Analyzing Frameworks for HPC Systems Regression Testing

Sadie Nederveld & Berkelly Gonzalez

Mentors:
Alden Stradling, Kathleen Kelly, Travis Cotton

LA-UR-20-26069

1

Background

What is Systems Regression Testing?

Our working definition: Tests that the system was set up correctly: eg. files have correct permissions, nodes can communicate, required mounts are present, updates have been implemented.

Checks to make sure that changes, including flaw fixes, that have been made do not get reverted and that new changes don't break other things that were previously working.

Checks things required for the specific system consistently over time via automation, not just in conjunction with maintenance windows.

Goal: identifying problems with the system before they become problems for the customers.

Early Research

- Began with a broad overview what regression testing is and looked for existing frameworks for HPC systems regression testing
  - “HPC Systems Acceptance Testing: Controlled Chaos”
  - “Getting Maximum Performance Out of HPC Systems”
  - ReFrame Readthedocs
- ~~ReFrame?~~ ← not actually systems regression!
- No frameworks for HPC systems regression testing
- Now what?

Our Project

- Goal: Identify frameworks that could be used for systems regression testing, and evaluate them to see which one(s) would work. If none, then determine that a systems regression framework would need to be built.

- Identified Pavilion, NHC, and Interstate as potential candidates

- Implemented Pavilion, Interstate, and NHC, and analyzed them based on a list of 22 characteristics.

2

Methods

Test Scripts

- Verify the host names of all nodes have not been altered
- Verify the node image on the management node has desired packages installed
- Ssh into the nodes and verify packages were installed
- Verify that the management node can ssh into expected systems but is denied entry into forbidden systems
- Check that required processes are currently running

Implementing Pavilion, Interstate, & NHC

Our Cluster:

a virtual cluster with one management node and 10 compute nodes

Pavilion:

“a Python 3 (3.5+) based framework for running and analyzing tests targeting HPC systems. It provides a rich YAML-based configuration system” - pavilion2.readthedocs.io

Interstate:

“a data-driven application that executes sub scripts for cluster testing or to change the state of the cluster” - Sam Sanchez, 2020

NHC:

written in bash, designed to periodically check the health of nodes or to be used for pre-job validation or post-job cleanup - NHC’s README.md

Desired Characteristics

Framework Capabilities

- Trivial framework extension
- More elaborate tools
- Test groups
- Tests encapsulated
- Test nesting
- Continuous testing
- Ongoing support

Output Tools

- Easy to find results both immediately after running and later (without splunk)
- Only view failed tests
- Output interpretation tools
- Clean output

Tracking Tools

- Alerts to email/command line/central page
- Export to time-series storage for plotting/reports
- Git

