CERN

Networking at the WLCG: R&D activities and Data Challenge 2024 testing

CERN
IT Department CS Group

International Symposium on Grids and Clouds (ISGC24)
24th - 29th March 2024

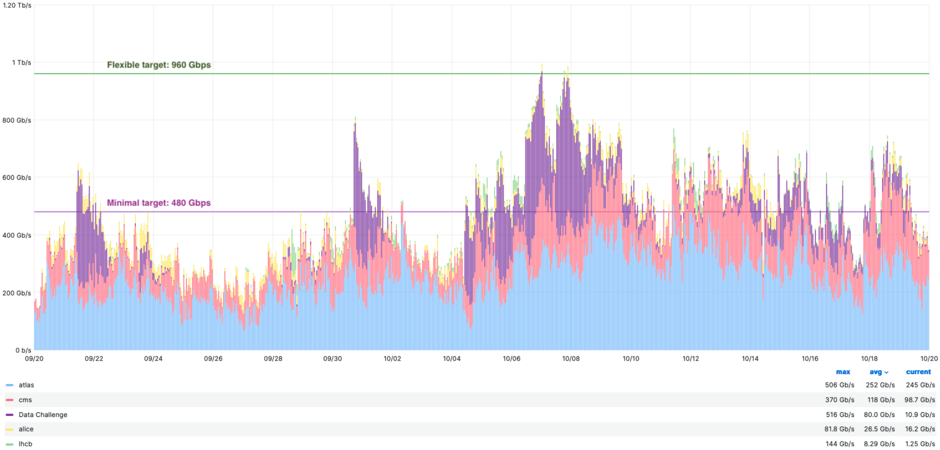
Carmen Misa Moreira
Edoardo Martelli

Outline

Introduction

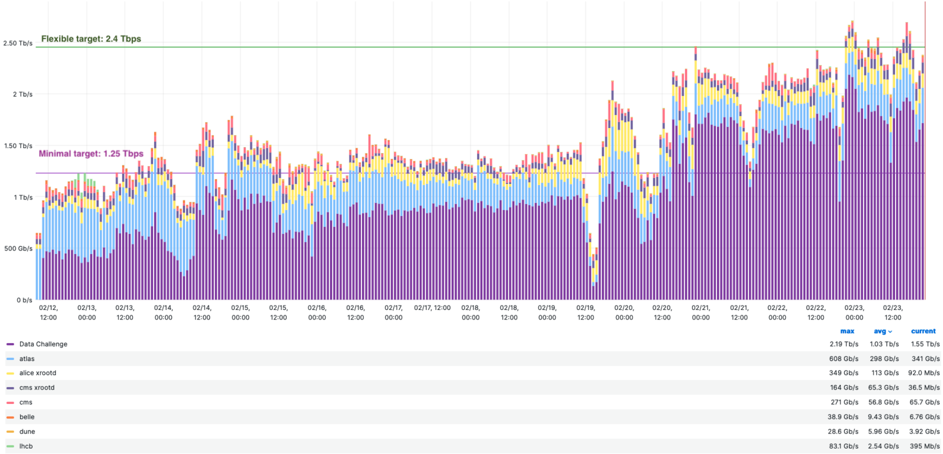
- Objectives of WLCG Data Challenge:
  - Demonstrate readiness for expected HL-LHC data rates by 2029
  - A data challenge roughly every 2 years
- Target goals:
  - WLCG DC21: 10% rate of HL-LHC
  - WLCG DC24: 25% rate of HL-LHC
- Lots of efforts on coordinating the data challenges across multiple experiments in terms of design, procedures, monitoring, and injection

