4-GCCcore-7.1.0                 |      /mxg-hpc/users/ssi29/easybuild/modules/all/Bison/3.0.4-GCCcore-7.1.0.lua
Bison/3.0.4                               |      /mxg-hpc/users/ssi29/easybuild/modules/all/Bison/3.0.4.lua
Bison/3.3.2                               |      /mxg-hpc/users/ssi29/easybuild/modules/all/Bison/3.3.2.lua
Bison/3.2.2-GCCcore-7.4.0                 |      /mxg-hpc/users/ssi29/easybuild/modules/all/Bison/3.2.2-GCCcore-7.4.0.lua
Bison/3.0.4-GCCcore-6.4.0                 |      /mxg-hpc/users/ssi29/easybuild/modules/all/Bison/3.0.4-GCCcore-6.4.0.lua
Bison/3.0.4-GCCcore-8.1.0                 |      /mxg-hpc/users/ssi29/easybuild/modules/all/Bison/3.0.4-GCCcore-8.1.0.lua
Bison/3.0.5-GCCcore-6.4.0                 |      /mxg-hpc/users/ssi29/easybuild/modules/all/Bison/3.0.5-GCCcore-6.4.0.lua
Bison/3.3.2-GCCcore-8.3.0                 |      /mxg-hpc/users/ssi29/easybuild/modules/all/Bison/3.3.2-GCCcore-8.3.0.lua
Bison/3.0.5-GCCcore-8.1.0                 |      /mxg-hpc/users/ssi29/easybuild/modules/all/Bison/3.0.5-GCCcore-8.1.0.lua
GCC/6.4.0-2.28                            |      /mxg-hpc/users/ssi29/easybuild/modules/all/GCC/6.4.0-2.28.lua
GCC/7.1.0-2.28                            |      /mxg-hpc/users/ssi29/easybuild/modules/all/GCC/7.1.0-2.28.lua
GCC/8.1.0-2.30                            |      /mxg-hpc/users/ssi29/easybuild/modules/all/GCC/8.1.0-2.30.lua
GCC/8.3.0                                 |      /mxg-hpc/users/ssi29/easybuild/modules/all/GCC/8.3.0.lua
GCC/7.4.0-2.31.1                          |      /mxg-hpc/users/ssi29/easybuild/modules/all/GCC/7.4.0-2.31.1.lua
GCCcore/6.4.0                             |      /mxg-hpc/users/ssi29/easybuild/modules/all/GCCcore/6.4.0.lua
```

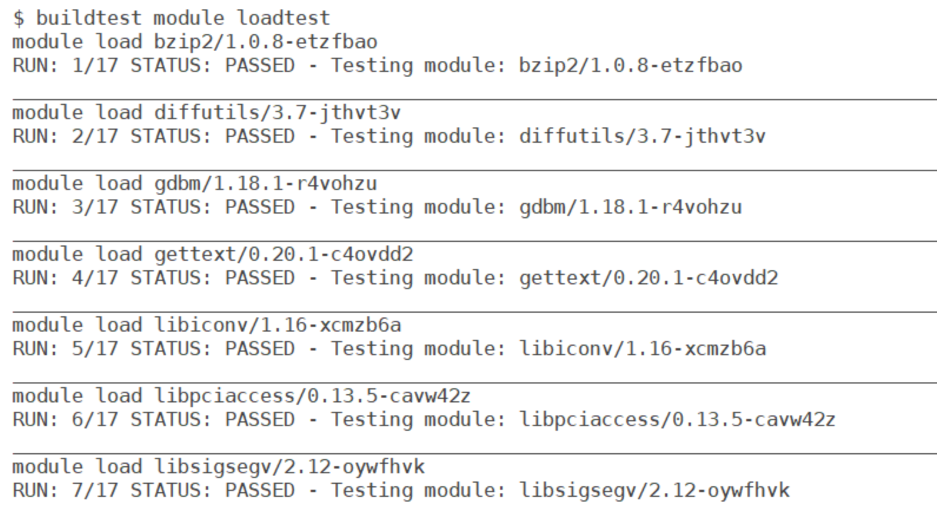