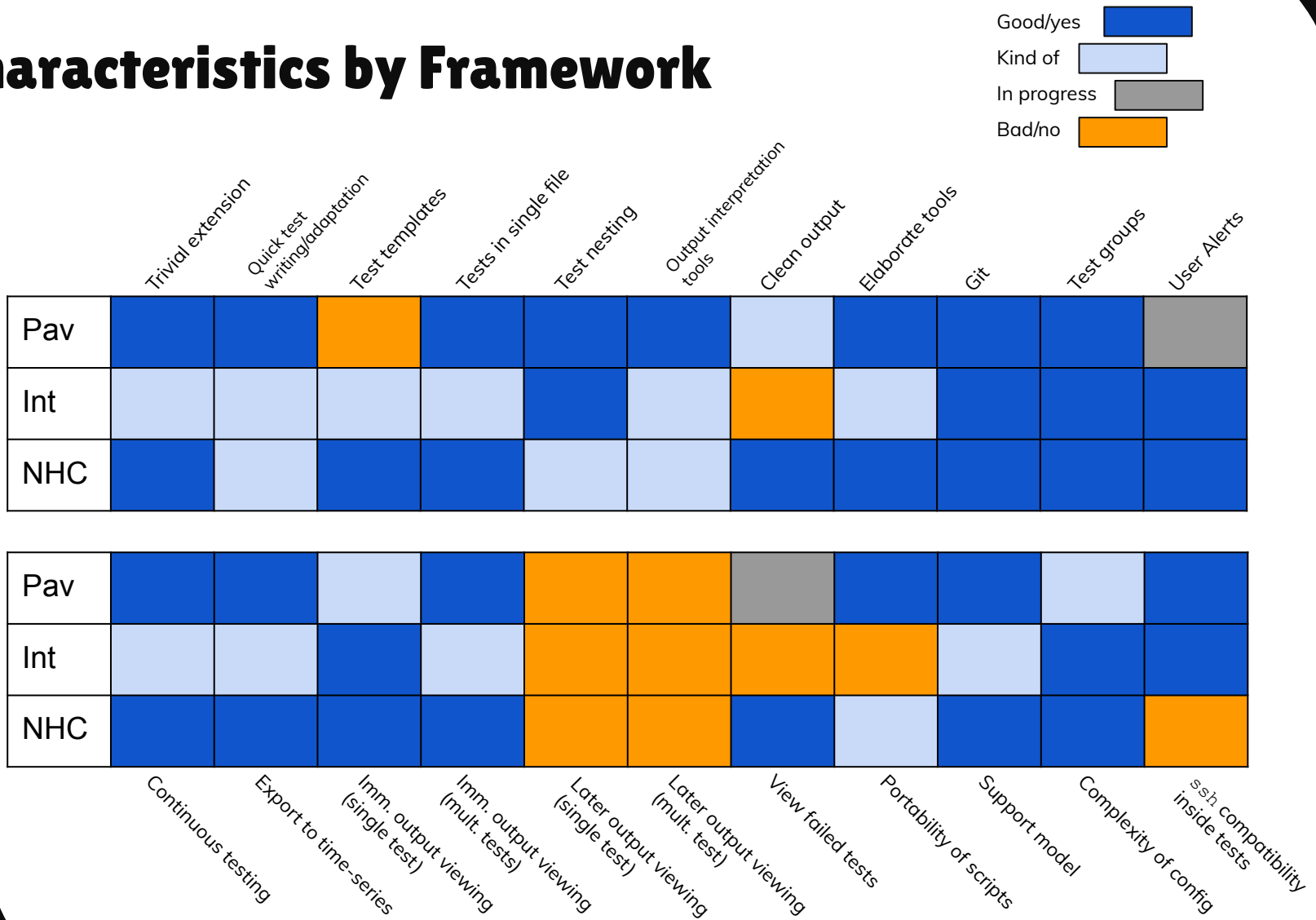
Test Creation

- Portability of test scripts
- Quick test writing/adaptation
- Test templates
- Simple configuration
- ssh compatibility (within scripts)

