

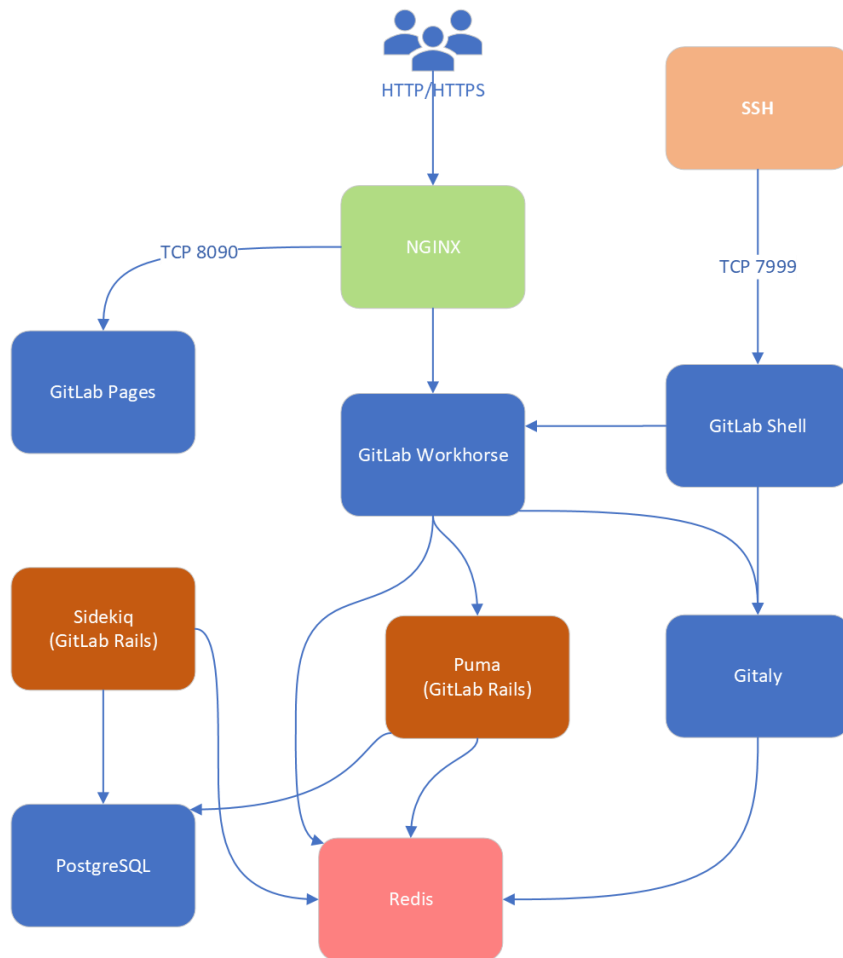


# **Version control and DevOps for accelerator and experiments: experience and outlook**

Ismael Posada Trobo

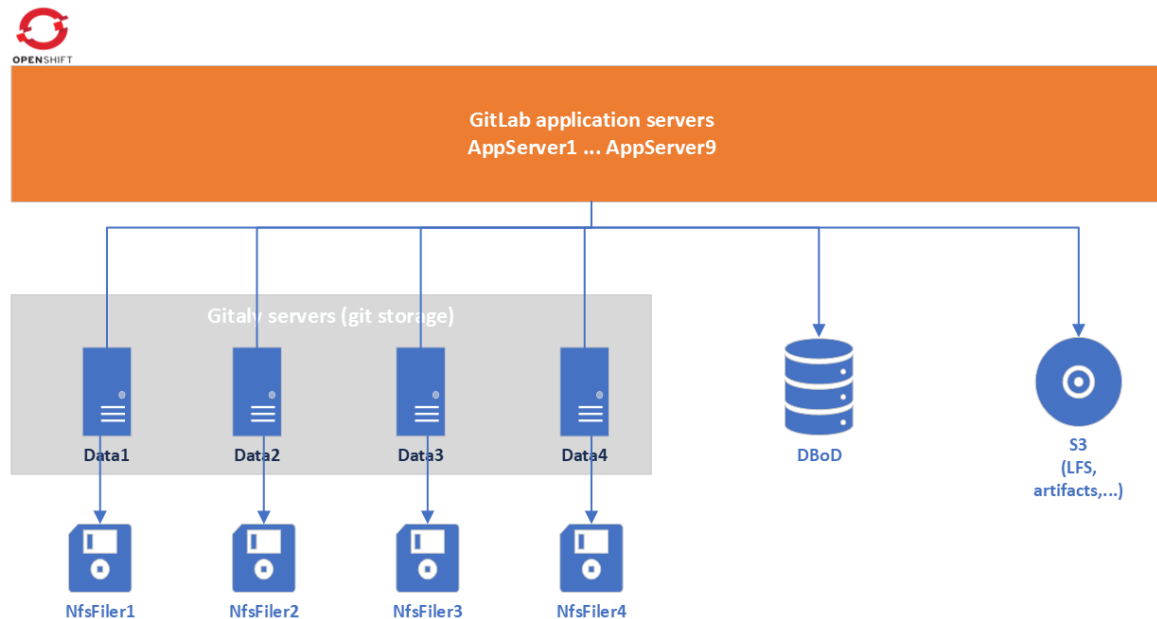
CHEP 2023 – Norfolk, Virginia (US) - 09<sup>th</sup> May 2023