WLCG Data Challenge 2021



- Successfully reached the minimal and flexible target models

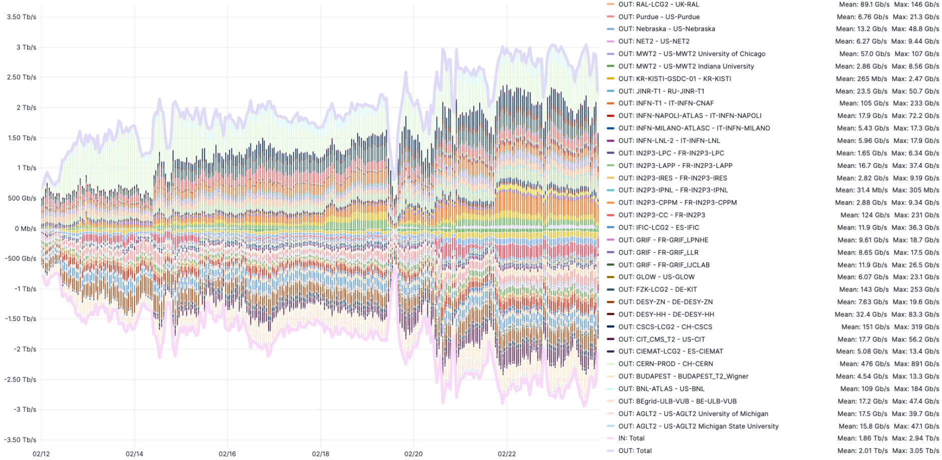
WLCG Data Challenge 2024



- Global throughput rates were achieved
- Reached 2.5 Tbps for ∼9 hours
- Minimal model: Tier 0 export, Tier 1's to Tier 1's, Tier 1's to associated Tier 2's
- Flexible model: full mesh everywhere from everywhere

Outcomes of WLCG DC24

Network performance



- The backbone network exhibit great performance. Some sites had the LHCOPN link down but had backup links in place
- No congestion on the network, peak at 3 Tbps
- The bottlenecks were mostly due to storage configuration or storage hardware limitations.

WLCG DC24 was a major success...

At the network level:

- It is a very useful exercise to find bottlenecks within sites
- Stress tests impacted on the network sites and overloaded storage endpoints

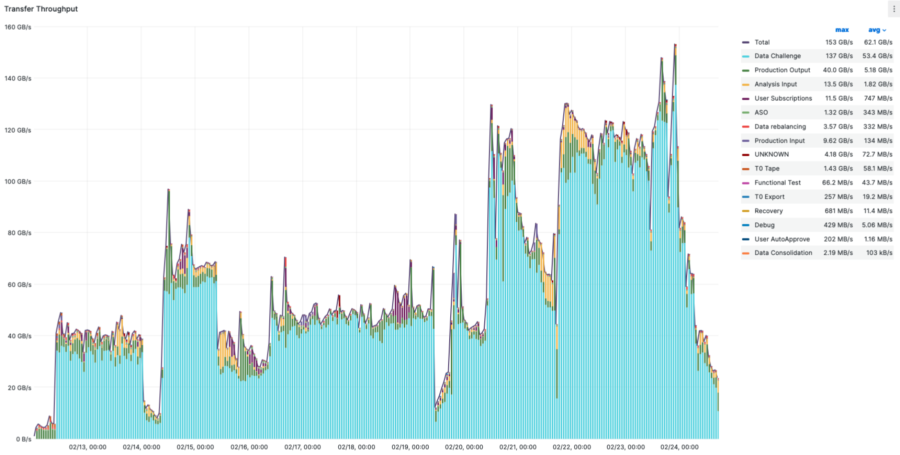
At the application level:

- Test scalability and push services to extreme rates above their normal operation
- FTS ran at double of its normal transfer rate
- Rucio proved to be able to scale and meet demands of DC24

WLCG DC24 was a major success...

