



Data flowing on the Stream

High throughput EOS instance for ALICE O2 running on CentOS Stream

Cristian Contescu & Andreas Joachim Peters on behalf of the CERN EOS team

EOS Workshop – March 2022

The O² storage system design

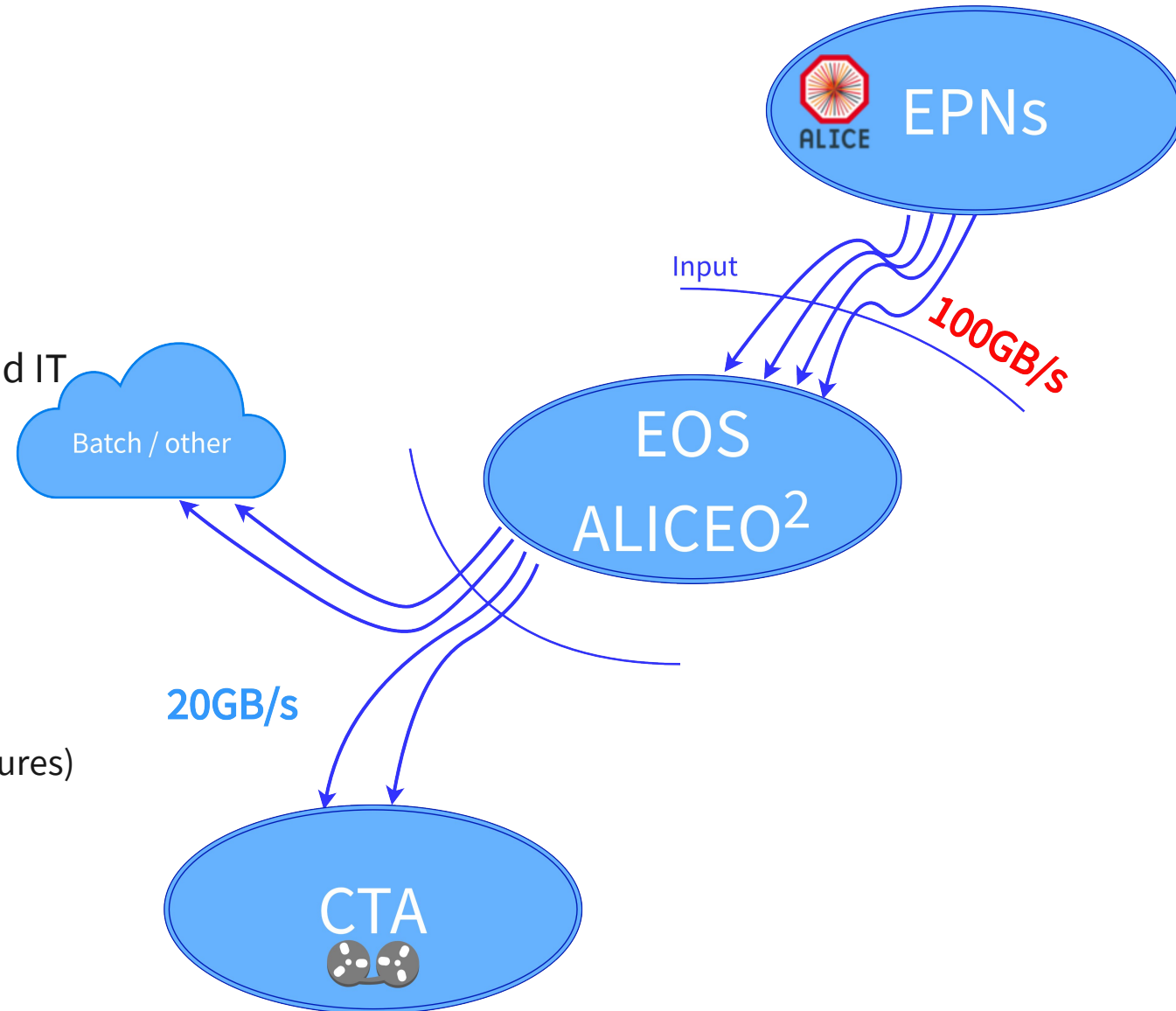
The setup in brief

- **ALICE: 250 Event Processing Nodes (EPNs)**

- IB \Rightarrow ETH gateways \Rightarrow EOSALICEO2
- 1.2 Tbps redundant links between ALICE GWs and IT

- **IT: 74 disk servers**

- **Theory: homogeneous hardware**
- **Practice: 3 types of hardware**
 - Intel based servers:
 - 10x nodes with 96 disks
 - 16x nodes with 60 disks (high-density enclosures)
 - AMD based servers:
 - 48x nodes with 96 disks
- ~6500 rotational disks (14TB each)
- 100GE network technology



Big bang (2020): How did we get here?

ALICE O²: The Heavy-Ion run

- **ALICE setup: 250 EPNs (Event Processing Node), each with 8 GPUs, each GPU generating a Compressed Time Frame every 20 seconds**
 - 2000 data sources in total
- **A Compressed Time Frame (CTF) will correspond to a single 1GB file**
 - CTF has to be copied to EOS in less than 20 seconds
- **Data sources transfer data to EOS in (a kind of) round robin fashion at 10 ms intervals**
 - every 10 ms a new file will be created and 1GB of data will start to be transferred
- **EOS expected data intake:**
 - $\frac{2000 \text{ files} \times 1\text{GB}}{20\text{s}} = 100\text{GB} / \text{s}$

