Black Box Testing of APEL

Steve Jones
University of Liverpool
September 2017

Site Used For The Test

- I did a black box test of APEL at Liverpool for the month of September 2016.
- We had four clusters then.
  - ARC/HTCondor on SL6
  - CREAM/Torque system (now demolished)
  - Local VAC
  - Remote VAC (from central computer service)
  - ARC/HTCondor system on Centos 7 (came later)
- All these systems recorded the work they did and passed it into APEL.
- But were all the records being counted right?

What is it?

- Black-box testing examines the functionality without peering into the internal workings. The tester knows what the software is supposed to do but need not be aware of how it does it. [wikipedia]
- So I won't worry about how APEL creates, transfers, transforms or stores its data.
- Instead I will compare what goes into APEL (accounting logs) and what comes out (i.e. what APEL shows when I query it.)
- Then I can tell that it works end-to-end, and I will know:

  - My site is set up well to work with APEL.
  - APEL does its job well.

Overall Goal Of The Test

- I hoped that test method would give overall results that were within ~ 0.5% of the values represented in the egi portal.
- I expected (and I got!) small discrepancies, which could be down to minor boundary case problems, time zones, rounding and slightly false assumptions.

I “Reimplemented” The Parsers

- Black box testing hides most implementation details. But, for a test entry point, I needed to read the primitive records with my own parsers. Once I had those, I did not need to care that:
  - CREAM/Torque sends its data via a dedicated APEL system, which has a local mysql db.
  - ARC/HTCondor sends its data via an application called Jura, which archives its data in files, not in a DB.
  - Each VAC factory machine (i.e. each VM host) sends its own APEL records one at a time when the need arises. Again, it has no DB, but uses files.
  - Etc.
- Note: now the test will also show that APEL's parsers are correct, since I “reimplemented” the parsers myself.

Results: Sync Record Counts

- ARC/Condor -
  - Test Sync 179,929, Apel Sync 179,929
- CREAM/Torque -
  - Test Sync 66,047, Apel Sync 66,047
- Vac (main) -
  - Test Sync 21,386, Apel Sync 21,385
- Vac (remote) -
  - Test Sync 3,779, Apel Sync 3,777

Results: HS06 Comparisons

- ARC/Condor -
  - Site work 7,089,027, Apel work 7,089,222
- CREAM/Torque -
  - Site work 1,780,150, Apel work 1,780,198
- Vac (main) -
  - Site work 4,071,437, Apel work 4,109,924
- Vac (aux) -
  - Site work 50,248, Apel work 50,524

(Work is in HS06 hours)

Final Result

- With the test method, I get overall,
  - Total Site HS06 hours: 12,990,862
  - Total Apel HS06 hours : 13,029,868
- This is a delta of 0.3%; within the 0.5% tolerance I allowed myself.
- To squeeze the final error right out would need more work to understand the criteria used completely. It's close enough.
- Hence I codfied the method so I could do this (say) every month to ensure the records I see in the APEL tables actually match what work we did (i.e. we used the black box test to audit the site).
- I describe the method in the next slides.

Method

- We don't _know_ that the APEL system is correct.
- We don't _know_ if my site uses the APEL system correctly.
- So I measure the work we did independently of the APEL system.
- If my measurement is equal* to the measurement exposed by the APEL system, then the chance of a significant error being present in either system is small.
- And if that outcome holds for a long period, then the chance of an error asymptotically approaches zero.

* Where “equal” really means “very close”.

Realising The Method

- I wrote a suite of tools to realise the idea for each of the systems we used at the site.
  - VAC
  - CREAM/Torque
  - ARC/HTCondor
- I assume node benchmarking and configuration has been done properly (perhaps using this or similar):

  https://www.gridpp.ac.uk/wiki/Benchmarking_procedure

- Since I was the only user, I only paid attention to correctness; I made no special effort to make the tools user-friendly.
- I take the readings on the most convenient system to parse. On the CREAM/Torque system, I parse the Torque “batch system logs”. On the ARC/HTCondor system, I parse the ARC CE's “usage records”. And on VAC, I visit every VAC factory and parse their “apel archive” files.