# GitLab Infrastructure at CERN – Quick Overview



- **GitLab has been running at CERN since 2012**
- **Key infrastructure at CERN, suitable for our needs!**
- **Widely used at CERN and outside CERN for collaborators**
- **More than 17k active users**
- **More than 120k projects**
- **More than 5k jobs/hour**
- **Several externalized components:**
  - PostgreSQL: Database on Demand Service (provided at CERN).
  - GitLab Pages: OpenShift 4 (deployed at CERN).
  - Gitaly (repo data): **custom solution** using CephFS storage (provided by the Storage Team at CERN).

# GitLab Infrastructure at CERN – Initial stage



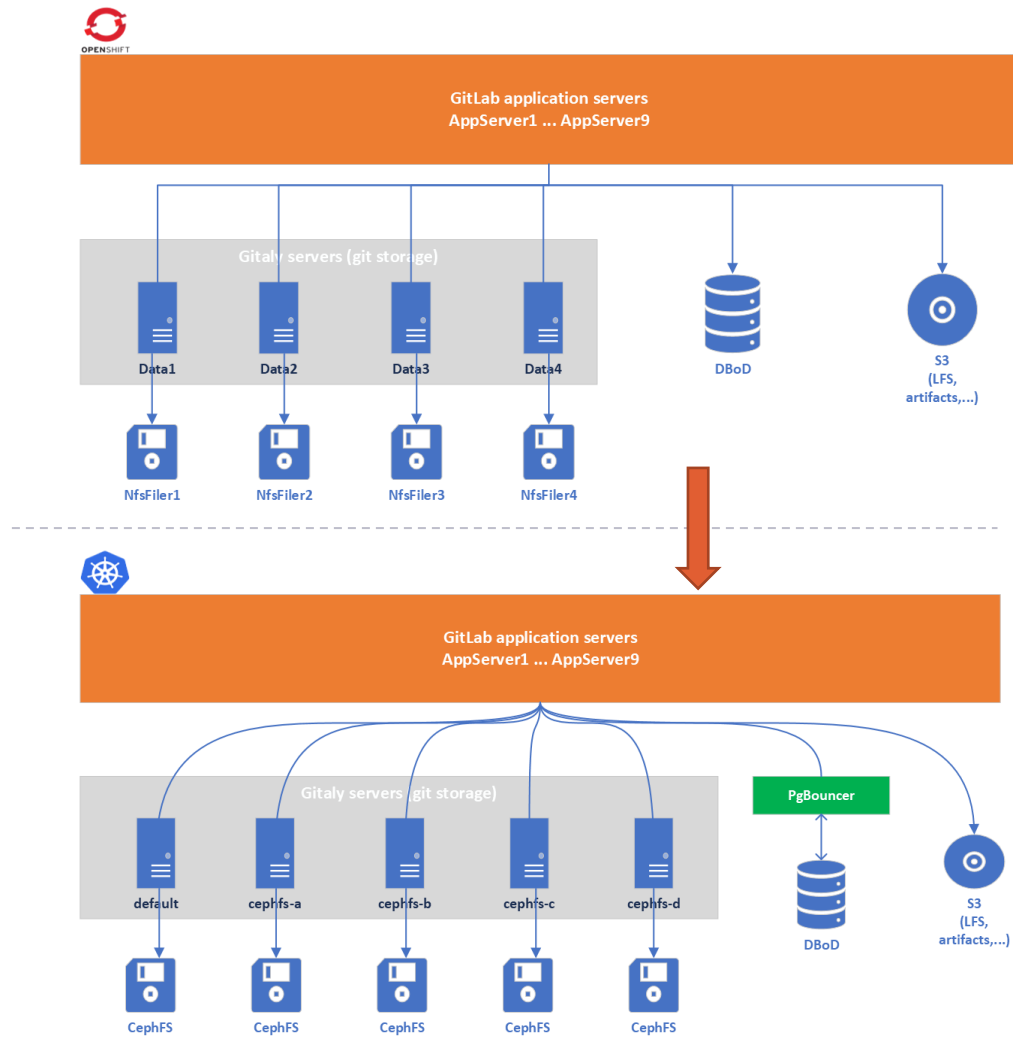
- **Custom non-supported infra (pre-migration)**