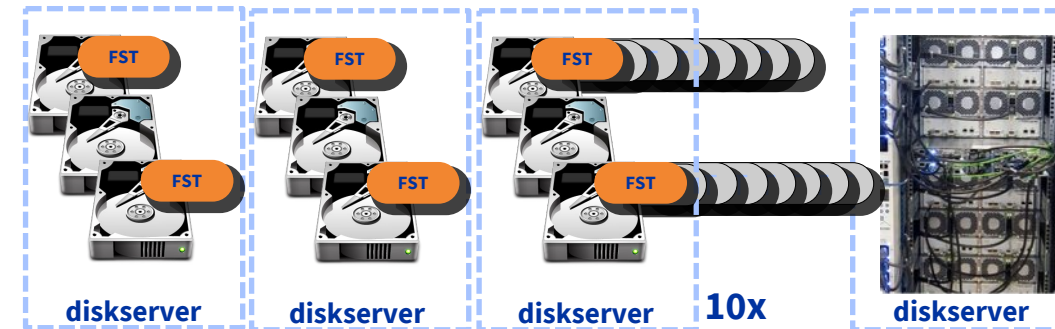
ALICE O²: The Heavy-Ion run

- **ALICE setup: 250 EPNs (Event Processing Node), each with 8 GPUs, each GPU generating a Compressed Time Frame every 20 seconds**
 - 2000 data sources in total
- **A Compressed Time Frame (CTF) will correspond to a single 1GB file**
 - CTF has to be copied to EOS in less than 20 seconds
- **Data sources transfer data to EOS in (a kind of) round robin fashion at 10 ms intervals**
 - every 10 ms a new file will be created and 1GB of data will start to be transferred
- **EOS expected data intake:**
 - $\frac{2000 \text{ files} \times 1\text{GB}}{20\text{s}} = 100\text{GB / s}$

2020: Initial cluster, OS Tweaks, Network infrastructure and node-to-node tests

Initial R&D cluster

- **3x head nodes running in high availability mode**
- **10x storage nodes:**
 - Intel-based (Cascade Lake)
 - 4x disk trays (24 disks per tray)
 - => 96x disks (14TB) - 1344TB
 - 100Gbps NICs => 1Tbps total throughput => ~120GB /s
 - 12GB/s throughput from the disk controllers
 - disks definitely able to saturate this bandwidth => 120GB / s
 - Each storage node running multiple FST services
 - 12 disks per service



OS & Network tweaks, CPU governor & I/O schedulers

Following and adapting TCP tweak instructions for 40Gbps+ links from:

- <https://fasterdata.es.net/host-tuning/linux/>
- <https://www.es.net/assets/Uploads/100G-Tuning-TechEx2016.tierney.pdf>

```
net.ipv4.tcp_rmem = 4096 87380 2147483647 # socket size
net.ipv4.tcp_wmem = 4096 65536 2147483647 # socket size
net.core.rmem_max = 2147483647           # window size
net.core.wmem_max = 2147483647           # window size
net.core.default_qdisc = fq              # queue discipline
net.ipv4.tcp_congestion_control = htcp    #| bbr (for C8)
```

```
Ring parameters for enp59s0:
Pre-set maximums:
RX:                               8192
RX Mini:      n/a
RX Jumbo:     n/a
TX:                               8192
Current hardware settings:
RX:                               8192
RX Mini:      n/a
RX Jumbo:     n/a
TX:                               8192
```

Tuned profile to set up CPU governor and I/O scheduler:

```
# cat /etc/tuned/active_profile
eos-diskserver

# cat /etc/tuned/eos-diskserver/tuned.conf

[main]
summary=EOS Diskserver (inherits throughput performance)
include=throughput-performance

[disk]
elevator=cfq #| bfq (for C8)
```

Network infrastructure and data redundancy

- **Switch uplinks: 500Gbps initially (2:1 blocking factor)**
 - 10x 100Gbps nodes connected to them
- **Final setup: 1:1 blocking factor**
- **Data redundancy is important**
 - this generates additional traffic
 - in case of replicated system: at least one additional write operation needed (doubling the network traffic)
 - in case of erasure coding: $EC(n+x)$
 - one storage node will act as gateway
 - receives the entire file & stores 1 chunk
 - fans out $n-1$ chunks (+ x parity chunks) to other storage nodes

Network infrastructure and data redundancy

- **Switch uplinks: 500Gbps initially (2:1 blocking factor)**
 - 10x 100Gbps nodes connected to them
- **Final setup: 1:1 blocking factor**
- **Data redundancy is important**
 - this generates additional traffic
 - in case of replicated system: at least one additional write operation needed (doubling the network traffic)
 - in case of erasure coding: $EC(n+x)$
 - one storage node will act as gateway
 - receives the entire file & stores 1 chunk
 - fans out $n-1$ chunks (+x parity chunks) to other storage nodes

Node-to-node tests & some initial hurdles

- **Initial setup: MGM and storage nodes running CERN CentOS 7**
- **Per node network performance: OK (~92Gbps == 11GB/s)**
- **Disk subsystem performance: not OK (~6GB/s) => oops, wrong SAS controllers => replaced => OK (12GB/s)**
- **Disk + Network together: only ~1/2 of the expected throughput, why?**
 - Two options:
 - Start debugging where the bottleneck may come from (possibly time consuming)
 - Trying a different kernel (main line kernel built by ELRepo; same kernel config as official RHEL one)

Node-to-node tests & some initial hurdles

- **Initial setup: MGM and storage nodes running CERN CentOS 7**
- **Per node network performance: OK (~92Gbps == 11GB/s)**
- **Disk subsystem performance: not OK (~6GB/s) => oops, wrong SAS controllers => replaced => OK (12GB/s)**
- **Disk + Network together: only ~1/2 of the expected throughput, why?**
 - Two options:
 - Start debugging where the bottleneck may come from (possibly time consuming)
 - Trying a different kernel (main line kernel built by ELRepo; same kernel config as official RHEL one)



Bingo!

2020: The road to CentOS 8

The road to CentOS 8

- **Running CC7 + Main Line Linux kernel => expected performance**
- **Nodes need to be accessible from other WLCG sites (external firewall opening)**
- **Security Team:**
 - Concerns regarding timely security patches when using 3rd party kernel => can't open external firewall ports
 - “How about you try CentOS 8 which is already supported?”
- **Planning to go CentOS 8:**
 - Couple of weeks of preparation
 - Adjusting Puppet profiles, getting familiar with new systems like firewalld (replacing iptables), etc.
 - Upgrading storage nodes one by one => no data availability issue, no downtime