```
Total Software Modules: 74
Total LUA Modules: 74
Total non LUA Modules: 0
```

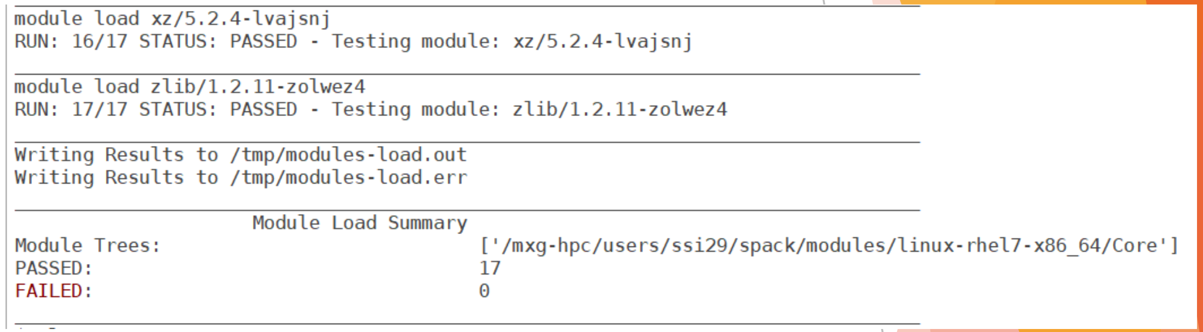
GitHub: https://github.com/HPC-buildtest/buildtest-framework
Documentation: http://buildtest.rtfd.io

Module Load Testing

- Problem: Verify all modules in a software stack



```
$ buildtest module loadtest
module load bzip2/1.0.8-etzfbao
RUN: 1/17 STATUS: PASSED - Testing module: bzip2/1.0.8-etzfbao
________________________________________________________________________________
module load diffutils/3.7-jthvt3v
RUN: 2/17 STATUS: PASSED - Testing module: diffutils/3.7-jthvt3v
________________________________________________________________________________
module load gdbm/1.18.1-r4vohzu
RUN: 3/17 STATUS: PASSED - Testing module: gdbm/1.18.1-r4vohzu
________________________________________________________________________________
module load gettext/0.20.1-c4ovdd2
RUN: 4/17 STATUS: PASSED - Testing module: gettext/0.20.1-c4ovdd2
________________________________________________________________________________
module load libiconv/1.16-xcmzb6a
RUN: 5/17 STATUS: PASSED - Testing module: libiconv/1.16-xcmzb6a
________________________________________________________________________________
module load libpciaccess/0.13.5-cavw42z
RUN: 6/17 STATUS: PASSED - Testing module: libpciaccess/0.13.5-cavw42z
________________________________________________________________________________
module load libsigsegv/2.12-oywfhvk
RUN: 7/17 STATUS: PASSED - Testing module: libsigsegv/2.12-oywfhvk
```