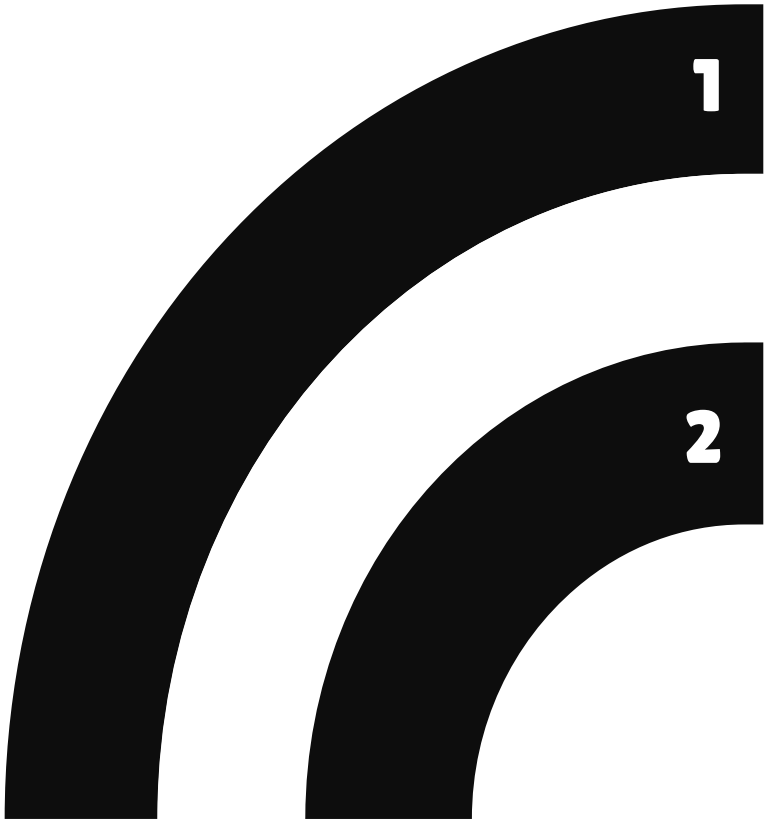
3

Results

Characteristics by Framework



Similarities between the Frameworks



All can have different test groups

- Pavilion: test suites (yaml)
- Interstate: modes within system config
- NHC: configs

None have easy viewing of test results after the test is run

- Pavilion: suite numbers are hard to find
- Interstate/NHC: output sent to loggers

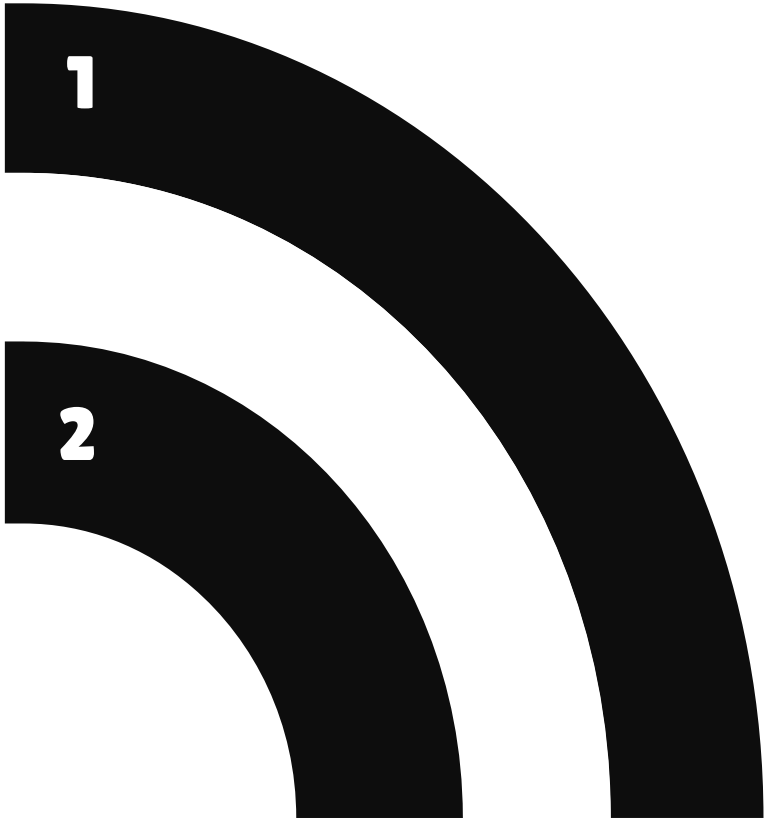
Differences between the Frameworks

Portability of test scripts

- Pavilion: takes in any script
- Interstate: have to tailor scripts
- NHC: have to tailor scripts

Built-in tests

- Pavilion: nothing built-in
- Interstate: has tests, but not useful for systems regression
- NHC: comes with many useful built-in tests



