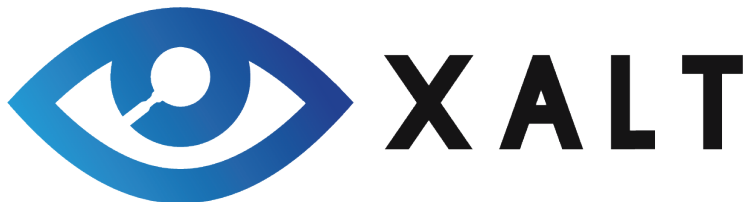


XALT Monthly Meeting: Controlling Executables Tracked

Robert McLay

Sept. 9, 2021

XALT: Outline



- ▶ XALT's DB can easily get overwhelmed
- ▶ Three ways to control what gets recorded
- ▶ TACC's goals are to track science
- ▶ Config/TACC_config.py discussion
- ▶ Sampling

Preventing XALT's DB getting overwhelmed

- ▶ Must do something to prevent too much data
- ▶ Want to understand what is run on your system w/o tracking everything
- ▶ Tales of Woe:
 - ▶ One 2 hour job ran over 2M times \Rightarrow 4.5 days of data
 - ▶ One user used a 4 core MPI program to train a neural network
- ▶ XALT supports sampling for both MPI and Non-MPI programs

Three ways to control the amount of recorded data

- ▶ Use path_patterns to ignore many executables based on path.
- ▶ Sampling to reduce the amount of executables tracked based on runtime.
- ▶ Pre-ingest filtering to sample data already tracked. (only works w/ storing data in the MySQL DB)

Tracking Science

- ▶ We want to know what science is being done
- ▶ What apps are run and build by users or in /opt/apps
- ▶ Sample apps like cp, perl, awk, and sed
- ▶ Sites are free to configure XALT as they see fit.

Config.py \Rightarrow generated code

- ▶ For speed reasons, the patterns are converted to a lex program
- ▶ This lex program is converted into executable code
- ▶ These patterns are then directly callable from the XALT shared library.
- ▶ This pattern matching is very fast.
- ▶ Downside: Any changes in path_patterns must be rebuilt.

Config/TACC_config.py

- ▶ This file is what TACC uses.
- ▶ Note that the first match found is used.
- ▶ Mark R, MATLAB and Python as PKGS that track imports
- ▶ Keep cp, sed, awk, perl and similar from /bin or /usr/bin
- ▶ Ignore anything else in /usr/bin or /bin
- ▶ Ignore compilers, mpi wrappers, shell scripts (/bin/bash is ignored)
- ▶ Ignore test programs from Cmake and autoconf's config

Sampling

```
interval_array = [  
    [ 0.0, 0.0001 ],  
    [ 1800.0, 0.01 ],  
    [ 7200.0, 1.0 ],  
    [ sys.float_info.max, 1.0 ]  
]  
mpi_interval_array = [  
    [ 0.0, 0.0001 ],  
    [ 900.0, 0.01 ],  
    [ 1800.0, 1.0 ],  
    [ sys.float_info.max, 1.0 ]  
]
```


path_patterns

```
path_patterns = [  
    ['PKGS', r'.*\python[0-9.]*'],  
    ['KEEP', r'^\bin\gawk'],  
    ['KEEP', r'^\bin\sed'],  
    ['KEEP', r'^\bin\perl'],  
    ['SKIP', r'^\bin\.'],  
    ['SKIP', r'^\opt\intel\.'],  
    ['SKIP', r'^\opt\apps\git\.'],  
    ['SKIP', r'^\opt\apps\cmake\.'],  
    ['SKIP', r'^\opt\apps\autotools\.'],  
    ['SKIP', r'^\opt\apps\intel[0-9][0-9\_]*\impi\.'],  
    ['SKIP', r'^\opt\apps\gcc[0-9][0-9\_]*\mvapich2\.'],  
    ['SKIP', r'^\home1\apps\intel\.'],  
    ['SKIP', r'.*\mpicc'],  
    ['SKIP', r'.*\mpicxx'],  
    ['SKIP', r'.*\mpif90'],  
    ['SKIP', r'.*\conftest'],  
    ['SKIP', r'.*\CMakeTmp\cmTryCompileExec[0-9][0-9]*'],  
    ['SKIP', r'.*\CMakeTmp\cmTC_[a-f0-9][a-f0-9]*'],  
]
```

Before XALT-2.10.30

A site did something like this:

```
path_patterns = [  
    ['KEEP', r'^\./opt\./apps\./.*'],  
    ['SKIP', r'.*'],  
]
```

Where XALT was installed /opt/apps/xalt

xalt_configuration_report (XALT-2.10.30) and later

```
*-----*
Array: pathPatternA
*-----*
===== src/tmpl/xalt_config.py =====
0: SKIP => .*\/bin\/my\_uuidgen
1: SKIP => .*\/logger
2: SKIP => .*\/xalt\_print\_os
3: SKIP => .*\/xalt\_syslog\_to\_db
4: SKIP => .*\/xalt\_extract\_record.x
5: SKIP => .*\/xalt\_configuration\_report.x
6: SKIP => .*\/xalt\_syshost
7: SKIP => .*\/xalt\_record\_pkg
8: SKIP => ^\/opt\/apps\/xalt\/xalt\/.*
===== /opt/apps/xalt/Config/simpleConfig.py =====
9: KEEP => ^\/opt\/apps\/.*
10  SKIP => .*
===== src/tmpl/xalt\_config.py =====
11: KEEP => .*
```

Conclusions

- ▶ Thanks for listening
- ▶ Questions?