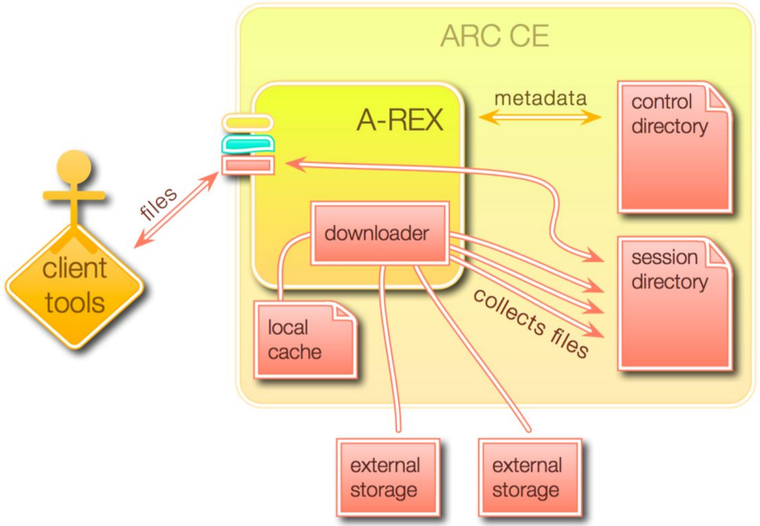
ARC Cache

David Cameron
University of Oslo
DOMA Access, 16.10.18

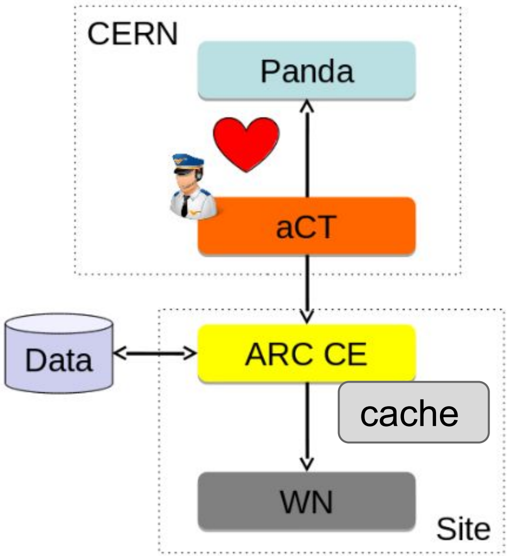
ARC Cache intro

- ARC CE maintains an internal cache of input files on a shared filesystem
- Before submitting to the batch system input files for jobs are downloaded to the cache by the CE
    - Then symlinked to the job's working directory
    - The download is skipped if the file is already in the cache
- Cache space is managed using LRU with high and low watermarks in arc.conf



ARC Cache usage

- Where is it useful?
  - Sites with a shared filesystem and no local grid storage
- Currently used in production for ATLAS at NDGF computing sites and several HPCs in Europe and Asia
  - Piz Daint (CH), SuperMUC (DE), IT4I (CZ), MareNostrum (ES), Lustiania (ES), Tianhe-1 (CN)
- Requires jobs submitted in push mode to ARC CE via ARC Control Tower (aCT)
  - aCT picks jobs from Panda, then submits to CE with correct job requirements (wall time, memory etc) and input files defined



ARC Cache R&D

- How to use in a pilot model, or without ARC CE?
  - Have the pilot download data with “arccp -y /cache”
  - Requires a tool to manage cache space (cache-clean script used by ARC CE)
- Integration into Rucio
  - Publication of contents as a “volatile” RSE
  - ARC CE transfer system as an alternative transfer tool to FTS, to pre-populate cache

Study of cache hit rates

- Data was taken from data transfer logs of each job from one ARC CE at SiGNET (NDGF site in Slovenia)
  - Mix of production and analysis workloads
- Time period 1-12 April 2018
- Only data files
  - Excluding pilot tarball, wrapper etc used by every job