CentOS 8: Some hurdles & tests

- **UEFI installs set the OS as the main boot (regardless of the initial BIOS setting to PXE boot)**
 - workaround for flipping the boot order to PXE when (re)installing nodes
 - also easier to change boot order from the OS command line with EFI enabled
- **Initial 'full-stack' (network+disks) tests rerun after the upgrade to CentOS 8**
 - performance slightly improved (~60% of nominal vs. ~50% of nominal)
 - but this time, performance tests clearly pointed towards the network subsystem
- **By process of elimination we've discovered where the problem might reside: the firewall**
 - with the firewall disabled, nominal network performance was achieved
 - Intriguing entry in firewalld.conf: **IPv6_rpfilter=yes**
 - Drilling down into the nftables rules generated by this option (and confirming with RedHat):

```
icmpv6 type { nd-router-advert, nd-neighbor-solicit } accept  
meta nfproto ipv6 fib saddr . iif oif missing log prefix "rpfilter_DROP: " drop
```

CentOS 8: Some hurdles & tests

- **UEFI installs set the OS as the main boot (regardless of the initial BIOS setting to PXE boot)**
 - workaround for flipping the boot order to PXE when (re)installing nodes
 - also easier to change boot order from the OS command line with EFI enabled
- **Initial 'full-stack' (network+disks) tests rerun after the upgrade to CentOS 8**
 - performance slightly improved (~60% of nominal vs. ~50% of nominal)
 - but this time, performance tests clearly pointed towards the network subsystem
- **By process of elimination we've discovered where the problem might reside: the firewall**
 - with the firewall disabled, nominal network performance was achieved
 - Intriguing entry in firewalld.conf: **IPv6_rpfilter=yes**
 - Drilling down into the nftables rules generated by this option (confirming with RedHat)

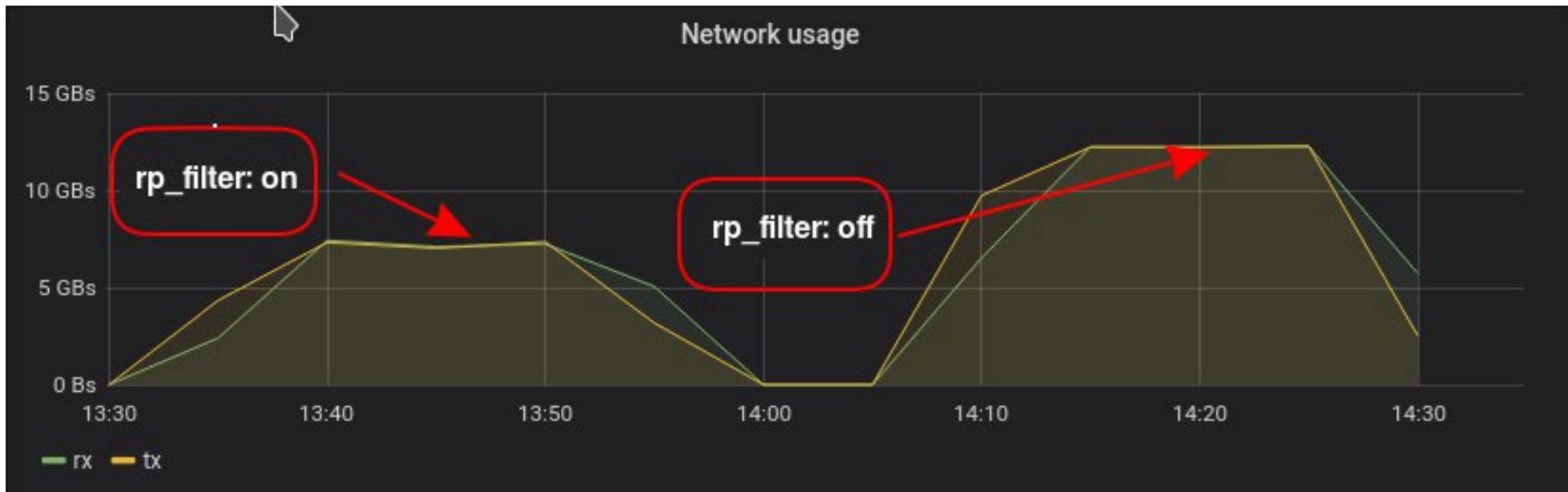
icmpv6 type { nd-router-advert, nd-neighbor-solicit } accept

meta nfproto ipv6 fib saddr . iif oif missing log prefix "rpfilter_DROP: " drop



The culprit

CentOS 8: understand the firewall performance penalty

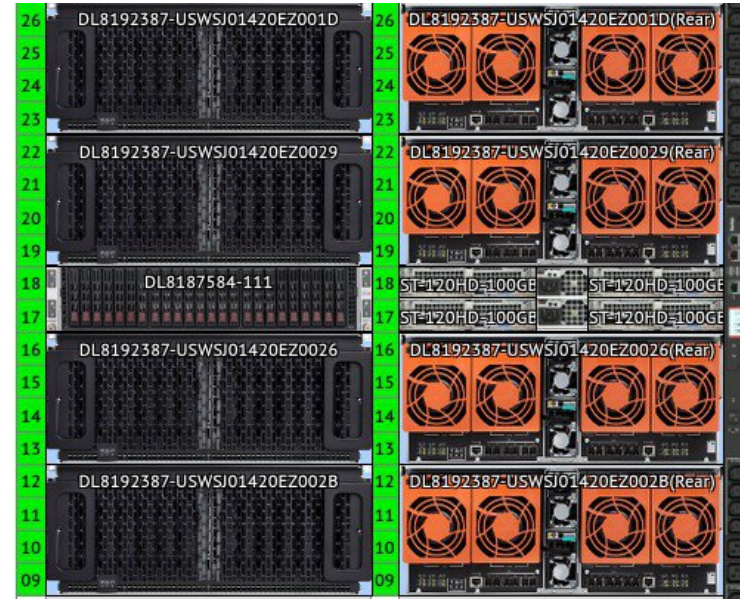


- Full performance restored
- Decided to keep IPv6 `rp_filter` off (not really justified when running a single interface / IP)
- In the meantime we also migrated from `firewalld` to straight `nftables` and this rule is not generated by default

Spring 2021: R&D cluster, new nodes

Spring 2021: Additional R&D Hardware

- **16x High Density JBOD**
 - Same front-end hardware
 - 2x trays (30 disks each) per node
 - dual-path connection to frontend
 - ~840 TB / node (14TB drives)
 - 100Gbit/s NICs
- **Cluster at 1/3 of capacity**
 - becomes less homogenous