- 18 nodes OpenShift 3 infrastructure.
- Custom containerized solution based on Helm
  - Custom Omnibus solution
  - Custom Gitaly (git data) deployment

- **End of life of key components**

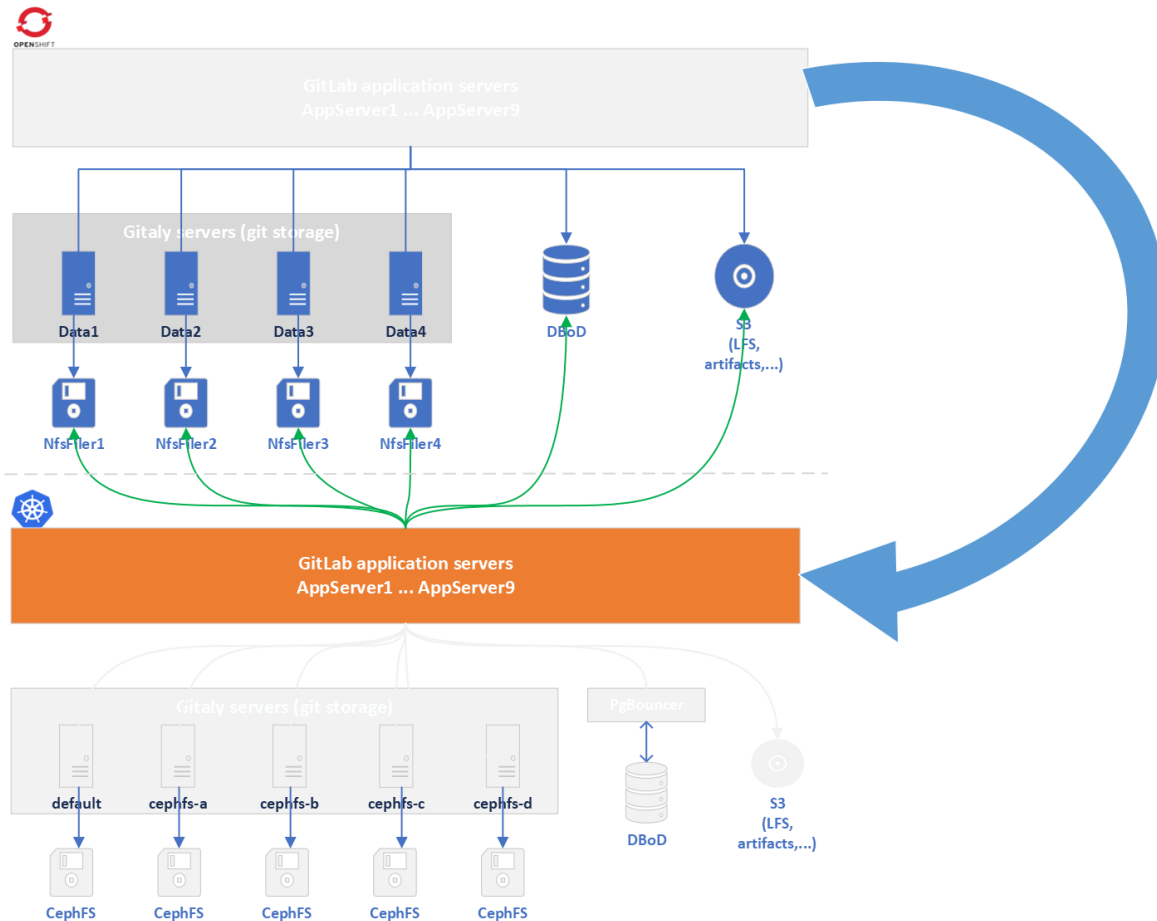
- OpenShift 3 in June 2022.
- NFS file system for git data end of support in November 2022.