```
module load xz/5.2.4-lvajsnj
RUN: 16/17 STATUS: PASSED - Testing module: xz/5.2.4-lvajsnj
________________________________________________________________________________
module load zlib/1.2.11-zolwez4
RUN: 17/17 STATUS: PASSED - Testing module: zlib/1.2.11-zolwez4
________________________________________________________________________________
Writing Results to /tmp/modules-load.out
Writing Results to /tmp/modules-load.err
________________________________________________________________________________
                    Module Load Summary
Module Trees:                         ['/mxg-hpc/users/ssi29/spack/modules/linux-rhel7-x86_64/Core']
PASSED:                               17
FAILED:                               0
________________________________________________________________________________
```

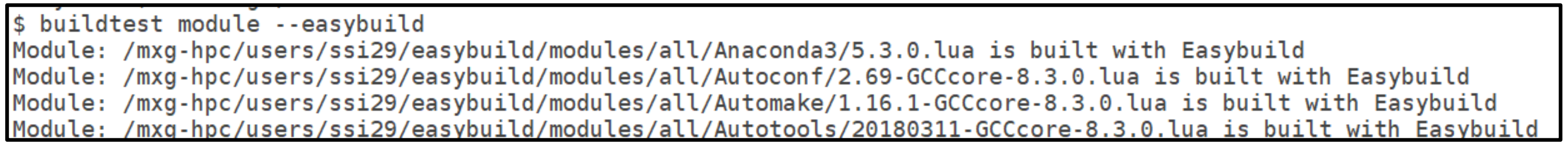
GitHub: https://github.com/HPC-buildtest/buildtest-framework
Documentation: http://buildtest.rtfd.io

Reporting Easybuild & Spack Modules

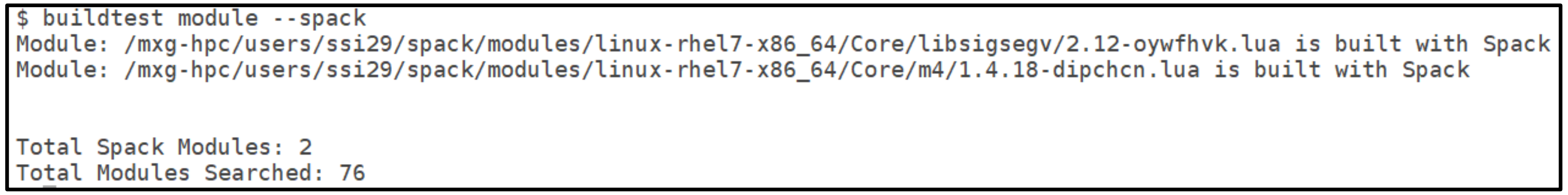
- Often times you want to know auto-generated modules (Easybuild, Spack) vs modules created manually.
- This can be done by searching for a unique string in module file embedded by both package managers

```
Built with EasyBuild version 3.7.1
```

```
Module file created by spack (https://github.com/spack/spack) on 2019-04-11 11:38:31.191604
```



```
$ buildtest module --easybuild
Module: /mxg-hpc/users/ssi29/easybuild/modules/all/Anaconda3/5.3.0.lua is built with Easybuild
Module: /mxg-hpc/users/ssi29/easybuild/modules/all/Autoconf/2.69-GCCcore-8.3.0.lua is built with Easybuild
Module: /mxg-hpc/users/ssi29/easybuild/modules/all/Automake/1.16.1-GCCcore-8.3.0.lua is built with Easybuild
Module: /mxg-hpc/users/ssi29/easybuild/modules/all/Autotools/20180311-GCCcore-8.3.0.lua is built with Easybuild
```

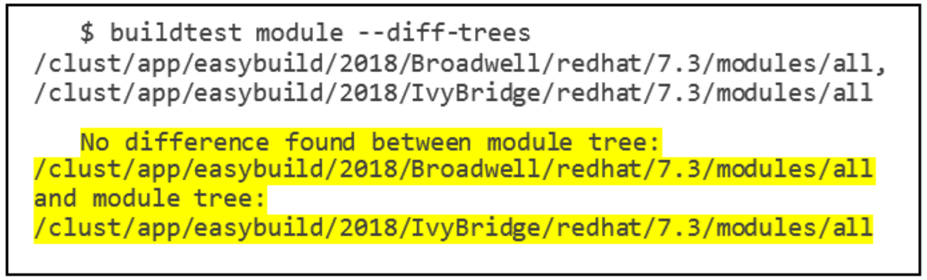


```
$ buildtest module --spack
Module: /mxg-hpc/users/ssi29/spack/modules/linux-rhel7-x86_64/Core/libsigsegv/2.12-oywfhvk.lua is built with Spack
Module: /mxg-hpc/users/ssi29/spack/modules/linux-rhel7-x86_64/Core/m4/1.4.18-dipchcn.lua is built with Spack