Raw numbers

- 14177 unique files were downloaded to cache
- 68387 files were already cached (each cache hit of the same file is counted)
- 36179 unique files were already cached
- Naive cache-hit ratio of 82.8%
  - Starting from a warm cache

In more detail

- Most used files (hit count, filename)

1548 panda.0408182750.492125.lib._13669340.13145195029.lib.tgz
1150 AOD.05536542._000001.pool.root.1
1132 panda.0409081432.8074.lib._13669277.13147637952.lib.tgz
707 panda.0412170344.283124.lib._13702696.13174834478.lib.tgz
689 panda.0408015155.51359.lib._13665634.13142602942.lib.tgz
428 panda.0406213309.751814.lib._13660211.13138024850.lib.tgz
419 panda.0411155552.708753.lib._13694942.13168445743.lib.tgz
383 panda.0412130327.756600.lib._13701264.13172797838.lib.tgz
275 panda.0406154926.166815.lib._13658415.13136095000.lib.tgz
248 group.phys-gener.madgraph5223p4.363608.MGPy8EG_N30NLO_Wenu_Ht140_280_13TeV.TXT.mc15_v1._00001.tar.gz
202 panda.0404172008.118576.lib._13630830.13123934780.lib.tgz
199 panda.0406114855.300231.lib._13656780.13134782656.lib.tgz
184 panda.0406142042.481548.lib._13657959.13135685021.lib.tgz
182 panda.0402195751.863505.lib._13630124.13096051421.lib.tgz
173 panda.0406140438.591229.lib._13657774.13135439801.lib.tgz
163 panda.0406165914.440641.lib._13659017.13136460875.lib.tgz

panda...lib.tgz is user code required by every job in an analysis task

Excluding lib.tgz

Analysis test jobs input

1150 AOD.05536542._000001.pool.root.1
248 group.phys-gener.madgraph5223p4.363608.MGPy8EG_N30NLO_Wenu_Ht140_280_13TeV.TXT.mc15_v1._00001.tar.gz
138 group.phys-gener.madgraph5223p4.363605.MGPy8EG_N30NLO_Wenu_Ht70_140_13TeV.TXT.mc15_v1._00001.tar.gz
135 EVNT.740625._000008.pool.root.1
132 group.phys-gener.sherpa020201.363491.Sherpa_221_NNPDF30NNLO_lllv_13TeV.TXT.mc15_v1._00001.tar.gz
100 EVNT.04972714._000023.pool.root.1
97 EVNT.740625._000007.pool.root.1
91 EVNT.740625._000009.pool.root.1
79 EVNT.04972714._000022.pool.root.1
71 EVNT.740625._000001.pool.root.1
68 EVNT.740625._000010.pool.root.1
66 EVNT.04972714._000036.pool.root.1
62 EVNT.04972714._000026.pool.root.1
61 HITS.10701323._008539.pool.root.1
60 EVNT.740625._000004.pool.root.1