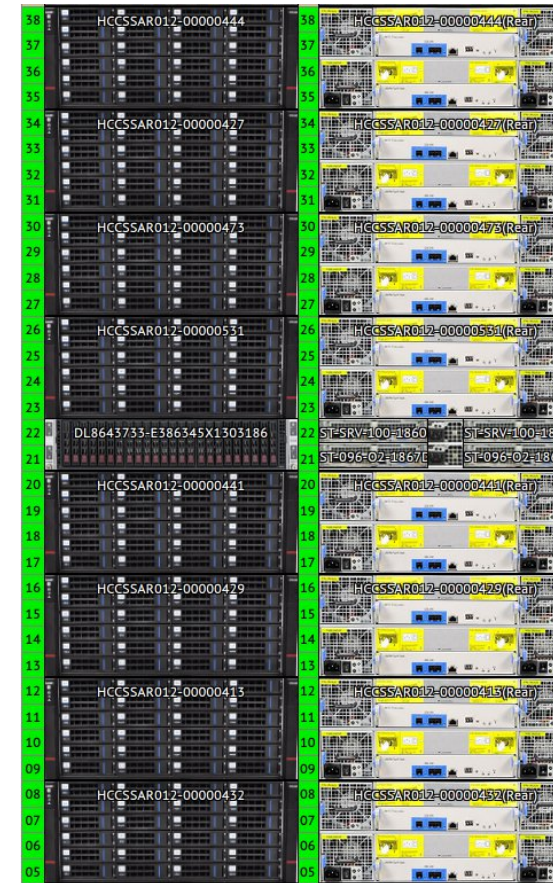


```
/etc/multipath.conf
...
defaults {
    user_friendly_names yes
    find_multipaths yes
    enable_foreign "^$"
    path_grouping_policy "multibus" # default: failover
    #default is "failover"
}
...
```

August/September 2021: Closer to the final setup & jumping into the Stream

New capacity: 48 new nodes added in the cluster

- **+64.5PB raw**
- **48x standard JBOD**
 - 4x trays each = 96 disks per system unit
 - 1344TB
 - 100Gbps Ethernet
 - AMD CPU architecture (EPYC 7302)
 - First in a while in our fleet



New capacity: Installation process

- **Prepared and installed nodes directly with CentOS Stream 8**
 - following the CentOS 8 EOL announcement in December 2020
 - Straightforward installation process
 - Few Kickstart tweaks needed (only because we use a custom Kickstart file)
- **Nodes auto-registered to the cluster and quickly brought up for production**
 - ALICE was expecting them for the Tape Data Challenge
 - But most importantly: for the detector and O2 software commissioning tests which took place in October 2021
- **‘Old’ R&D nodes remained in the production setup running CentOS8**

New capacity: some weird behavior

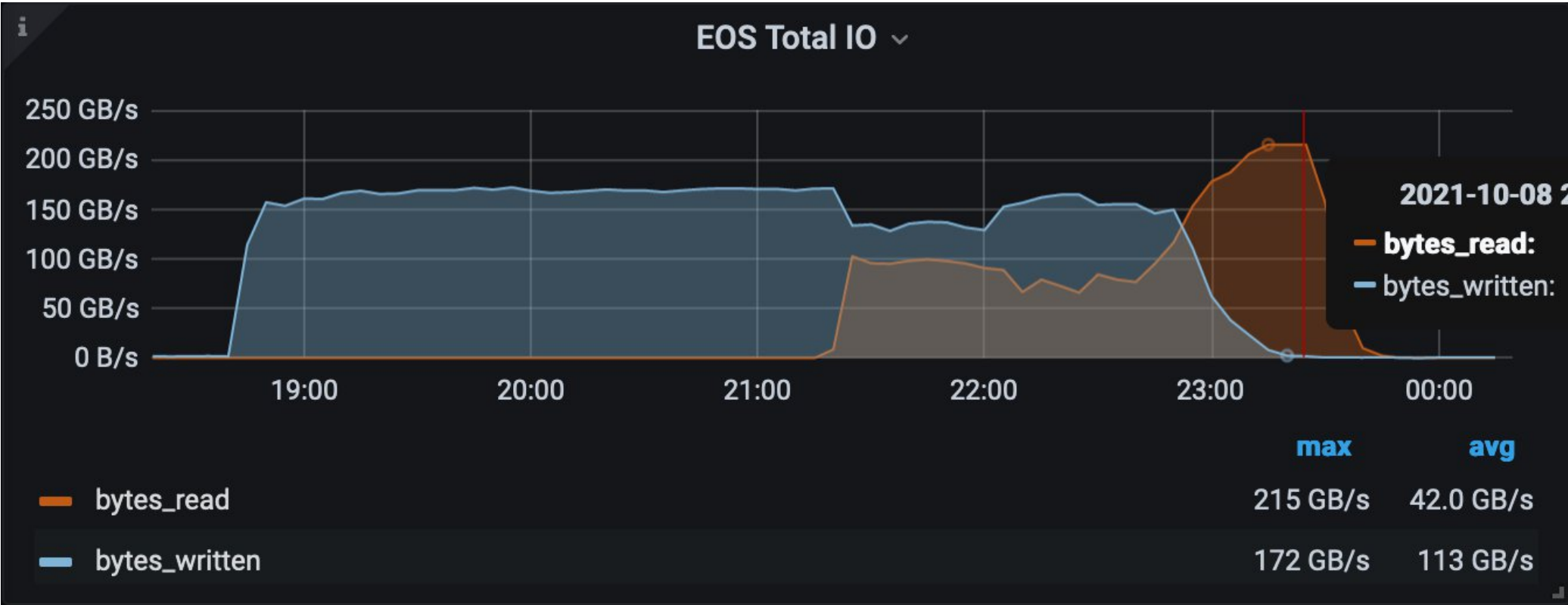
- **IT Performance tests using the full cluster have started**
- **Our monitoring system caught some weird behavior**
 - Some of the new nodes have started to display random system reboots
 - System and BMC logs => no interesting entry
- **Suspected OS issue and opened ticket with Linux support**
 - Checking the kdump: all nodes crashing with the same “NULL pointer dereference” signature
 - Bug plagued at least two older version of the kernel, so downgrading was not considered as a valid workaround:
 - 4.18.0-331.el8 and 4.18.0-338.el8
 - A patched kernel from Red Hat already existed and fixed the issue - tested and validated the solution
 - fix also present in CentOS Stream 8 since 4.18.0-348.el8 (RHEL 8.5 kernel)

```
[600606.592081] BUG: unable to handle kernel
NULL pointer dereference at 0000000000000120
[600606.593280] PGD 2009a4067 P4D 2009a4067
PUD 1be535067 PMD 0
[600606.594337] Oops: 0000 [#1] SMP NOPTI
[600606.595375] CPU: 10 PID: 854 Comm:
kworker/10:1H Kdump: loaded Not tainted 4.18.0-
338.el8.x86_64 #1
[600606.596417] Hardware name: Supermicro AS -
2124BT-HNTR/H12DST-B, BIOS 2.1 05/07/2021
[600606.597454] Workqueue: kblockd
blk_mq_timeout_work
[600606.598478] RIP:
0010:blk_mq_put_rq_ref+0x9/0x40
```