Retrospective Importance of Characteristics

Very Important

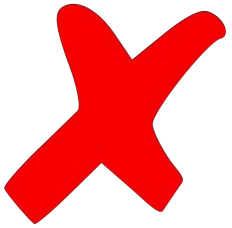
- Ability to export to a time series storage
- Good support model
- Clean and/or configurable output
- Quick test writing/adaption

Not so important

- Ability to only see the results of failed tests
- Ability to send email alerts

4

Conclusions

Interstate 

- Very good at what it does: changing the state of the system
- Designed to be simple so it was the smallest of the three frameworks we looked at in terms of features and documentation
- Currently missing too many necessary features to be used for systems regression testing on a wider scale
  - \+ Can set up splunk to watch it
  - \- Because it is not open source doesn't have as much support
  - \- Output cluttered
  - \- Idiosyncrasies that make adapting and writing tests a little difficult

Pavilion 

+ Splunk compatible
+ Open source, and LANL team is willing to take feature suggestions for future releases
+ Great result parsing
    - But the output is not easy to read or view without splunk
+ Able to run any test it is given

NHC 

+ Splunk compatible (via syslog file)
+ Only shows failed tests
+ Open source, widely used in HPC
+ Writing new tests is straightforward
    - pre-written tests need to be converted to bash and match NHC format

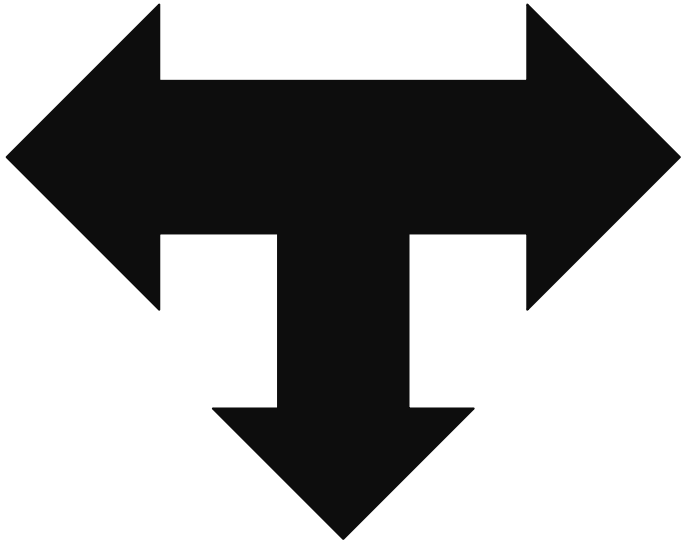
Pavilion

- No built-in tests or test templates
- No user/email alerts (in the works)
- Can’t view only failed tests (in the works)
+ Allows for test nesting
+ Extremely flexible, will run any kind of script you give it

NHC

+ Comes with several useful checks
+ Has user/email alerts
+ Only outputs failed tests
- Test nesting is tricky
- Checks have to be written in bash and have to conform to NHC’s expectations

Pavilion



NHC

Pavilion + NHC

Pavilion yaml to Run NHC

```
run_nhc:

   summary: runs NHC

   run:
      cmds:
         - “nhc -c {{nhc_confs}}”

   result_parse:
      regular expressions:
         error:
            regular expressions:
‘ERROR: (.*)’
            match_type: all

         result:
            regular expressions:
‘ERROR: (.*)’
```

Looking Back & Going Forward

- Systems regression testing is an underserved area in HPC
- Application testing can point out system failures, but does not do enough to monitor systems
- Need a consistent way of performing systems regression tests
- Test Pavilion and NHC together on a larger production system
- Determine if that solution is viable, or if another route needs to be explored

Acknowledgements

Thank you to:

- Paul Ferrell, Jen Green, and Francine Lapid from the Pavilion team, Sam Sanches and Corey Blount from Interstate, and Michael Jennings of NHC for the tutorials they gave us and the many questions they answered and bugs they helped us resolve.
- Our mentors, Alden, Kaki, and Travis
- Reid Priedhorsky and Julie Wiens
- Lowell Wofford and the TAs from bootcamp
- The other SI interns (special shoutout to Christine, Yolanda, and Anaira)
- 

Thank you!