Tools

- To get the tools from github:
  - git clone https://github.com/gridpp/audit.git
- Structure:

```
.
|-APEL
|-ARC
|-general
|-Torque
|-VAC
|---controlnode
|---factory
```

Measuring VAC

For global figures (not specific to any experiment). Install the software on all the VAC nodes, off the root dir. Also install it on a control node, i.e. a node that you can use to access all the vac nodes with ssh but without a password (ssh-agent, ssh-add ...)

```
cd audit/VAC/controlnode/

Get the UNIX epochs for the start and end of the period in question

startEpoch=`date --date="Jan 01 00:00:00 UTC 2017" +%s`
endEpoch=`date   --date="Feb 01 00:00:00 UTC 2017" +%s`

Get list of all the vacnodes in a file called vacnodes, then:

for n in `cat vacnodes`; do
  ./simpleAudit.sh  $n /var/lib/vac/apel-archive  \
    1483228800  1485907200;
done > table.jan

Now process the file to get the result.

./accu.pl table.jan

The work done for that period, in HS06 hours, should pop out. The job count
for the month is represented by the number of lines in the table file.
```

Measuring ARC

Archiving must be turned on in /etc/arc/conf. Check:

https://www.gridpp.ac.uk/wiki/Example_Build_of_an_ARC/Condor_Cluster

Make a list of all the usage reports (location varies, check /etc/arc.conf)

```
ls /var/urs > /tmp/urs
```

Get the ones for jobs that started in (say) sept

```
for f in `cat /tmp/urs`; do
  grep -l "EndTime.2016-09" /var/urs/$f;
done > /tmp/urs.sept
```

Parse them to make the table

```
for t in `cat /tmp/urs.sept `; do  ./parseUrs.pl $t; done > table.sept
```

Sum up the table

```
cat table | ~/scripts/accu.pl
```

The usage for the month should pop out. The job count for the month is represented by the number of lines in the table file.

Measuring Torque

```
# Note: ... can't test mcore fully (we don't use torque for mcore)

Get a list of the files that cover the period. Some records for Oct might lie
in Aug or Sept, so list those too. The location of the files may vary; check
with your admin guy.

ls /var/lib/torque/server_priv/accounting/201609* > recordFilesCoveringPeriod
ls /var/lib/torque/server_priv/accounting/201610* >> recordFilesCoveringPeriod
ls /var/lib/torque/server_priv/accounting/201611* >> recordFilesCoveringPeriod

Get the UNIX epochs for the start and end of the period
startEpoch=`date --date="Oct 01 00:00:00 UTC 2016" +%s`
endEpoch=`date   --date="Nov 01 00:00:00 UTC 2016" +%s`

You have to pass the script the Publishing Benchmark to which you scale. At
Liverpool, we scale to 10 HS06, which is 2500 bogoSpecInt2K

./extractRecordsBetweenEpochs.pl recordFilesCoveringPeriod 2500 \
   $startEpoch  $endEpoch > table.oct

Add up the tables to get the result for the month.

./accu.pl table.oct

The work done for that month, in HS06 Hours, should pop out. The job count for
the month is represented by the number of lines in the table file.
```

Check Sync Records

Checking VAC systems (sync)
****

Visit each node in your VAC cluster, and count the files under (e.g.) /var/lib/vac/apel-archive which match the month you want to check (note that this method gives better results than actually opening the parsing the files to find where their end date lies! This is a mystery.)

Checking CREAM/Torque systems (sync)
****

On the Torque headnode, surf through all the records under server_priv/accounting/... Open them up and find E records with end times inside the month you want to check. Count them all.

Checking ARC/Condor systems (sync)
****

Surf though all the URS archive records, finding ones that have "EndTime.2016-08", i.e. where August in the month you want to check. Count all those.

If all is well, the grand total of the VAC, CREAM/Torque and ARC/Condor totals will equal the APEL sync record total within a few records.

How To Get The APEL Figures

- Using the tools, I now know what work each part of my site did.
- Next I need to find what APEL has recorded, and compare it.
- If they are about the same, I'm done with it. If not, I'll have to figure out what went wrong.
- So the next slides will show how I got the APEL data used to compare with the black box test described earlier.
-

Use The Accounting Portal

- Browse to https://accounting.egi.eu
- Surf to Research Infrastructures, Tier 2, UK
- Set metric as Sum Wallclock Work HS06 Hours
- Surf to Northgrid, Liverpool (or whatever)
- Chose your Month and All VOs
- Set Row variable to Submit Host
- Hit update to get the breakdown by CE, and the totals for that month.
- Since a VAC factory equates to a CE, you will need to download the data as CSV and use awk to tot up the VAC work into one figure.
- And that's it; just compare the figures and they will match my test outputs in the first part of these slides.

Further Work

- The software is rough and ready, since I only intended to use it myself.
- But the test was successful as a proof of concept, and the tools are useful enough if a person really wants to know their setup is working, and doesn't mind a bit of hacking (e.g. awk and whatnot.)
- If I were to bring this idea into the formal software baseline, I'd spend a week or two rounding off the rough edges and documenting the process properly.
- It all depends on what the project needs. If it is a requirement for sites to audit their accounting from time to time, it would be right to make a systematic effort to make the process seamless for any user, regardless of their knowledge or experience.
- Also, other parsers would be needed for the systems we don't use.
- In summary, it's rough and ready but the results are good.

That's all

- For support: sjones@hep.ph.liv.ac.uk