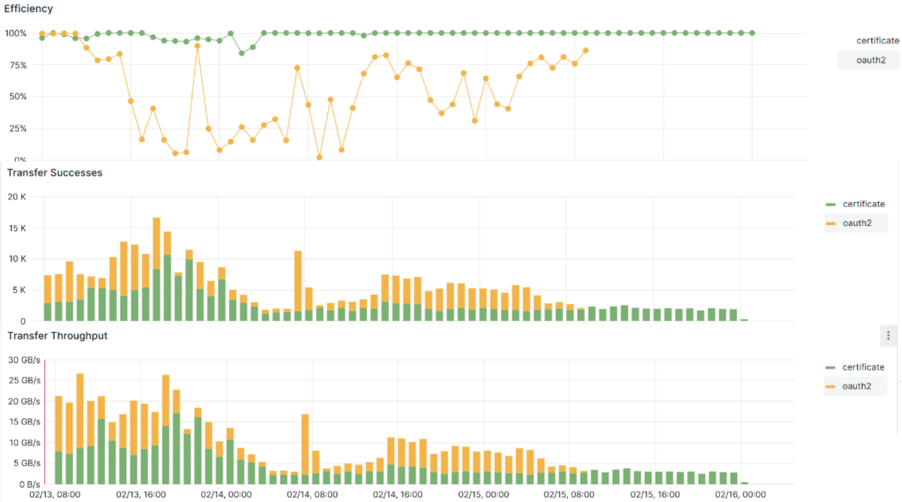
- FTS went above and beyond its usual 10K concurrent transfers per instance
- fts3-atlas.cern.ch sustained over 20K transfers for 17 hours
- DB RAM increased from 80 GB to 120 GB

WLCG DC24 was a major success…



- CMS experiment transfer throughput
- FTS was limiting the number of parallel transfers (fixed by changing FTS config)
- Rate was not achieved immediately because the injector tool used small files as input (fixed by changing configuration)
- February 21^{st}: Rucio couldn’t handle deletions due to large backlog

WLCG DC24 was a major success…



- Efficiency of token-based transfers are much lower compared to certificate-based

...but also useful for troubleshooting!

Encountered issues:

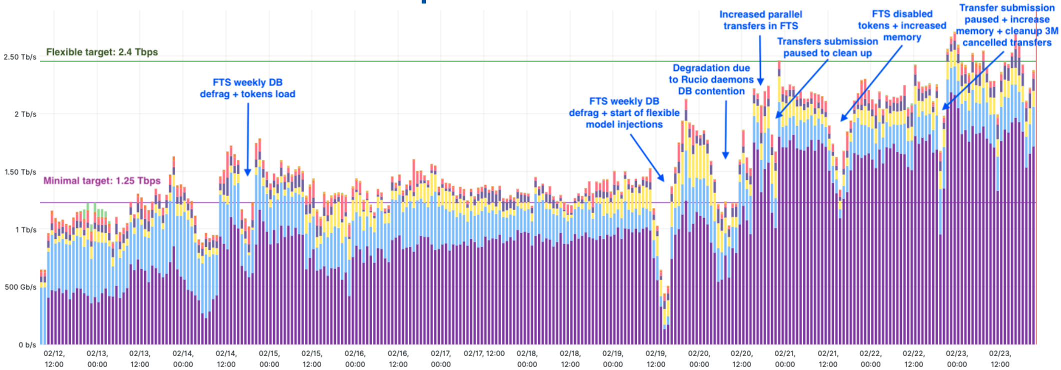
- Transfer submission to FTS: ATLAS reported poor submission performance. The top of the queue was dominated by transfer requests of expired replication rules
- Handling of expired replication rules: ATLAS reported inability to delete expired replication rules on large datasets in their early stages of replication
- Deletion overlap on slower sites: CMS reported poor deletion performance at some sites
  - Underlying issue is the rate of deletion at the sites themselves. But a design in Rucio does not handle this well, leading to multiple threads working on the same files
  - This hinders performance since affected dataset reuse and storage occupancy

...but also useful for troubleshooting!