Do you have any questions?

Berkelly Gonzalez

berkelly.gonzalez@gmail.com

(805) 698-6465

Sadie Nederveld

snederveld@lanl.gov
sadie.nederveld@gmail.com

(612) 816-0618

LA-UR-20-26069

Sources

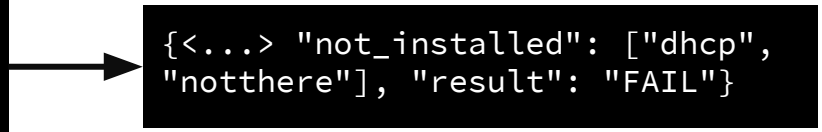
1. Hadri, B., Kortas, S., Fielder, R., & Markomanolis, G. S. (2017). Regression Testing on Shaheen Cray XC40: Implementation and Lessons Learned
2. Jennings, M., Lee, I., Robbert, M., Wickberg, T., & Kruiten, M. (2018, December 8). LBNL Node Health Check (NHC). Retrieved July, 2020, from https://github.com/mej/nhc
3. Johnson, N. (2019). *Software Testing for HPC* [PowerPoint Slides]. http://www.archer.ac.uk/training/course-material/2019/07/pract-sw-dev/06-Testing.pdf.
4. Khuvis, S., You, Z., Na, H., Brozell, S., Franz, E., Dockendorf, T., Gardiner, J., & Tomko, K. (2019, July). *A Continuous Integration-Based Framework for Software Management* [Abstract]. Retrieved from: https://dl.acm.org/doi/pdf/10.1145/3332186.3332219
5. Lang, J. (2017). *Linux Cluster Institute: Node Health Check (NHC)*. Lecture presented at LCI Workshop.
6. LANL Pre-Team (2019). Pavilion2. Retrieved July, 2020, from https://pavilion2.readthedocs.io/
7. Peltz, P., & Fields, P. (2016). *HPC Systems Acceptance: Controlled Chaos*. SC - HPC System Professionals Workshop ‘16. https://core.ac.uk/download/pdf/213848055.pdf
8. ReFrame 3.1. (n.d.). Retrieved June, 2020, from https://reframe-hpc.readthedocs.io/en/stable/
9. Vergara, V. M., & Hadri, B. (2019) *HPC System Testing: Procedures, Acceptance, Regression Testing, and Automation* [Session Description]. https://sc19.supercomputing.org/proceedings/bof/bof_pages/bof195.html
10. Young, C., Ball, K., Qualkenbush, A., Potts, M., Pasti, C., & Avart, D. (n.d.). Getting Maximum Performance Out of HPC Systems. *RedLine Performance Solutions*.

Output Examples

Pavilion

```
result_parse:
      regular expressions:
         not_installed:
            regular expressions: 'package (.*)
not installed'
            match_type: all

         result:
            regular expressions: 'not installed'
            action: store_false
```



```
{<...> "not_installed": ["dhcp",
"notthere"], "result": "FAIL"}
```

NHC

```
ERROR:  nhc:  Health check failed:
check_connect_ssh: ssh n11 returned 255.
ERROR:  nhc:  1 health checks failed.
```

Interstate

```
[    INFO    ] ------------------------------
[    INFO    ] Running on cluster: LITHIUM
[    INFO    ] Running in mode: BASIC
[    INFO    ] ------------------------------
[    INFO    ] Running on host: - li-master
[    INFO    ] Executing master scripts
[    INFO    ] ------------------------------
[   STATUS   ] crond is  active
[    INFO    ] Mode: [basic] mode not defined
[compute]. skipping...
```