New capacity: Final verdict

- **The overall results (running with RS(12,10) [10+2]):**
 - 48x 'Clients' inside CERN IT (100GBE connectivity)
 - Average bandwidth write-only > **140GB/s**
 - bandwidth reading+writing (1:2 ratio) **100 GB/s** reading + **110 GB/s** writing concurrently
 - bandwidth read-only > **225 GB/s** (peaks at **248 GB/s**)

New capacity: Final verdict (II)



Wait, there is more...

- **Overall tests were good, but...**

- Individual node tests pointed to a bandwidth issue with the disk controllers
 - Getting only 4-6GB/s out of the expected 12GB/s
 - Investigating the PCIE link speeds of the controllers: all good this time
 - Escalated the issue to the HW Procurement and Repair Team

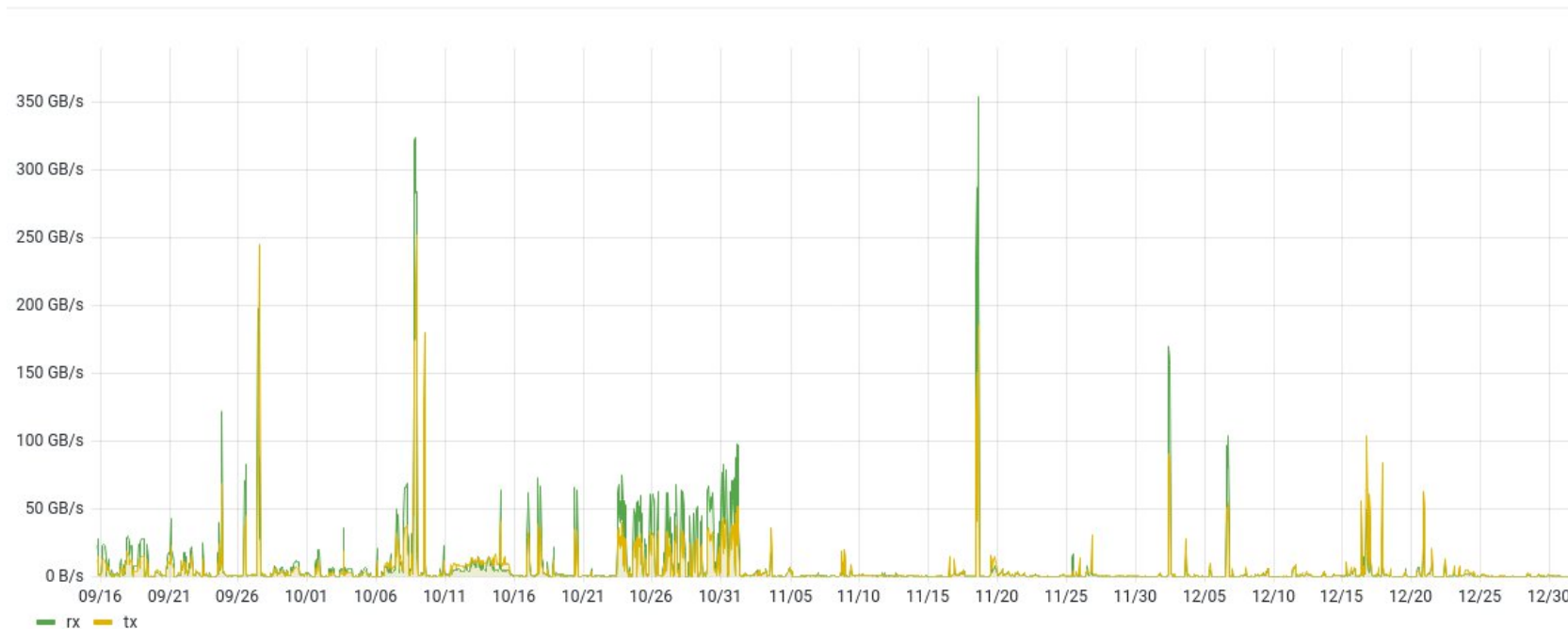
- **Meet the... BIOS settings**

- These are new AMD nodes, so were less familiar
- Procurement team inspected the BIOS settings
 - Found some suboptimal relevant tweaks
 - Ran a coordinated BIOS settings apply + reboot campaign

- * NUMA Per Socket => 1 (better BW handling)
- * Force Inter-Chip Global Memory Interconnect Link Width => 2 (for low latency)
- * PCIe 10-bit Tag => enabled (increased BW)
- * APIC Mode => x2APIC (better interrupts handling with high-cores count nodes)
- * Enhanced Preferred IO => Enabled (eases ordering of PCIe packets to reduce overhead)
- * APBDIS => 1 (Disable switching of P-States on the infinity interconnect)