# Migration – How we made it?



- **Both infrastructures running at the same time**
- **3 stages**
  - Move GitLab application first
  - Move git repos on a later stage
    - Git repos were accessed from any of the infrastructures!
  - Stabilization
- **Road to a Cloud Hybrid infrastructure**
  - 18 nodes Magnum Kubernetes.
  - Official supported containerized solution based on Helm.
  - CephFS file system for git data.

# Moving GitLab application - Migration



- **Move GitLab application first**
  - Magnum Kubernetes infra just running redis.
  - Gitaly connected through NodePort and Endpoints.
  - Switch off OpenShift 3 GitLab application, preserving redis running.
  - Send redis data over from one infra to another.
  - Reload redis in Magnum Kubernetes to pick up the desired data.
  - Bring up GitLab application on Magnum Kubernetes.
- **It works!**
- **But...**

# Moving GitLab application - Errors, bugs, pitfalls (I)

- **Several components misconfigured → indentation problems**
  - No notifications sent from GitLab
  - **GitLab LDAP sync misbehaving**
    - Thousands of projects lost their ownership.
  - Several of them luckily mitigated with default values.
- **Kerberos keytab misconfiguration for 1 server**
  - Not supported by GitLab yet! → Custom solution.
  - Duplicate value in Active Directory.
  - Some users experiencing problems while connecting through Kerberos.

# Moving GitLab application - Errors, bugs, pitfalls (II)

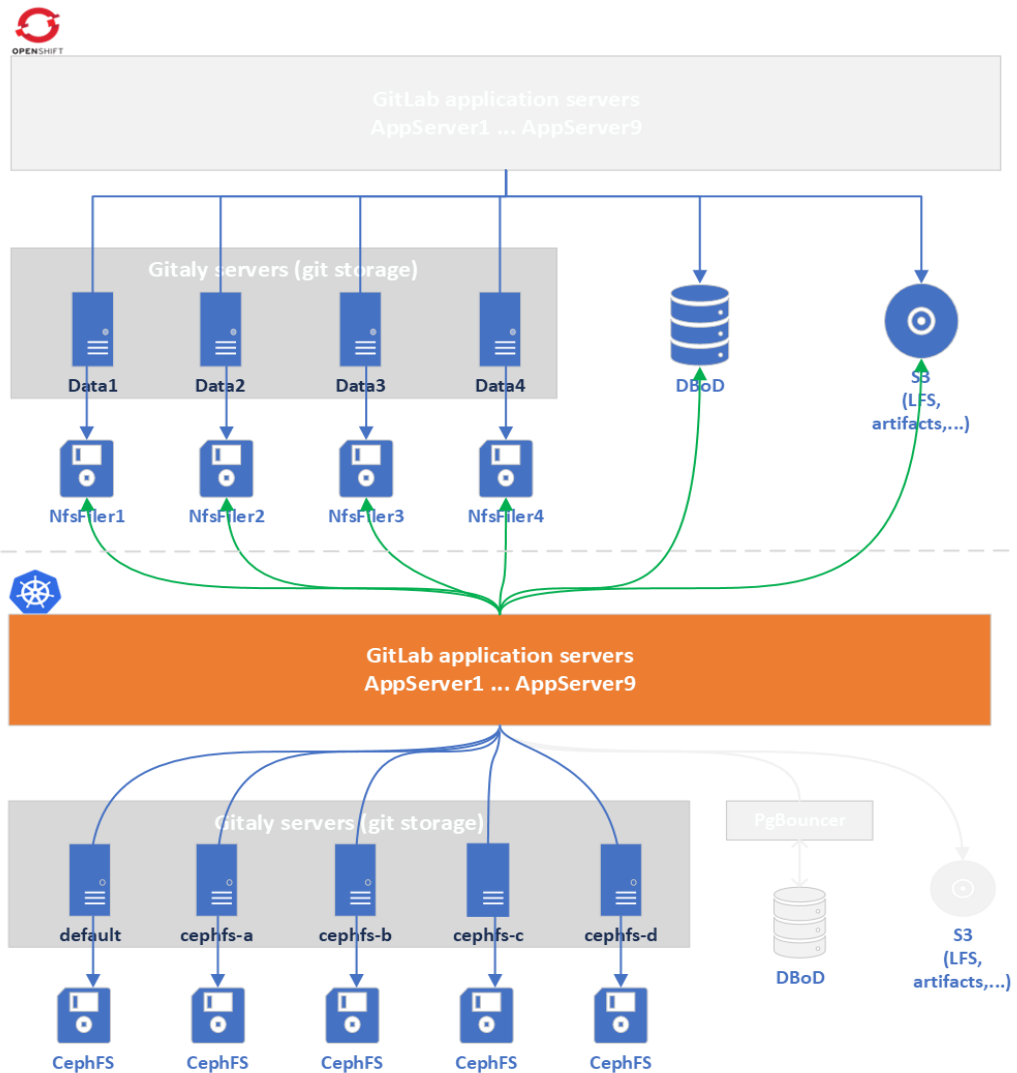
- **Moving from HAProxy to Nginx web server**