More issues:

- Poor performance of the IAM server that affected all sites using tokens. Since every transfer require at least 2 tokens, submission rate dropped (∼0.5 Hz on average)
- Single-use refresh-tokens were discovered on the fly (fixed by IAM config change)
- Token refreshment: FTS is supposed to renew storage tokens before transfer starts if the lifetime left is short. 10h tokens were refreshed into 1h tokens (fixed by IAM config change)
- Once a StoRM WebDAV endpoint becomes overloaded and threads saturated, transfers fail... they are not queued or delayed. The more transfers are submitted, the worse it gets
- Monitoring inconsistency: FTS ipver bug

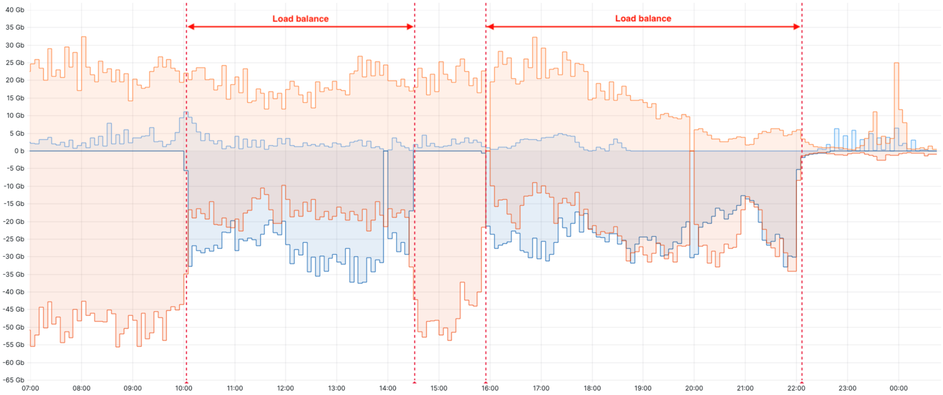
WLCG DC24 retrospective



	max	avg	current
Data Challenge	2.19 Tb/s	1.03 Tb/s	1.55 Tb/s
atlas	608 Gb/s	298 Gb/s	341 Gb/s
alice xrootd	349 Gb/s	113 Gb/s	92.0 Mb/s
cms xrootd	164 Gb/s	65.3 Gb/s	36.5 Mb/s
cms	271 Gb/s	56.8 Gb/s	65.7 Gb/s
belle	38.9 Gb/s	9.43 Gb/s	6.76 Gb/s
dune	28.6 Gb/s	5.96 Gb/s	3.92 Gb/s
lhcb	83.1 Gb/s	2.54 Gb/s	395 Mb/s

R&D activities

NOTED (Network Optimised Transfer of Experimental Data)



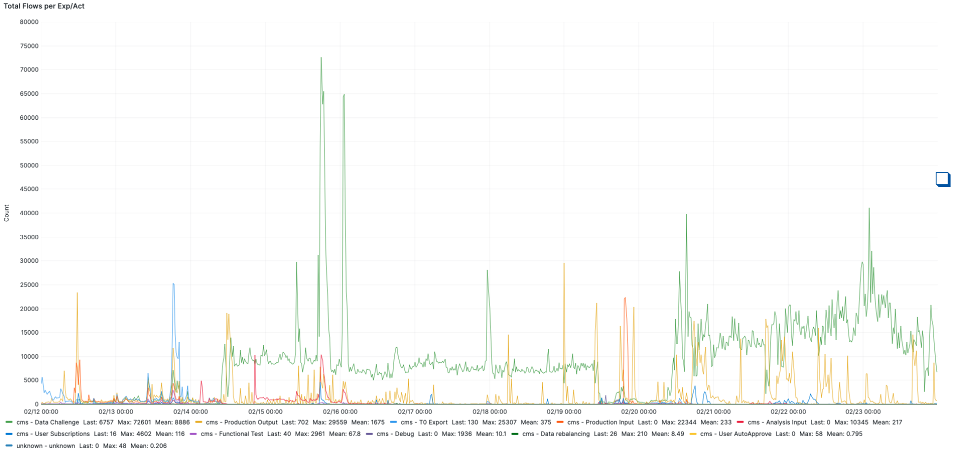
Name	Min	Max	Mean	Last *	Total
Backup Link Incoming to CERN	0 b	11.1 Gb	1.89 Gb	966 kb	406 Gb
Backup Link Outgoing from CERN	0 b	37.5 Gb	14.9 Gb	307 b	3.21 Tb
Primary Link Incoming CA-TRIUMF to CERN	0 b	32.4 Gb	15.5 Gb	437 Mb	3.33 Tb
Primary Link Outgoing CERN to CA-TRIUMF	0 b	55.6 Gb	24.3 Gb	962 Mb	5.22 Tb

