Fabrizio Furano: “From IO-less to Networks”

Exercises

Test setup for LAN exercises

- Prepare a 8GB file with your unique name

```
>dd of=/tmp/<yourname>.dat if=/dev/zero bs=1048576 count=8192
```

  - Copy it to the xrootd server of the school

```
>xrdcp /tmp/<yourname>.dat root://esc09-master:1095//<yourname>.dat
```

  - Copy it back to check it

```
>xrdcp –v -f root://esc09-master:1095//<yourname>.dat /dev/null
```

  - Copy the input file to your home dir

```
>cp /nfsmaster/track2_furano/Track2progs/inputfile.txt ~
>export PATH=/nfsmaster/track2_furano/xrootd-20091012/bin/arch:$PATH
```

Exercise: local seq sparse access

- Write a program which reads 1Kb every 10KB up to the end of the file.
  - Clear the cache before each run with the tool “clearcache”
  - See how it performs
  - Estimate the average apparent latency per request

Exercise: local seq sparse access

- Write a program which reads 1Kb every 10KB
  - But this time it does it backwards
  - The reads must be the same as the previous exercise
  - See how it performs
  - Estimate the average apparent latency per request

TestXrdClient_read

- A test program for xrootd data access
  - Interprets the standard input as a sequence of requests to satisfy
  - The cmd line parameters modify the way it works
    - Read ahead size, buffer cache size, readv usage, …

TestXrdClient_read

```
This program gets from the standard input a sequence of
 <length> <offset>             (one for each line, with <length> less than 16M)
 and performs the corresponding read requests towards the given xrootd URL or to ALL
 the xrootd URLS contained in the given file.

Usage: TestXrdClient_read <xrootd url or file name> <blksize> <cachesize> <vectored_style>
    <inter_read_delay_ms> [--check] [-DSparmname stringvalue]... [-DIparmname intvalue]...

 Where:
  <xrootd url>          is the xrootd URL of a remote file
  <rasize>              is the read ahead size. Can be 0.
  <cachesize>           is the size of the internal cache, in bytes. Can be 0.
  <vectored_style>      means 0: no vectored reads (default),
                              1: sync vectored reads,
                              2: async vectored reads, do not access the buffer,
                              3: async vectored reads, copy the buffers
                                (makes it sync through async calls!)
                              4: no vectored reads. Async reads followed by sync reads.
                                (exploits the multistreaming for single reads)
                              5: don't read, but write data which is compatible with the --check option.
  <inter_read_delay_ms> is the optional think time every 100 reads.
                        note: the think time will comsume cpu cycles, not sleep.
  --check               verify if the value of the byte at offet i is i%256. Valid only for the single
    url mode.
 -DSparmname stringvalue
                        set the internal parm <parmname> with the string value <stringvalue>
                         See XrdClientConst.hh for a list of parameters.
 -DIparmname intvalue
                        set the internal parm <parmname> with the integer value <intvalue>
                         See XrdClientConst.hh for a list of parameters.
                         Examples: -DSSocks4Server 123.345.567.8 -DISocks4Port 8080 -DIDebugLevel 1
```

Playing with remote data

- Execute the testload testrandom.txt
  - (true data access from an ATLAS job)
  - With TestXrdClient_read
  - Using the naif synchronous reads
    - vectored_style set to 0
    - Cache set to 0
    - Read ahead set to 0
    - Which is what is officially used up to now
    - 10ms of “think time” every 100 reads
    - Try at least 5 times, pick the best result, document it
    - Estimate the average total latency per request
    - Estimate the average CPU/wall time measure

Playing with remote data

- Execute the testload testrandom.txt
  - (true data access from an ATLAS job)
  - With TestXrdClient_read
  - Using the "Average window" readahead
    - vectored_style set to 0
    - Add "-DIReadAheadStrategy 2" to enable it
  - 10ms of "think time" every 100 requests
- Sparse, sequential load (easy case)
  - Try (at least 3-5 times each, pick the best result):
    - Cache sizes: 30000000(30M) up to 100000000(100M)
    - Read ahead size: from cache/10 to cache*3/4
  - Document the results, find your preferred option and explain why you think it's better
  - Estimate the average CPU/wall time measure

Playing with remote data

- Execute the testload testrandom.txt
  - (true data access from an ATLAS job)
  - With TestXrdClient_read
  - 10ms of “think time” every 100 requests
  - Using the “async readv” technique for sparse loads
    - Try at least 3-5 times, pick the best result
  - Compare the results with the previous runs