- No migration path for configuration
- Random users getting 403 error page (uploading artifacts)
  - Several hardcoded values upstream
  - `client_max_body_size` accommodated.
- Random users getting 400 error page (cookies too large)
  - `large_client_header_buffers` accommodated.

- **GitLab application bugs discovered**

- Webhooks showing wrong `git_ssh_url`
  - Reported and fixed by GitLab at [https://gitlab.com/gitlab-org/charts/gitlab/-/merge\\_requests/2782](https://gitlab.com/gitlab-org/charts/gitlab/-/merge_requests/2782)
- GitLab exporter component not connecting through Redis sentinel
  - Reported to GitLab at <https://gitlab.com/gitlab-org/charts/gitlab/-/issues/3813>

# Moving Git repos - Migration



- **Move git repos on a later stage**
  - Gitaly storages inter-communicated.
  - Use of NodePort and Endpoints for cross-cluster communication.
  - GitLab Workhorse pointing to the root hostname
- **Ignition...** 🚀



# Moving Git repos - Errors, bugs, pitfalls (I)

```
[38320.758201] BUG: kernel NULL pointer dereference, address: 0000000000000000
[38320.772502] #PF: supervisor read access in kernel mode
[38320.774478] #PF: error_code(0x0000) - not-present page
[38320.776094] PGD 8000000192290067 P4D 8000000192290067 PUD 17851e067 PMD 0
[38320.778108] Oops: 0000 [#1] PREEMPT SMP PTI
[38320.779751] CPU: 4 PID: 526844 Comm: gitally Not tainted 5.16.13-200.fc35.x86_64 #1
[38320.781908] Hardware name: RDO OpenStack Compute, BIOS 1.11.0-2.el7 04/01/2014
[38320.783978] RIP: 0010:ceph_fsinc+0x251/0x7e0 [ceph]
[38320.785690] Code: ff ff 4c 8d b0 e0 fc ff ff 48 39 c5 75 15 eb 60 49 8b 86 20 03 00 00 4c 8d b0 e0 fc ff ff 48 39 c5 74 4d 49 8b be 40 03 00 00 <48> 63 47 08 39 d8 0f 83 ac 00 00 00 49 83 3c c7
[38320.793692] RSP: 0018:fffffa05fcc73fe78 EFLAGS: 00010202
[38320.795331] RAX: fffff8de47d0f0e0 RBX: 0000000000000001 RCX: 00000000000000f3
[38320.797347] RDX: 0000000000000000 RSI: ffffffffcc0c683c1 RDI: 0000000000000000
[38320.799350] RBP: fffff8de691dbda60 R08: 0000000000000008 R09: fffff8de38bcbd350
[38320.801349] R10: 0000000000000000 R11: 0000000000000000 R12: fffff8de691dbdb00
[38320.817226] R13: fffff8de691dbda80 R14: fffff8de47d0f0d0 R15: fffff8de38bcbd350
[38320.819169] FS: 00007f59017fa700(0000) GS:ffff8de9f5400000(0000) knlGS:0000000000000000
[38320.821294] CS: 0010 DS: 0000 ES: 0000 CR0: 0000000080050033
[38320.823071] CR2: 0000000000000008 CR3: 00000001099e4004 CR4: 00000000003706e0
[38320.825040] DR0: 0000000000000000 DR1: 0000000000000000 DR2: 0000000000000000
[38320.826922] DR3: 0000000000000000 DR6: 00000000fffe0ff0 DR7: 0000000000000400
[38320.828791] Call Trace:
[38320.829757] <TASK>
[38320.830642] __x64_sys_fsinc+0x33/0x60
[38320.831871] do_syscall_64+0x3b/0x90
