XRootd Monitoring Discussion

XRootD Workshop @ JSI Ljubljana

March 31, 2023

Matevž Tadel, UCSD

Overview

- About monitoring in general
- Monitoring sources / streams available in XRootd
  - Summary monitoring
  - Detailed monitoring
  - g-stream
- Demos: Fabio, Ilija
- Discussion

Introduction

On monitoring in general ...

- Monitoring means a lot of things:
  - Developer→sys admin / monitoring analyser→manager→reporter to funding agencies
  - Service view vs. VO vs. network
  - Where / how it is collected – job / server / work-load command service or agent
- What XRootd internal monitoring does is provide information for developer / user / monitoring analyser on the level of XRootd service.
  - We provide data that are best (or only possible) to be collected in the running program.
  - Next level integration is required for manager / VO reports
    - Collect data for several instances / sites
    - Correlate with workload management information

Monitoring perspectives I.

- XRootd perspective – Are your servers configured and used right?
  - XRootD daemon performance
    - Memory, CPU, storage & network usage; threads, data buffers, number of connections
  - Cluster performance:
    - File lookup & Redirection to “right” servers
  - We care because you would complain otherwise → when you do, it helps us understand what the actual problem is.

- Site / operator perspective – Are things running smoothly
  - all servers up and reasonably balanced, users are able to get data out
  - is xrootd usage within expected parameters
  - are users doing bad™ things

Monitoring perspectives II.

- VO / Data-federation perspective
  - Is the site working for us? ➞ authentication & storage access configuration
  - Is site & federation performance acceptable:
    - Lookup, redirection, and file open rate
    - Read / request rate – global and per connection
      - Impacts CPU efficiency for remote reading
    - Storage and WAN throughput & latency
  - How suitable the application is for remote access
  - Accounting that can be correlated with centrally controlled activities.

⇒ Management plots

⇒ Plots for funding agencies

Built-in Monitoring from 30kft

- Report what XRootD processes are doing
  - on the level of a whole process →**Summary monitoring**
  - on the level of individual user session / open file →**Detailed monitoring**
    - Includes also redirection & staging events
  - events / records of stuff users are actually interested in → **g-stream**
- All are sent as UDP packages to up to two destinations
- Implemented so as to have minimal impact on servers.
- Detailed monitoring is somewhat stateful (packet loss is a big problem).
- Ideally, collectors should run “close” to servers.

Summary monitoring

Periodic reports from xrootd and cmsd daemons in XML format

xrd.report dest1[,dest2] [every rsec] [-]option

option: all | buff | info | link | poll | process | prot[ocols] | sched | sgen | sync | syncwp [[-]option]

E.g., xrd.report xrootd.t2.ucsd.edu:9931 every 30s all sync

Summary record

```
<statistics tod="1421698118" ver="v3.3.5" src="cabinet-8-8-6.t2.ucsd.edu:1094" tos="1418409578"
pgm="xrootd" ins="anon" pid="3541" site="T2_US_UCSD"><stats
id="info"><host>cabinet-8-8-6.t2.ucsd.edu</host><port>1094</port><name>anon</name></stats><stats
id="buff"><reqs>110624</reqs><mem>176465920</mem><buffs>358</buffs><adj>0</adj></stats><stats
id="link"><num>1</num><maxn>122</maxn><tot>5301</tot><in>526680393</in><out>1749220925590</out><c
time>36508960</ctime><tmo>249066</tmo><stall>3</stall><sfps>0</sfps></stats><stats
id="poll"><att>1</att><en>249066</en><ev>249072</ev><int>0</int></stats><stats
id="proc"><usr><s>11863</s><u>39543</u></usr><sys><s>5465</s><u>697087</u></sys></stats><stats
id="xrootd"><num>4680</num><ops><open>55092</open><rf>0</rf><rd>21972049</rd><pr>0</pr><rv>137063
</rv><rs>9095834</rs><wr>0</wr><sync>0</sync><getf>0</getf><putf>0</putf><misc>61578</misc></ops>
<aio><num>0</num><max>44</max><rej>41</rej></aio><err>17690</err><rdr>0</rdr><dly>0</dly><lgn><nu
m>4679</num><af>3</af><au>4673</au><ua>0</ua></lgn></stats><stats
id="ofs"><role>server</role><opr>1</opr><opw>0</opw><opp>0</opp><ups>0</ups><han>1</han><rdr>0</r
dr><bxq>0</bxq><rep>0</rep><err>0</err><dly>0</dly><sok>0</sok><ser>0</ser><tpc><grnt>0</grnt><de
ny>0</deny><err>0</err><exp>0</exp></tpc></stats><stats
id="sched"><jobs>831528</jobs><inq>0</inq><maxinq>6</maxinq><threads>48</threads><idle>45</idle><
tcr>115</tcr><tde>67</tde><tlimr>0</tlimr></stats><stats
id="sgen"><as>0</as><et>0</et><toe>1421698118</toe></stats></statistics>
```

Summary contents

- For xrootd & cmsd

info:	name, port, host
link:	in/out transfers, # of connections, …
proc:	sys and user cpu usage
sched:	total / used threads, max task queue length
sgen:	time needed for generation of the report

- For xrootd

buff:	data buffer number, total size, # of requests
ofs:	files-system level operation counts
oss:	list of used paths / configured spaces + free space
poll:	# of polling operations / events
xrootd:	# of different operations, logins on protocol level

- For cmsd, mostly relevant for manager cmsds

cmsm:	per server statistics of redirections, responses, …

Detailed monitoring

- Inspect in detail what servers are doing
  - xrootd process only
  - redirectors can also report every redirection
- “Standard operations”:
  - Session begin / end, authentication, user info (u-stream with authentication info)
  - Open (d-stream – map session id to filename) / close a file
  - Reporting of read / write events:
    - report totals on file close
    - periodic “progress” updates (f-stream)
    - individual requests (t-stream with io option)
    - unpacked vector read requests (t-stream with iov option)
- Caveats about purpose
  - Collecting and archiving this information gets hard in a large, non-uniform federation.
  - Somebody has to analyze this information and stay on top of it.