More tests and taking data during LHC Pilot Beam

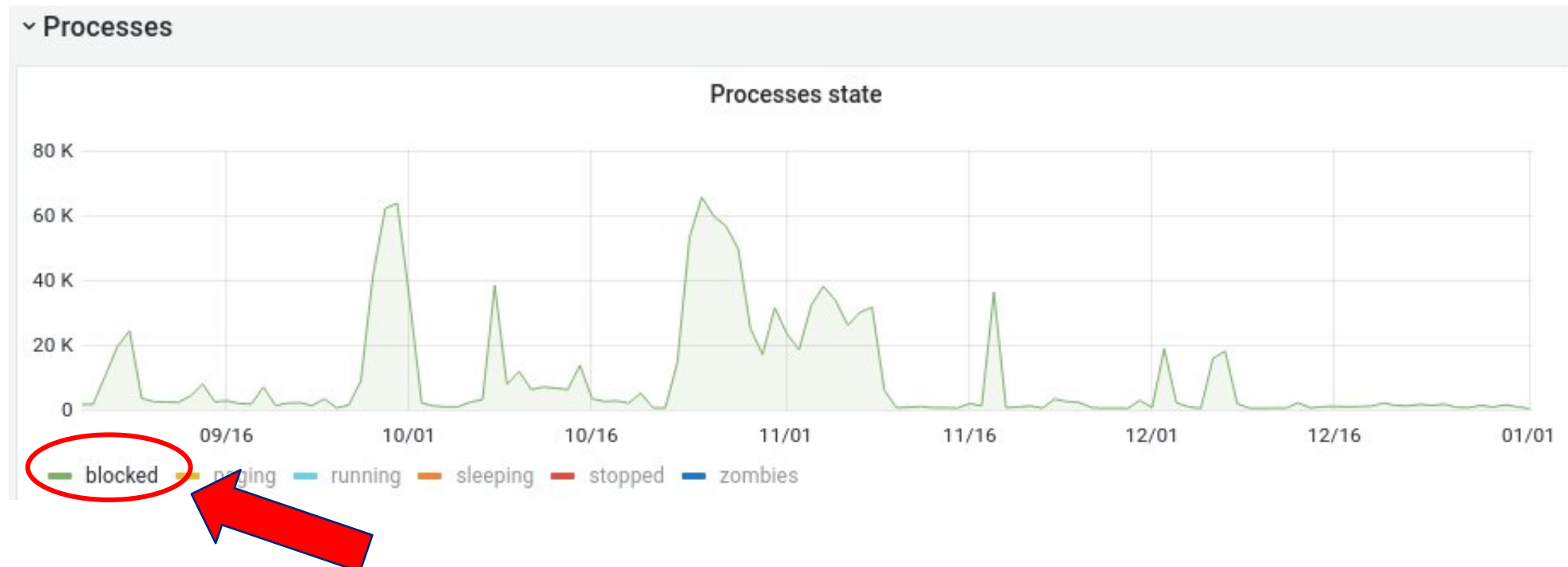
- Internal tests: ✓
- ALICE Pilot Beam data taking: ✓



Everything green...

- ...however, our monitoring system:

Node: st-096-o2-xxxxxx.cern.ch:1101 has a load of 1018.15, threshold: 70.
Node: st-096-o2-yyyyyy.cern.ch:1101 has a load of 425.29, threshold: 70.
Node: st-096-o2-zzzzzz.cern.ch:1101 has a load of 448.24, threshold: 70.



The 'D' state inferno...

- Inspecting the storage nodes, we spotted many processes getting stuck in IOWait (for a very long time)
 - FST threads blocked, but also, e.g. smartctl, puppet agent runs, critical for disk health checks and configuration deployment
 - node reboots needed to clean up
- All nodes exhibiting this behavior running: **4.18.0-348.el8** (EL 8.5)
- Nov 2021: opened ticket with Linux support => Red Hat support involved and analyzed Kdump
 - Issue deemed to be or show up in the BFQ I/O scheduler
 - Switched from BFQ to MQ-DEADLINE
 - problem gone, but
 - cluster performance lower (10-15%)
 - I/O priorities cannot be used (for e.g.: file scrubbing)

```
Nov 2 06:31:46 st-096-o2-184e66.cern.ch kernel: INFO: task xrootd:13150
blocked for more than 120 seconds.
Nov 2 06:31:46 st-096-o2-184e66.cern.ch kernel: Not tainted 4.18.0-
348.el8.x86_64 #1
Nov 2 06:31:46 st-096-o2-184e66.cern.ch kernel: "echo 0 >
/proc/sys/kernel/hung_task_timeout_secs" disables this message.
Nov 2 06:31:46 st-096-o2-184e66.cern.ch kernel: task:xrootd state:D
stack: 0 pid:13150 ppid: 1 flags:0x00004080
Nov 2 06:31:46 st-096-o2-184e66.cern.ch kernel: Call Trace:
Nov 2 06:31:46 st-096-o2-184e66.cern.ch kernel: __schedule+0x2c4/0x700
Nov 2 06:31:46 st-096-o2-184e66.cern.ch kernel: ?
__switch_to_asm+0x35/0x70
Nov 2 06:31:46 st-096-o2-184e66.cern.ch kernel: ?
__switch_to_asm+0x35/0x70
Nov 2 06:31:46 st-096-o2-184e66.cern.ch kernel: ?
xfs_buf_read_map+0x1b5/0x310 [xfs]
Nov 2 06:31:46 st-096-o2-184e66.cern.ch kernel: schedule+0x37/0xa0
Nov 2 06:31:46 st-096-o2-184e66.cern.ch kernel:
schedule_timeout+0x274/0x300
Nov 2 06:31:46 st-096-o2-184e66.cern.ch kernel: ? __schedule+0x2cc/0x700
Nov 2 06:31:46 st-096-o2-184e66.cern.ch kernel: ?
blk_finish_plug+0x21/0x2e
Nov 2 06:31:46 st-096-o2-184e66.cern.ch kernel: ?
__xfs_buf_submit+0x122/0x1d0 [xfs]
Nov 2 06:31:46 st-096-o2-184e66.cern.ch kernel: ?
xfs_buf_read_map+0x1b5/0x310 [xfs]
Nov 2 06:31:46 st-096-o2-184e66.cern.ch kernel:
wait_for_completion+0x97/0x100
Nov 2 06:31:46 st-096-o2-184e66.cern.ch kernel: ?
__xfs_buf_submit+0x122/0x1d0 [xfs]
Nov 2 06:31:46 st-096-o2-184e66.cern.ch kernel: xfs_buf_iowait+0x22/0xd0
[xfs]
Nov 2 06:31:46 st-096-o2-184e66.cern.ch kernel:
__xfs_buf_submit+0x122/0x1d0 [xfs]
...
```

The 'D' state inferno...