[38320.833493] entry_SYSCALL_64_after_hwframe+0x44/0xae
[38320.834963] RIP: 0033:0x48361b
[38320.836042] Code: e8 8a 76 fe ff eb 88 cc cc cc cc cc cc cc cc e8 5b bc fe ff 48 8b 7c 24 10 48 8b 74 24 18 48 8b 54 24 20 48 8b 44 24 08 0f 05 <48> 3d 01 f0 ff ff 76 20 48 c7 44 24 28 ff ff ff
[38320.841450] RSP: 002b:000000c00bdf2208 EFLAGS: 00000216 ORIG_RAX: 000000000000004a
[38320.843446] RAX: ffffffff8de47d0f0e RBX: 000000c000052000 RCX: 0000000000048361b
[38320.845288] RDX: 0000000000000000 RSI: 0000000000000000 RDI: 000000000000004b
[38320.847103] RBP: 000000c00bdf2248 R08: 000000c00bdf2201 R09: 000000c00bdf2240
[38320.849182] R10: 000000c00bdf21cc R11: 0000000000000216 R12: 000000c00bdf21d8
[38320.850997] R13: 0000000000000000 R14: 000000c00a350680 R15: 00007f59405b855a
[38320.852794] </TASK>
[38320.853612] Modules linked in: ceph libceph fscache netfs ipt_rpfiler xt_multiport iptable_mangle xt_set iptable_raw ip_set_hash_net ip_set_hash_ip ipip tunnel4 ip_tunnel bpf_preload veth ip6t
xt_statistic xt_nat xt_MASQUERADE xt_mark xt_addrtype ipt_REJECT nf_reject_ipv4 nft_chain_nat nf_nat nft_counter xt_comment xt_contrack nft_compat nf_tables ip_set nfnetlink ip_vs_sh ip_vs_wrr ip
intel_rapl_common snd_pscsp snd_pcm drm_kms_helper snd_timer snd_virtio_balloon joydev soundcore cec i2c_piix4 drm ip_tables xfs dm_multipath crct10dif_pclmul crc32_pclmul virtio_net crc32c_intel r
ipmi_devintf ipmi_msghandler fuse
[38320.882242] CR2: 0000000000000000
[38320.883360] ---[ end trace 9c08f421f18dd660 ]---
[38320.884687] RIP: 0010:ceph_fsinc+0x251/0x7e0 [ceph]
[38320.886087] Code: ff ff 4c 8d b0 e0 fc ff ff 48 39 c5 75 15 eb 60 49 8b 86 20 03 00 00 4c 8d b0 e0 fc ff ff 48 39 c5 74 4d 49 8b be 40 03 00 00 <48> 63 47 08 39 d8 0f 83 ac 00 00 00 49 83 3c c7
[38320.890719] RSP: 0018:fffffa05fcc73fe78 EFLAGS: 00010202
[38320.892315] RAX: fffff8de47d0f0e0 RBX: 0000000000000001 RCX: 00000000000000f3
[38320.894116] RDX: 0000000000000000 RSI: ffffffffcc0c683c1 RDI: 0000000000000000
[38320.895913] RBP: fffff8de691dbda60 R08: 0000000000000008 R09: fffff8de38bcbd350
[38320.897704] R10: 0000000000000000 R11: 0000000000000000 R12: fffff8de691dbdb00
[38320.899486] R13: fffff8de691dbda80 R14: fffff8de47d0f0d0 R15: fffff8de38bcbd350
[38320.901266] FS: 00007f59017fa700(0000) GS:ffff8de9f5400000(0000) knlGS:0000000000000000
[38320.903233] CS: 0010 DS: 0000 ES: 0000 CR0: 0000000080050033
[38320.904766] CR2: 0000000000000008 CR3: 00000001099e4004 CR4: 00000000003706e0
[38320.906601] DR0: 0000000000000000 DR1: 0000000000000000 DR2: 0000000000000000
[38320.908443] DR3: 0000000000000000 DR6: 00000000fffe0ff0 DR7: 0000000000000400
[38320.910278] Kernel panic - not syncing: Fatal exception
```