Total Spack Modules: 2
Total Modules Searched: 76
```

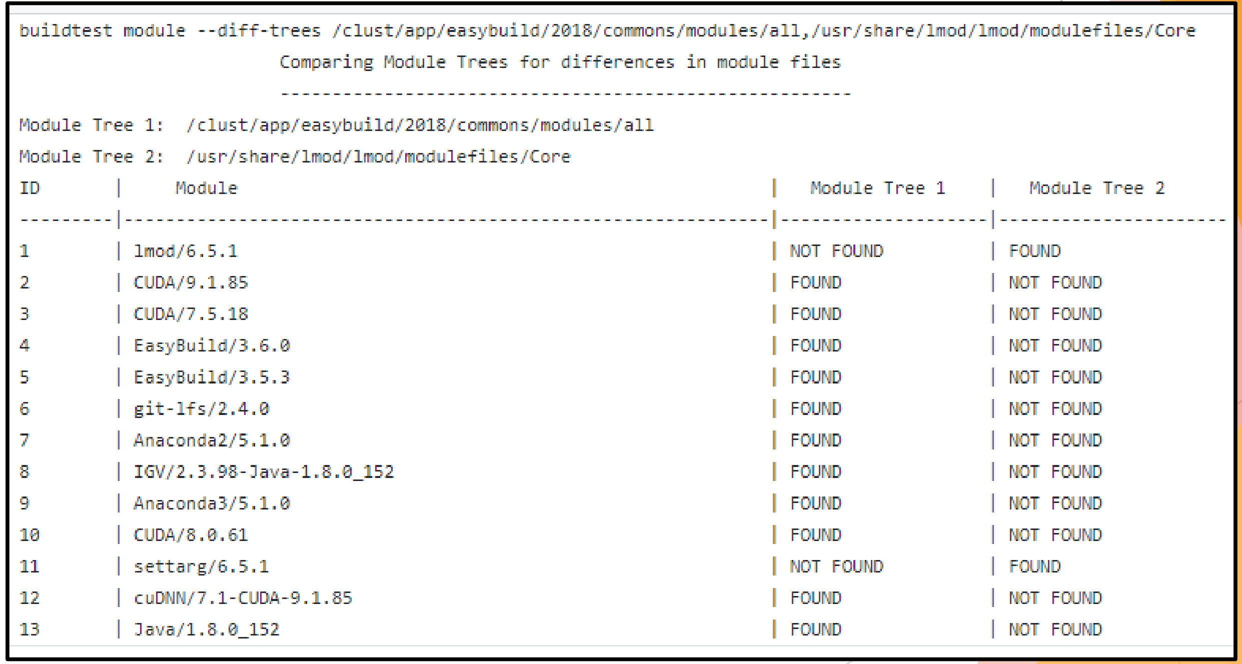
GitHub: https://github.com/HPC-buildtest/buildtest-framework
Documentation: http://buildtest.rtfd.io

Difference between Module Trees

- Problem: Building a Parallel Software Stack for each Architecture in a heterogeneous cluster and avoid asymmetries in modules between software stack.
- Solution: Difference between two module trees by Module Full Name



```
   $ buildtest module --diff-trees
/clust/app/easybuild/2018/Broadwell/redhat/7.3/modules/all,
/clust/app/easybuild/2018/IvyBridge/redhat/7.3/modules/all

   No difference found between module tree:
/clust/app/easybuild/2018/Broadwell/redhat/7.3/modules/all
and module tree:
/clust/app/easybuild/2018/IvyBridge/redhat/7.3/modules/all
```



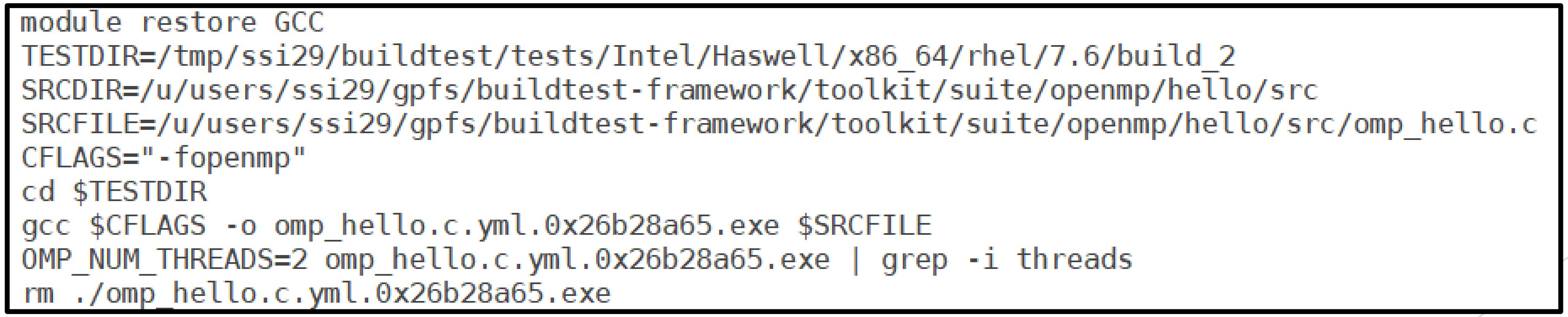
```
buildtest module --diff-trees /clust/app/easybuild/2018/commons/modules/all,/usr/share/lmod/lmod/modulefiles/Core
                              Comparing Module Trees for differences in module files
                              -------------------------------------------------------------