- CA-TRIUMF load balancing between LHCOPN and its backup link (from 21st to 23rd of February 2024)

RNTWG (Research Networking Technical Working Group) [link]

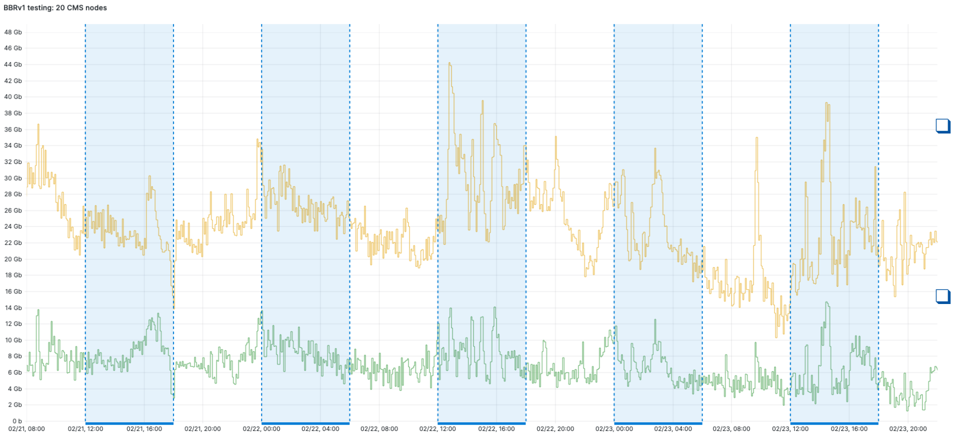
- R&D activities in network technologies in the areas of network visibility (packet marking), throughput (packet pacing) and SDN (orchestration) to better understand how the network flows perform along the path
  - Improve visibility into how network flows perform
  - Get insights into how experiments are using the networks and get additional data on the behaviour of transfers (traffic, paths, etc.)
- Network monitoring per flow: experiment and activity information
  - Packet marking: 20-bit flow label field of IPv6 header.
  - Flow marking: UDP firefly

Flow marking (UDP fireflies)



- During WLCG DC24 the UDP fireflies were deployed in 80% of the CMS infrastructure

Packet pacing: BBR (Bottleneck Bandwidth and Round-Trip Time)



Name	Last *	Min	Max	Mean
interface_if_octets.sum {value_instance: rx}	5.12 Gb	308 Mb	23.4 Gb	5.19 Gb
interface_if_octets.sum {value_instance: tx}	17.5 Gb	7.08 Gb	53.3 Gb	19.5 Gb

- During WLCG DC24 20 CMS nodes swapped every 6 hours between CUBIC and BBRv1 congestion protocol
- No evidence of gain nor loss using BBRv1

Packet pacing: BBR (Bottleneck Bandwidth and Round-Trip Time)

- During WLCG DC24 23 ATLAS nodes swapped every 6 hours between CUBIC and BBRv1 congestion protocol
- No evidence of gain nor loss using BBRv1

Conclusions

Conclusions

- Aim of WLCG DC24 is not only to achieve rates → find bottlenecks and issues
- Many lessons learned, now we understand better our infrastructure:
  - Transfer protocols: http doesn't have threads like gridftp used to have
  - Size of files affects the number of FTS requests and thus the achieved rate
  - The FTS weekly defragmentation of the database blocked the transfers twice
  - Cancelled jobs were accumulating in the DB making it unresponsive
  - Right now the only way to scale FTS is to add more memory: increased on fts3-atlas
  - Had to install a second high memory instance on fts3-pilot and move all the Tier 2's to the second instance to achieve the necessary rates
  - Token authentication deployed in 25 sites: switched off to achieve throughput
  - The FTS optimizer cycle eventually was taking 3 hours and couldn't be restored to a working state easily

Thanks for your attention!