Purpose of detailed monitoring

- Detailed accounting
    - When, who, from where, how long and how much
    - Every access is reported.
- Analyze data-access patterns
    - Improve application access to data
    - Tune parameters for a prefetching proxy-cache
- Misuse and abuse detection

Configuring detailed monitoring

```
xrootd.monitor [ options ] dest [ dest ]

 options:[all] [auth] [flush [io] intvl[m|s|h]] [fstat intvl[m|s|h] [lfn] [ops] [ssq] [xfr cnt]]

           [ident sec] [mbuff size[k] [rbuff size[k]] [rnums cnt] [window intvl[m|s|h]]

 dest:dest events host:port

 events:    [files] [fstat] [io[v]] [info] [redir] [user]
```

E.g., at UCSD for CMS:

```
xrootd.monitor all auth flush io 60s ident 5m mbuff 8k

                   rbuff 4k rnums 3 window 10s

                   dest files io info user redir xrootd.t2.ucsd.edu:9930

                   dest files iov info user      xrootd.t2.ucsd.edu:9932
```

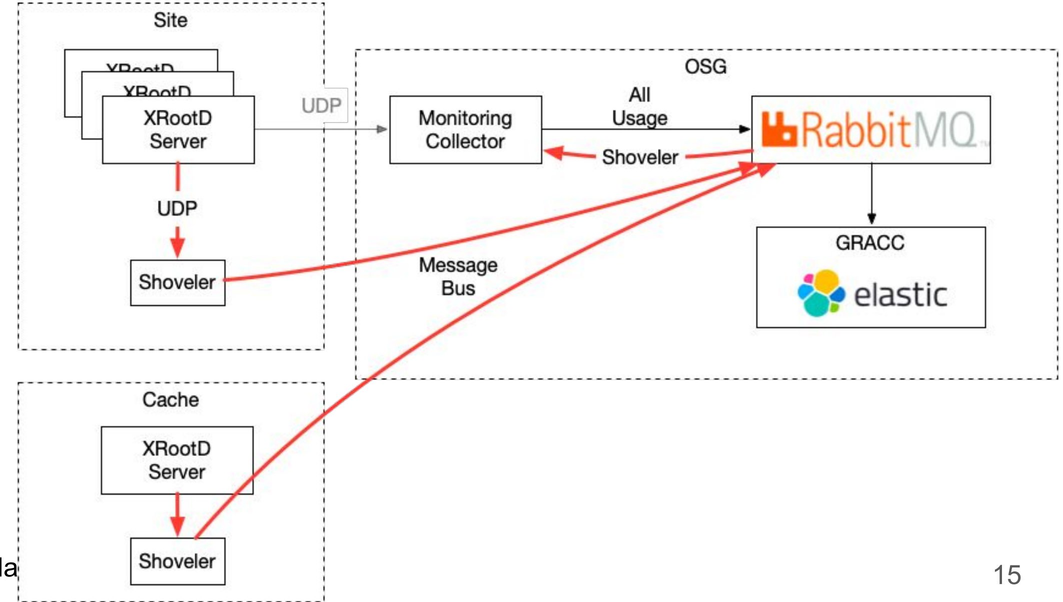
g-stream – JSON formatted records people want

```
xrootd.mongstream events use parms
events:   {all | ccm | pfc | tcpmon | tpc} [events]
parms:    [flush intvl[m|s|h]] [maxlen size[k]] [send fmt [noident] host:port]
fmt:      {cgi | json} [hdr] | nohdr
hdr:      dflthdr | sitehdr | hosthdr | insthdr | fullhdr
```

Event	**Explanation**
all	Selection applies to all of the events below.
ccm	cache context management information.
pfc	proxy file cache information (i.e. proxy disk caching).
tcpmon	**TCP** connection statistics at time of socket close.
tpc	Third party **c**opy for **http** and **xroot** protocols.

Shovelers, Collectors – the present

- OSG Shoveler: collect UDP and transfer out over guaranteed MQ protocol
  - UDP packet loss, esp. for larger UDP packets (for a very sad definition of large, 1440)
  - This runs close to the servers, out of the above danger
- OSG Collector: collects detailed monitoring and g-stream (summary?)
  - munches, aggregates, reports on to:
    - CERN ?? via AMQ
    - to OSG ES via RabbitMQ
  - not sure if it supports **t-stream io** and/or **iov**



Shovelers, Collectors – the past

- Summary monitoring:
  - mpxstats distributed with XRootd
  - xrd-rep-snatcher developed for AAA (perl): • Normalize input • Domain -> site name (obsoleted with all.sitename) • Calculate rates on relevant fields
- Detailed monitoring
  - XrdMon collector in Gled
    - used to run in two instances for whole AAA; maybe still run at CERN?
    - forwards *file-access-records* through AMQ
    - can store super detailed access records into ROOT files (fully unpacked **iov**)
      - Analysis code for those AnXrdMon
    - recently rebuilt with root-6 on Fedora 35 – so still alive.

Demos

Discussion …

-