Module Tree 1:  /clust/app/easybuild/2018/commons/modules/all
Module Tree 2:  /usr/share/lmod/lmod/modulefiles/Core
ID        |     Module                                                      |  Module Tree 1    |  Module Tree 2
---------|--------------------------------------------------------------|------------------|---------------------
1         | lmod/6.5.1                                                      | NOT FOUND         | FOUND
2         | CUDA/9.1.85                                                     | FOUND             | NOT FOUND
3         | CUDA/7.5.18                                                     | FOUND             | NOT FOUND
4         | EasyBuild/3.6.0                                                 | FOUND             | NOT FOUND
5         | EasyBuild/3.5.3                                                 | FOUND             | NOT FOUND
6         | git-lfs/2.4.0                                                   | FOUND             | NOT FOUND
7         | Anaconda2/5.1.0                                                 | FOUND             | NOT FOUND
8         | IGV/2.3.98-Java-1.8.0_152                                       | FOUND             | NOT FOUND
9         | Anaconda3/5.1.0                                                 | FOUND             | NOT FOUND
10        | CUDA/8.0.61                                                     | FOUND             | NOT FOUND
11        | settarg/6.5.1                                                   | NOT FOUND         | FOUND
12        | cuDNN/7.1-CUDA-9.1.85                                           | FOUND             | NOT FOUND
13        | Java/1.8.0_152                                                  | FOUND             | NOT FOUND
```

GitHub: https://github.com/HPC-buildtest/buildtest-framework
Documentation: http://buildtest.rtfd.io

Building with Lmod User Collection

1. Load the modules of interest
2. Save the modules in a user collection
3. Build the test by referencing the user collection

```
$ module list

Currently Loaded Modules:
  1) GCCcore/8.3.0   2) zlib/1.2.11-GCCcore-8.3.0
3) binutils/2.32-GCCcore-8.3.0   4) GCC/8.3.0

$ module save GCC
Saved current collection of modules to: "GCC"

$ buildtest build –c openmp.hello.omp_hello.c.yml –co GCC
```

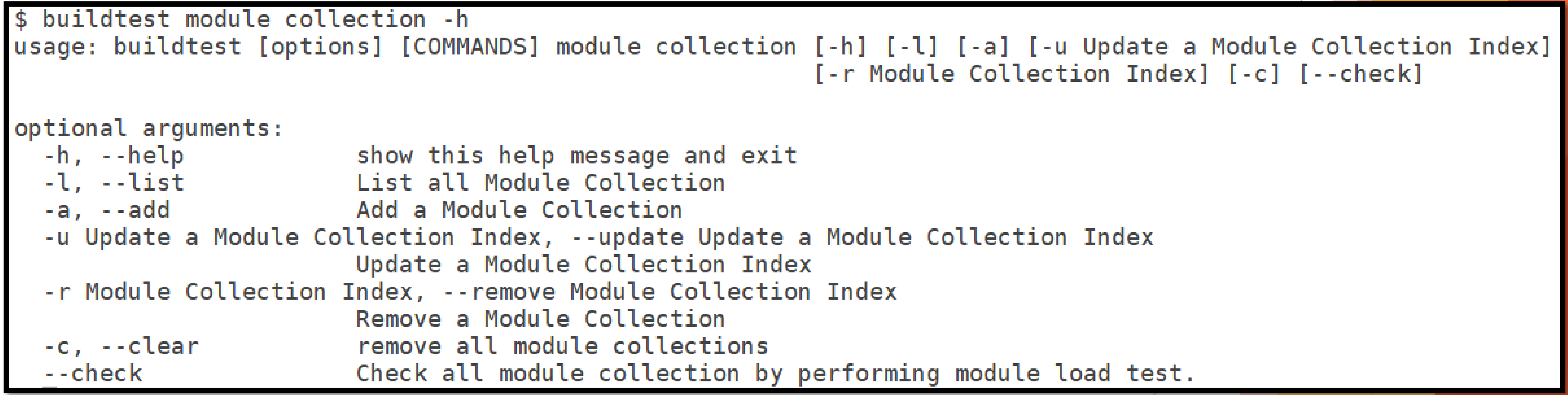