- Further info collected and provided to Red Hat
- Beginning of February 2022:
 - Patch created [1] and a test kernel provided (block/blk-mq.c):

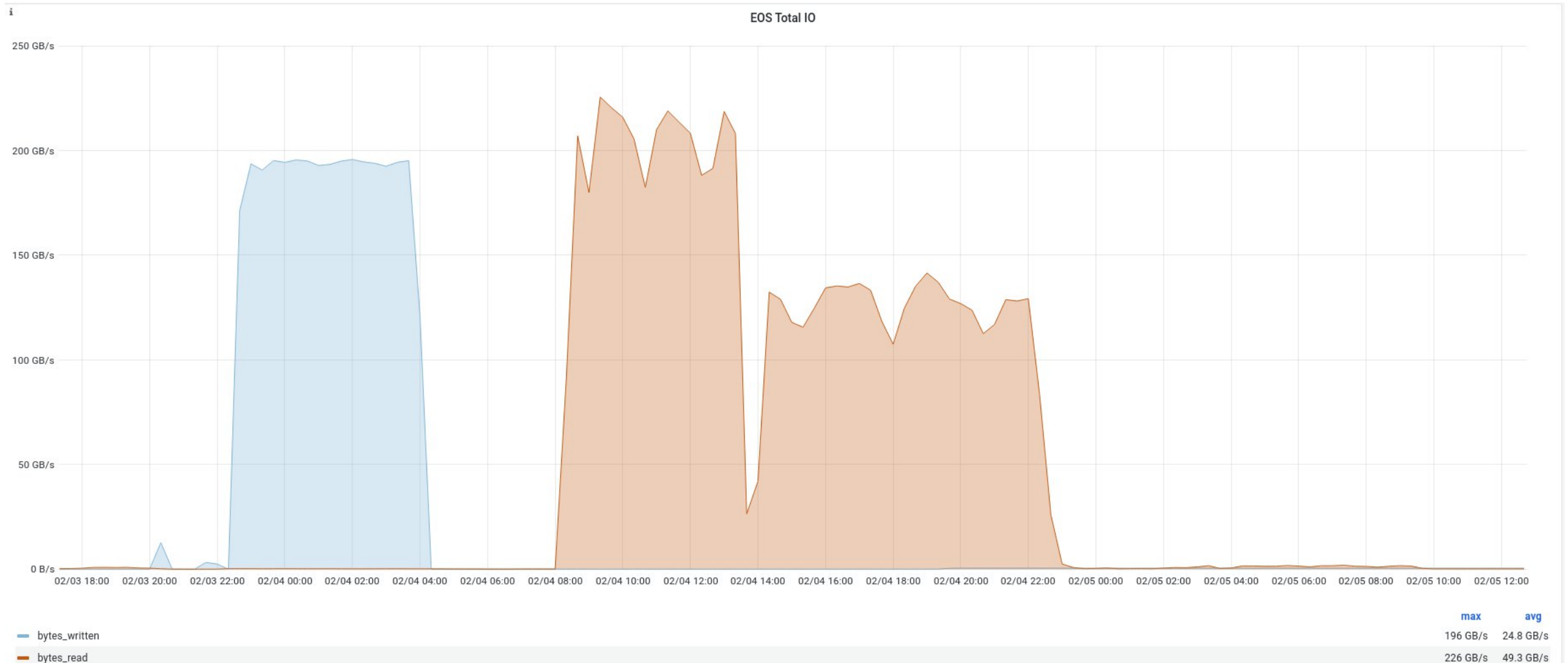
```
+ /*  
+  * If there is already a run_work pending, leave the  
+  * pending delay untouched. Otherwise, a hctx can stall  
+  * if another hctx is re-delaying the other's work  
+  * before the work executes.  
+  */  
+ if (delayed_work_pending(&hctx->run_work))  
+     continue;
```

- Experimental kernel tested: all tests pass
 - 'D'-state threads/processes stuck for a long time are gone
- Kernel **4.18.0-358.el8** also released for CentOS Stream 8 ~ at the same time
 - not containing the above fix, but
 - back porting the block subsystem from mainline kernel 5.14.x
 - issue is seemingly fixed or less likely to be encountered with this kernel as well