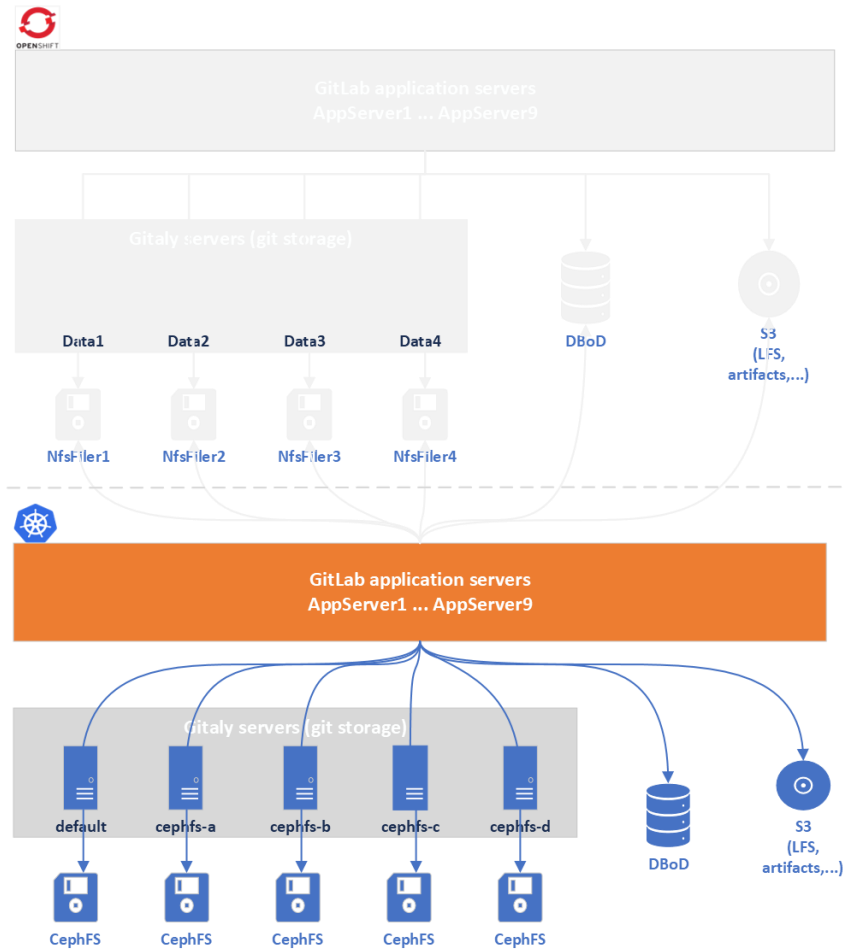
## The apocalypse ✨

- Migrated first 400 projects from NFS to CephFS.
- All good, no issues.
- Migrated rest of the 100k projects in bulk...
- **Hit Kernel bug → corruption of repositories started.**
- Maintenance mode? 🧑‍🔧
- Upgrade kernel in place? 🧑‍🔧
- Creation of a NodeGroup with specific version of the kernel? 🌟

# Moving Git repos - Errors, bugs, pitfalls (II)

- **Rediscover of an ancient issue, affecting ATLAS experiment**
  - Identified `gc.prunExpire` defaulting to 2 weeks (was 2 days)
  - Project with millions of loose objects
    - <https://git-scm.com/book/en/v2/Git-Internals-Maintenance-and-Data-Recovery>
  - At the same time, ATLAS experiment projects were performance offenders.