```
module restore GCC
TESTDIR=/tmp/ssi29/buildtest/tests/Intel/Haswell/x86_64/rhel/7.6/build_2
SRCDIR=/u/users/ssi29/gpfs/buildtest-framework/toolkit/suite/openmp/hello/src
SRCFILE=/u/users/ssi29/gpfs/buildtest-framework/toolkit/suite/openmp/hello/src/omp_hello.c
CFLAGS="-fopenmp"
cd $TESTDIR
gcc $CFLAGS -o omp_hello.c.yml.0x26b28a65.exe $SRCFILE
OMP_NUM_THREADS=2 omp_hello.c.yml.0x26b28a65.exe | grep -i threads
rm ./omp_hello.c.yml.0x26b28a65.exe
```

GitHub: https://github.com/HPC-buildtest/buildtest-framework
Documentation: http://buildtest.rtfd.io

Buildtest Module Collection System

- Lmod’s user collection must have unique collection name which is problematic when managing dozens of collections. Therefore, buildtest implements its own module collection system to store collections.
- Module Collection Operations:

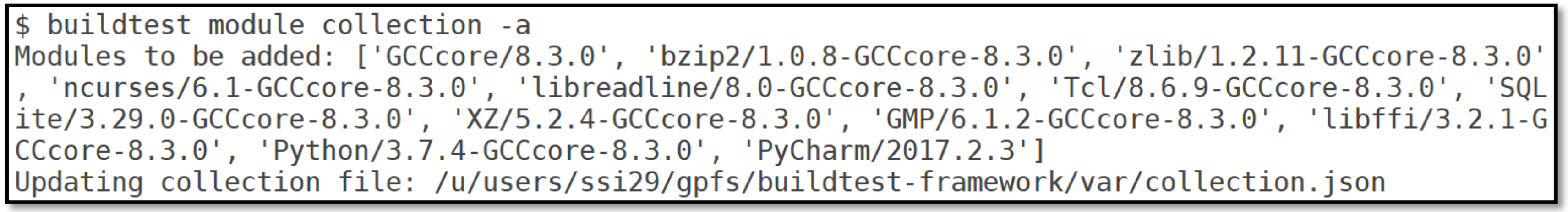


```
$ buildtest module collection -h
usage: buildtest [options] [COMMANDS] module collection [-h] [-l] [-a] [-u Update a Module Collection Index]
                                                        [-r Module Collection Index] [-c] [--check]

optional arguments:
  -h, --help            show this help message and exit
  -l, --list            List all Module Collection
  -a, --add             Add a Module Collection
  -u Update a Module Collection Index, --update Update a Module Collection Index
                        Update a Module Collection Index
  -r Module Collection Index, --remove Module Collection Index
                        Remove a Module Collection
  -c, --clear           remove all module collections
  --check               Check all module collection by performing module load test.
```

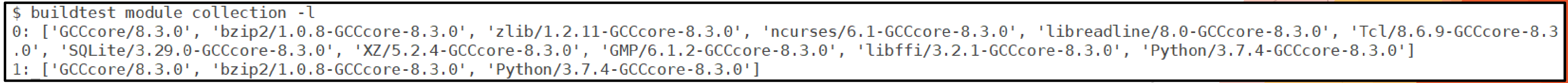
GitHub: https://github.com/HPC-buildtest/buildtest-framework
Documentation: http://buildtest.rtfd.io

Module Collection Operations

- Buildtest will store the module collection in collection.json that is self-maintained by buildtest
- To add modules to a collection use **buildtest module collection –a**
- Collection can be referenced by **collection id (0, 1, 2, …)**
- To list all module collections use **buildtest module collection –l**