HammerCloud tests input

Event generation input

Excluding lib.tgz gives 48317 cache hits: 77% cache hit

Statistics by datatype

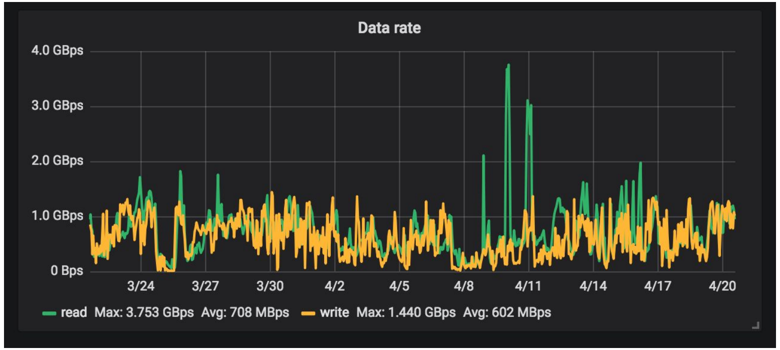
Files downloaded to cache but not used (within the time period of the study)	Unique files read from cache	% chance of being read from cache at least once
4944 DAOD	19396 DAOD	DAOD: 79.7
3329 AOD	9894 AOD	AOD: 74.8
1487 EVNT	3407 EVNT	EVNT: 69.6
1262 HITS	1560 HITS	HITS: 55.3
430 panda	1186 panda	panda: 73.4
303 data17_13TeV	286 data17_5TeV	
95 user	215 user	
76 TXT	72 DRAW_ZMUMU	
59 DRAW_ZMUMU	58 data17_13TeV	
32 DRAW_RPVLL	43 RDO	
22 RAW	29 DRAW_EGZ	
21 log	19 group	
19 data16_hip8TeV	13 log	
11 RDO		
10 DRAW_EGZ		

Cache details

- SiGNET cache is 250TB
- Ceph filesystem, mounted on the worker nodes (jobs access data directly from cache)
- Turnover is 5 days, i.e. 50TB/day, 600MB/s average download rate

```
pikolit ~ # /usr/libexec/arc/cache-clean -s -c /etc/arc.conf

Usage statistics: /ceph/grid/cache
Total deletable files found: 240657 (9681 files locked or in use)
Total size of deletable files found: 217 TB (16 TB locked or in use)
Used space on file system: 583 TB / 697 TB (83.63%)
At size (% of total)  Newest file                Oldest file
21 TB (10%)             Fri Apr 20 15:11:05 2018  Fri Apr 20 03:57:00 2018
43 TB (20%)             Fri Apr 20 03:56:59 2018  Thu Apr 19 18:57:36 2018
65 TB (30%)             Thu Apr 19 18:57:36 2018  Thu Apr 19 11:38:04 2018
86 TB (40%)             Thu Apr 19 11:37:57 2018  Wed Apr 18 17:49:19 2018
108 TB (50%)            Wed Apr 18 17:49:18 2018  Wed Apr 18 08:45:04 2018
130 TB (60%)            Wed Apr 18 08:45:04 2018  Tue Apr 17 20:39:44 2018
152 TB (70%)            Tue Apr 17 20:39:44 2018  Tue Apr 17 05:25:14 2018
173 TB (80%)            Tue Apr 17 05:25:14 2018  Mon Apr 16 22:15:03 2018
195 TB (90%)            Mon Apr 16 22:15:00 2018  Mon Apr 16 15:29:51 2018
217 TB (100%)           -                        Sun Apr 15 14:16:16 2018
```



Caching strategy

- Blindly caching all input files has been the ARC strategy for years but still seems to mostly work with current ATLAS jobs
    - In fact previous studies showed ~50% hit rate so caching is even better for current workflows
- However it can be that some jobs (eg analysis) read small fractions of the file - here byte-level caching is better but only if those small fractions are re-read
- At SIGNET there are two modes (two Panda queues) working in parallel for analysis:
    - Direct: panda estimates workdir size with all the input/output files - this is kept low at 5GB and forces most of the jobs to use direct I/O. It forbids large jobs that need local inputs
    - Caching: all the other jobs, where posix I/O is needed. those are also typically complicated (eg reprocessing, mc reco), where full input files are read
- The direct queue processes 3x more jobs than the caching one
- Note the direct queue does not count towards the previous statistics, which may distort the numbers

Conclusion

- With NDGF’s “data lake” (distributed dCache) it is essential to cache locally to jobs
    - 10+ years experience with ARC cache
- A model like ARC cache is also suitable for sites without grid storage or WN network connectivity like HPCs
- Issues with this model
    - If there are too many inputs to download, they saturate the WAN and keep cores unused
    - If the cache is not sized correctly it can have a fast turnover so some files can be downloaded many times over a short period
    - We do not yet use cached files locations in brokering - this can randomly distribute jobs using the same input over many centres