XALT: Understanding HPC Usage via Job Level Collection

Robert McLay

April 20, 2017

XALT: What runs on the system

- A U.S. NSF Funded project: PI: Mark Fahey and Robert McLay
- A Census of what programs and libraries are run
- Running at TACC, NICS, U. Florida, KAUST, ...
- Integrates with TACC-Stats.

Design Goals

- Be extremely light-weight
- Provide provenance data: How?
- How many use a library or application?
- Collect Data into a Database for analysis.

Design: Linker

- XALT wraps the linker to enable tracking of exec's
- The linker (ld) wrapper intercepts the user link line.
- Generate assembly code: key-value pairs
- Capture tracemap output from ld
- Transmit collected data in *.json format

Design: Launcher

- XALT 1 used to require a wrapper for aprun, mpirun, etc
- XALT 2 no longer needs to
- Hooray! Correct wrappers were a nightmare!

Design: Transmission to DB

- File: collect nightly
- Syslog: Use Syslog filtering (or ELK)
- Direct to DB.
- Future: RabbitMQ

Lmod to XALT connection

- Lmod spider walks entire module tree.
- Can build a reverse map from paths to modules
- Can map program & libraries to modules.
- /opt/apps/i15/mv2_2_1/phdf5/1.8.14/lib/libhdf5.so.9 $\Rightarrow$ phdf5/1.8.14(intel/15.02:mvapich2/2.1)
- Also helps with function tracking.
- Tmod Sites can still use Lmod to build the reverse map.

Database Changes (I)

- Tables sizes in XALT:

Table	Size in MB
joinrunenv	199603.00
joinrunobject	9388.00
joinlinkobject	5013.00
xaltrun	4613.00
xaltobject	4175.00
xaltlink	814.00

- `join_run_env` has 2.1 billion rows

Database Changes (II)

- Environment variables are important.
- But mainly for reproducing results
- Chose a few for SQL tests.

Database Changes (III): New Design

- Store complete env $\Rightarrow$ compressed json blob
- Filter Env's with Accept Test followed by Reject Test
- Instead of 250 vars per job $\Rightarrow$ 20 to 30.
- The Filter is site controllable!

Database Changes (IV): New Design

- The “join” tables are large
- Partition “join” tables by dates or index
- Precompute views nightly.

Protecting XALT (I): UTF8 Characters

- Linux supports UTF8 Characters in file names, env. vars.
- Python supports UTF8 if you know what you are doing.
- Switch XALT to use prepared statements
- Where `query="INSERT INTO table VALUE(?,?)"`
- This prevent SQL injection: “johnny drop tables;”
- Also supports UTF8 characters.

Protecting XALT (II): Python to C++

- Difficult to protect Python from users in every case
- Solution: LD_LIBRARY_PATH="@ld_lib_path@" PATH=/usr/bin:/bin C++-exec ...
- Everything that depends on PATH must be hard coded
- basename ⇒ /bin/basename
- Unique install for each operating system.
- Certain programs aren't in the same place: `basename`

Using XALT Data

- Targetted Outreach: Who will be affected
- Largemem Queue Overuse
- XALT and TACC-Stats

Tracking Non-mpi jobs (I)

- Originally we tracked only MPI Jobs
- By hijacking mpirun etc.
- Now we can use ELF binary format to track jobs

ELF Binary Format Trick

```
void myinit(int argc, char **argv)
{
  /* ... */
}
void myfini()
{
  /* ... */
}
  attribute((section(".initarray")))
       typeof(myinit) *init = myinit;
  attribute((section(".finiarray")))
       typeof(myfini) *fini = myfini;
```

Using the ELF Binary Format Trick

- This C code is compiled and linked in through the hijacked linker
- It can also be used with LD_PRELOAD
- We are using both...

Downsides

- Currently, we only track task 0 jobs.
- MPMD programs will only record the Task 0 job.
- We also lose the ability to capture return exit status

Challenges (I)

- Do not want to track mv, cp, etc
- Only want to track some executables on compute nodes
- Do not want to get overwhelmed by the data.

Answers

- XALT Tracking only when told to
- Compute node only by host name filtering
- Executable Filter based on Path
- Protection against closing stderr before fini.
- Site configurable!

Path Filtering

- Accept test, following an Ignore Test,
- Two files containing regex patterns, converted to code.
- Accept List Tests: Track /usr/bin/ddt, /bin/tar
- Ignore List Tests: /usr/bin, /bin, /sbin, ...

Using XALT 2

- A great deal of hardening
- Been running XALT 2 for 6 months with only 2 tckts.

Speeding up XALT 2

- XALT 2 generates 2 json records: at start and end
- Want to minimize measurement: Launcher jobs
- The most expensive operation is sha1sum of the shared libs
- Used to system sha1sum call in serial
- Now up-to 16 threads calls directly
- Tests show 1 sec first time 0.04 second time on Lustre.

XALT Demo

- Show modules hierarchy
- ml –raw show xalt
- Show debugging output
- type -a ld,mpirun
- Build programs
- Run tests
- Run utf8 program
- Show database results

Conclusion

- Lmod:
  - Source: github.com/TACC/lmod.git, lmod.sf.net
  - Documentation: lmod.readthedocs.org
- XALT:
  - Source: github.com/Fahey-McLay/xalt.git, xalt.sf.net
  - Documentation: doc/*.pdf, xalt.readthedocs.org