```
$ buildtest module collection -a
Modules to be added: ['GCCcore/8.3.0', 'bzip2/1.0.8-GCCcore-8.3.0', 'zlib/1.2.11-GCCcore-8.3.0'
, 'ncurses/6.1-GCCcore-8.3.0', 'libreadline/8.0-GCCcore-8.3.0', 'Tcl/8.6.9-GCCcore-8.3.0', 'SQL
ite/3.29.0-GCCcore-8.3.0', 'XZ/5.2.4-GCCcore-8.3.0', 'GMP/6.1.2-GCCcore-8.3.0', 'libffi/3.2.1-G
CCcore-8.3.0', 'Python/3.7.4-GCCcore-8.3.0', 'PyCharm/2017.2.3']
Updating collection file: /u/users/ssi29/gpfs/buildtest-framework/var/collection.json
```



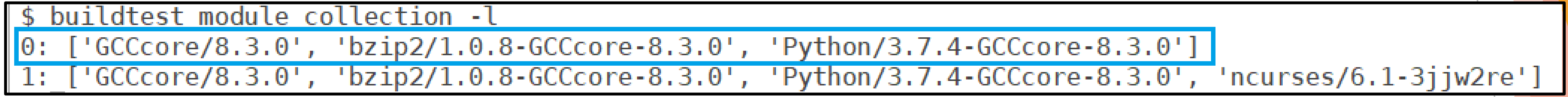
```
$ buildtest module collection -l
0: ['GCCcore/8.3.0', 'bzip2/1.0.8-GCCcore-8.3.0', 'zlib/1.2.11-GCCcore-8.3.0', 'ncurses/6.1-GCCcore-8.3.0', 'libreadline/8.0-GCCcore-8.3.0', 'Tcl/8.6.9-GCCcore-8.3
.0', 'SQLite/3.29.0-GCCcore-8.3.0', 'XZ/5.2.4-GCCcore-8.3.0', 'GMP/6.1.2-GCCcore-8.3.0', 'libffi/3.2.1-GCCcore-8.3.0', 'Python/3.7.4-GCCcore-8.3.0']
1: ['GCCcore/8.3.0', 'bzip2/1.0.8-GCCcore-8.3.0', 'Python/3.7.4-GCCcore-8.3.0']
```

GitHub: https://github.com/HPC-buildtest/buildtest-framework
Documentation: http://buildtest.rtfd.io

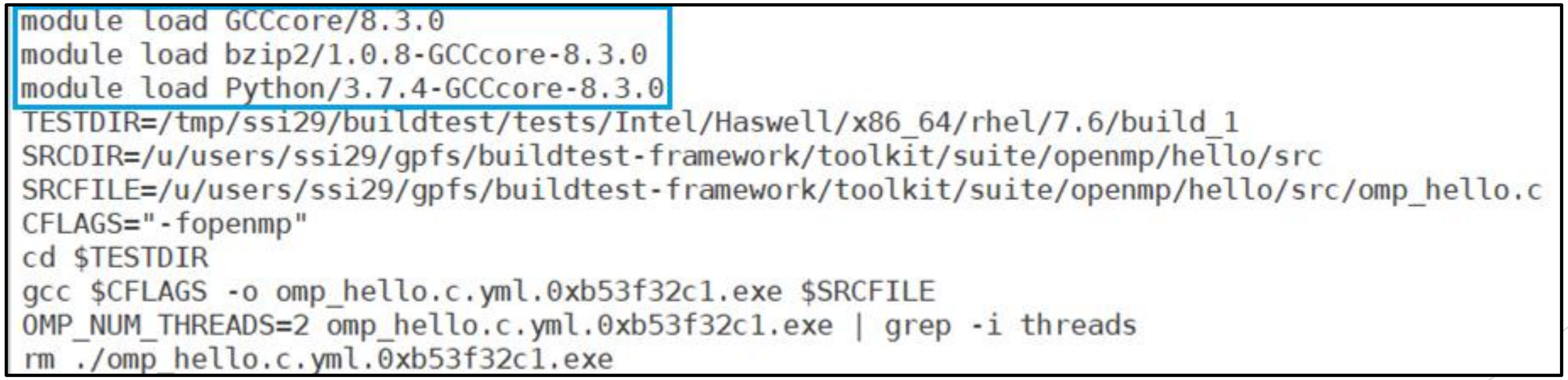
Building Test with Module Collection

- To build a test with module collection use the –-module-collection option.
- Let's build a test with collection 0 as follows:

buildtest build –c openmp.hello.omp_hello.c.yml –-module-collection 0



