TEXAS ADVANCED COMPUTING CENTER

WWW.TACC.UTEXAS.EDU

TEXAS

The University of Texas at Austin

Extracting Useful Data from XALT DB

Robert McLay

Jan. 20, 2022

XALT: Outline

- XALT Parts: Generating and Storing
- Quick Discussion about what is stored
- Caveats about XALT Records
- xalt_usage_report.py
- Kinds of reports available
- What kinds of reports would sites like?
- Future Topics for XALT zoom mtg.

XALT Generating and Storing

- ALL programs on system have XALT via LD_PRELOAD
- Some prgms generate *.json records
- These records can be written to a MySQL DB (not required)
- Afterward *.json records are deleted.

Caveats about XALT records

- Core-Hours not Node-Hours
- XALT knows about mpi tasks and threads
- It doesn't know if a node is shared.
- User might run multiple single core prgms on a node
- User might run ingle core prgms on one or more nodes
- XALT runs inside each program.
- It is blind to what happens outside of user prgm.
- It would require a daemon on each node to know.

XALT time records won't match Accounting

- XALT will filter out or sample programs
- It won't catch them all.
- XALT can double count (rarely)
- If user forks off another prgm $\Rightarrow$ double counting
- If user runs > 1 prgm per core.
- XALT uses real time not cpu time.

Python prgm: sbin/xalt_usage_report.py

- An attempt to be system agnostic.
- It provides a name mapping file.
- It reports data that TACC has found useful:
  - Overall Job counts
  - Self Built vs. Not.
  - It reports Top Execs by Core-Hours, Number of Runs, Number of users for All, MPI Only, Scalar
  - Top Module usage
  - Compiler usage
  - Library Usage

Overall Job counts

- Core hours for “system” prgms vs. user prgms.
- System prgms come from a module, User prgms don’t.
- We see about 5% for system prgms.
- Still have to teach how to build program.

Self-Built vs. Not.

- Beside system supplied program we have groups that share prgms
- Like to track that.
- Prgms built under XALT know build user.
- XALT knows the run user.
- We report 2 to 4 % non self-built prgm runs.

xalt_name_mapping.py

```
equiv_patternA = [
    [ r'^pmemd'                  , 'Amber*'       ],
    [ r'^sander'                 , 'Amber*'       ],
    [ r'^absorption.realx'  , 'BerkeleyGW*' ],
    [ r'^absorption.cplxx'  , 'BerkeleyGW*' ],
    [ r'^kernel.realx'      , 'BerkeleyGW*' ],
```

- Map prgms to projects
- Sites may need to modify to match their site
- Names get a when mapped
- Nothing stops a user naming Hello $\Rightarrow$ pmemd

Typical Reports: Top Execs Core-Hours

CoreHrs	# Runs	# Users	# Accts	Exec
289,924,113	710	13	7	CESM*
68,162,422	174,594	8	4	Chroma*
51,135,767	102,957	5	3	gene_fta
39,173,897	22,666	44	32	LAMMPS*
38,714,742	75,751	50	39	NAMD*
35,858,260	254,470	55	37	VASP*

Typical Reports: Top Execs Runs

CoreHrs	# Runs	# Users	# Accts	Exec
2,391,403	815,586	1	2	Rosetta*
2,402	583,091	136	97	mv
7,068	515,852	373	211	grep
11,942,647	499,463	391	218	Python*
24,176	414,788	1	2	MOPAC2016

Typical Reports: Top Execs Users

CoreHrs	# Runs	# Users	# Accts	Exec
11,942,647	499,463	391	218	Python*
7,068	515,852	373	211	grep
1,330	67,595	335	197	sed
12,779	121,983	322	182	gawk
2,402	583,091	136	97	mv
16,366	131,215	132	94	cp
137,580	21,731	103	62	perl

Typical Reports: Scalar Top Execs Core-Hours

CoreHrs	# Runs	# Users	# Accts	Exec
2,852,146	303,289	4	2	rockstr-glx
2,809,544	469,191	336	195	Python*
2,391,403	815,586	1	2	Rosetta*
908,214	186,031	23	11	R
747,719	22,940	1	2	squid

Side Note: Integrating XALT testing into github

- I would like to use github actions
- Have not figured out how to setup mysql inside container

Conclusions

- xalt_extract_record user_program to see the XALT Watermark
- Use file transport to check *.json
- Use logger transport to check syslog tracking
- Possibly use xalt testing locally.
- Willing to work with anyone who tries any system.

Future Topics?

- Package tracking
- Extracting results from the DB.
- Others?