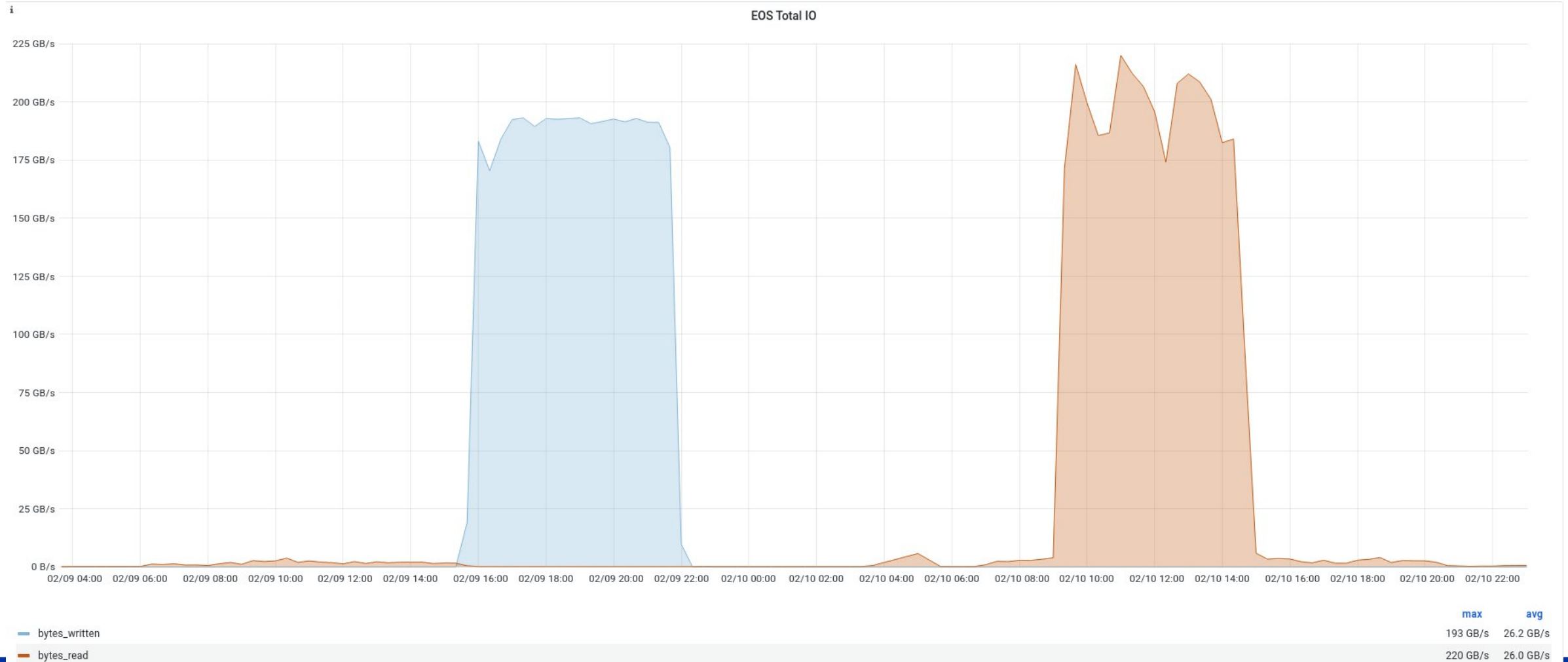
```
Nov 2 06:31:46 st-096-o2-184e66.cern.ch kernel: INFO: task xrootd:13150  
blocked for more than 120 seconds.  
Nov 2 06:31:46 st-096-o2-184e66.cern.ch kernel: Not tainted 4.18.0-  
348.el8.x86_64 #1  
Nov 2 06:31:46 st-096-o2-184e66.cern.ch kernel: "echo 0 >  
/proc/sys/kernel/hung_task_timeout_secs" disables this message.  
Nov 2 06:31:46 st-096-o2-184e66.cern.ch kernel: task:xrootd state:D  
stack: 0 pid:13150 ppid: 1 flags:0x00004080  
Nov 2 06:31:46 st-096-o2-184e66.cern.ch kernel: Call Trace:  
Nov 2 06:31:46 st-096-o2-184e66.cern.ch kernel: __schedule+0x2c4/0x700  
Nov 2 06:31:46 st-096-o2-184e66.cern.ch kernel: ?  
__switch_to_asm+0x35/0x70  
Nov 2 06:31:46 st-096-o2-184e66.cern.ch kernel: ?  
__switch_to_asm+0x35/0x70  
Nov 2 06:31:46 st-096-o2-184e66.cern.ch kernel: ?  
xfs_buf_read_map+0x1b5/0x310 [xfs]  
Nov 2 06:31:46 st-096-o2-184e66.cern.ch kernel: schedule+0x37/0xa0  
Nov 2 06:31:46 st-096-o2-184e66.cern.ch kernel:  
schedule_timeout+0x274/0x300  
Nov 2 06:31:46 st-096-o2-184e66.cern.ch kernel: ? __schedule+0x2cc/0x700  
Nov 2 06:31:46 st-096-o2-184e66.cern.ch kernel: ?  
blk_finish_plug+0x21/0x2e  
Nov 2 06:31:46 st-096-o2-184e66.cern.ch kernel: ?  
__xfs_buf_submit+0x122/0x1d0 [xfs]  
Nov 2 06:31:46 st-096-o2-184e66.cern.ch kernel: ?  
xfs_buf_read_map+0x1b5/0x310 [xfs]  
Nov 2 06:31:46 st-096-o2-184e66.cern.ch kernel:  
wait_for_completion+0x97/0x100  
Nov 2 06:31:46 st-096-o2-184e66.cern.ch kernel: ?  
__xfs_buf_submit+0x122/0x1d0 [xfs]  
Nov 2 06:31:46 st-096-o2-184e66.cern.ch kernel: xfs_buf_iowait+0x22/0xd0  
[xfs]  
Nov 2 06:31:46 st-096-o2-184e66.cern.ch kernel:  
__xfs_buf_submit+0x122/0x1d0 [xfs]  
...
```

[1] <https://marc.info/?l=linux-block&m=164366111512992>

Throughput tests with the 4.18.0-348 experimental kernel – Final_2 verdict



Throughput tests with the 4.18.0-358 kernel (Stream) – Final_3 verdict



Conclusions

- **Preparations and joint ALICE/IT tests have been very successful so far**
 - Excellent collaboration with ALICE and many IT teams: Network, HW Procurement, Resource Planning, Linux Support
- **No obvious show stoppers at the moment (anymore)**
- **We are confident that the instance will perform at more than the required O² performance**
- **More software developments to come in EOS⁵ for erasure coding and scheduling**
- **On the CentOS Stream side of things**
 - In principle we saw no major breakages in the userland, which is good
 - The hiccups on the kernel side are a bit more worrying, but:
 - As seen with the 'D' state problems, Enterprise Linux is also affected (not a Stream issue per se)
 - The feedback loop with Red Hat is much tighter and we appreciate that
 - Special measures might be needed (e.g.: more QA nodes to spot misbehavior and act faster)

The Linux future @ CERN & OS upgrades

- **CERN's Linux future for the close to medium term:**
 - CentOS Stream (8, 9?) for most of the use-cases
 - Enterprise Linux where specific cases require it
 - Already adopted CentOS Stream for EOS (only storage nodes for the time being)
- **EOS OS upgrades (how we do it)**
 - One by one upgrade of storage nodes (as we did for the ALICE02 C7->C8 migration); the C8->CS8 was much easier, as it was basically changing a Puppet variable resulting in:
 - changing OS repos
 - normal package upgrades
 - for the MGM not done yet (but, similar... upgrade passive MGM nodes first also one by one):
 - QuarkDB re-silvering if partition lost during install
 - Move MGM to one of the migrated slaves, upgrade former active node)

Thank you!
Questions?



home.cern