```
$ buildtest module collection -l
0: ['GCCcore/8.3.0', 'bzip2/1.0.8-GCCcore-8.3.0', 'Python/3.7.4-GCCcore-8.3.0']
1: ['GCCcore/8.3.0', 'bzip2/1.0.8-GCCcore-8.3.0', 'Python/3.7.4-GCCcore-8.3.0', 'ncurses/6.1-3jjw2re']
```



```
module load GCCcore/8.3.0
module load bzip2/1.0.8-GCCcore-8.3.0
module load Python/3.7.4-GCCcore-8.3.0
TESTDIR=/tmp/ssi29/buildtest/tests/Intel/Haswell/x86_64/rhel/7.6/build_1
SRCDIR=/u/users/ssi29/gpfs/buildtest-framework/toolkit/suite/openmp/hello/src
SRCFILE=/u/users/ssi29/gpfs/buildtest-framework/toolkit/suite/openmp/hello/src/omp_hello.c
CFLAGS="-fopenmp"
cd $TESTDIR
gcc $CFLAGS -o omp_hello.c.yml.0xb53f32c1.exe $SRCFILE
OMP_NUM_THREADS=2 omp_hello.c.yml.0xb53f32c1.exe | grep -i threads
rm ./omp_hello.c.yml.0xb53f32c1.exe
```

GitHub: https://github.com/HPC-buildtest/buildtest-framework
Documentation: http://buildtest.rtfd.io

Future Work

- Extend MPI support to include: IntelMPI, MPICH, MVAPICH2
- Extend compiler support to Intel, PGI, Clang.
- Setup CI server to run regression test for buildtest on every commit/PR
- Integrate CodeCov with CI build, codecov is already configured at https://codecov.io/gh/HPC-buildtest/buildtest-framework but coverage report are not automated
- Extend testtype: singlesource to support scripting languages such as Python, Perl, Ruby, R

GitHub: https://github.com/HPC-buildtest/buildtest-framework
Documentation: http://buildtest.rtfd.io

Conclusion

- Buildtest is a framework that automates test creation through YAML configuration. Buildtest comes with a repository of test configuration and source files, however community contribution is required in order to build a test repository with useful tests that will benefit the entire community.
- Software Stack Administrators can incorporate buildtest's software stack operation in their daily operation when managing their software stack.
- We need to build strong partnership in HPC community with respect to Software Stack Testing

GitHub: https://github.com/HPC-buildtest/buildtest-framework
Documentation: http://buildtest.rtfd.io

What's Next?

- **Clone**, **Star**, and/or **Fork** buildtest and join the community on SLACK.
- Contributing your Tests see: https://github.com/HPC-buildtest/buildtest-framework/blob/devel/toolkit/README.rst
- Contributing Guide: https://github.com/HPC-buildtest/buildtest-framework
- Report a Bug @ https://github.com/HPC-buildtest/buildtest-framework/issues

slack https://hpcbuildtest.slack.com/
HEROKU https://hpcbuildtest.herokuapp.com/
https://github.com/HPC-buildtest/buildtest-framework