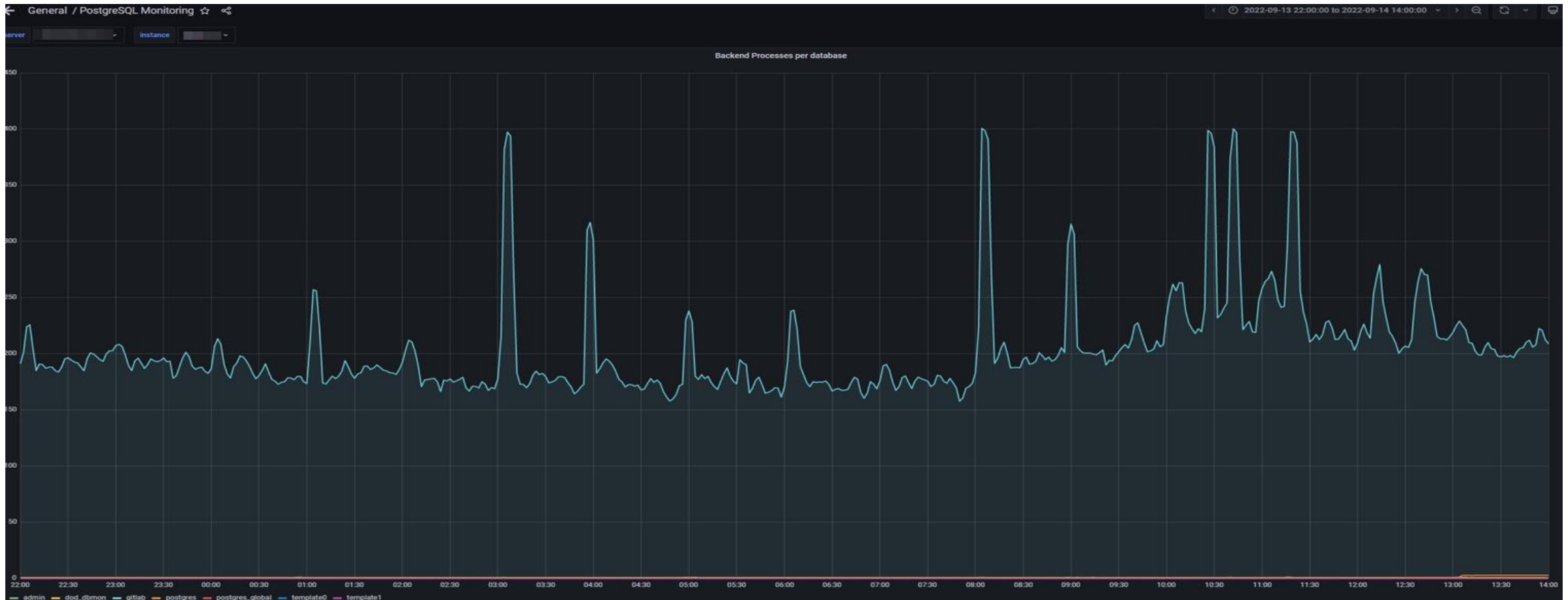
# Stabilization



- **We have survived!**
- **GitLab application and repository data fully functional**
- **Monitoring system improved**


# Stabilization

- **But... suspicious peaks in the database happening more often and suddenly.**



# Stabilization

- In brief, what was the bottleneck?




```
Samples: 3K of event 'cycles', Event count (approx.): 2146194646
Overhead Command Shared Object Symbol
47.17% postgres postgres [.] GetSnapshotData
1.72% postgres postgres [.] hash_search_with_hash_value
1.35% postgres postgres [.] _bt_compare
1.04% postgres postgres [.] AllocSetAlloc
0.99% postgres postgres [.] PostgresMain
0.90% postgres [kernel.vmlinux] [k] _raw_spin_lock_irqsave
0.77% postgres postgres [.] LWLockRelease
0.69% postgres [kernel.vmlinux] [k] _raw_spin_lock
0.66% postgres [kernel.vmlinux] [k] mutex_lock
0.64% postgres postgres [.] LockReleaseAll
0.62% postgres [kernel.vmlinux] [k] enqueue_task_fair
0.61% postgres postgres [.] AllocSetFree
0.54% postgres postgres [.] LWLockAcquire
0.54% postgres postgres [.] heap_hot_search_buffer
0.54% postgres [vdso] [.] __vdso_gettimeofday
0.53% postgres [kernel.vmlinux] [k] sock_wfree
0.50% postgres libc-2.31.so [.] __strlen_avx2
0.47% postgres [kernel.vmlinux] [k] enqueue_entity
0.46% postgres [kernel.vmlinux] [k] pollwake
0.46% postgres [kernel.vmlinux] [k] syscall_return_via_sysret
0.41% postgres [kernel.vmlinux] [k] skb_release_data
0.40% postgres postgres [.] hash_seq_search
0.39% postgres [kernel.vmlinux] [k] __ksize
0.38% postgres postgres [.] LockAcquireExtended
Tip: For hierarchical output, try: perf report --hierarchy
```

# Stabilization

- In brief, what was the bottleneck?

Samples: 3K of event 'cycles', Event count (approx.): 2146194646

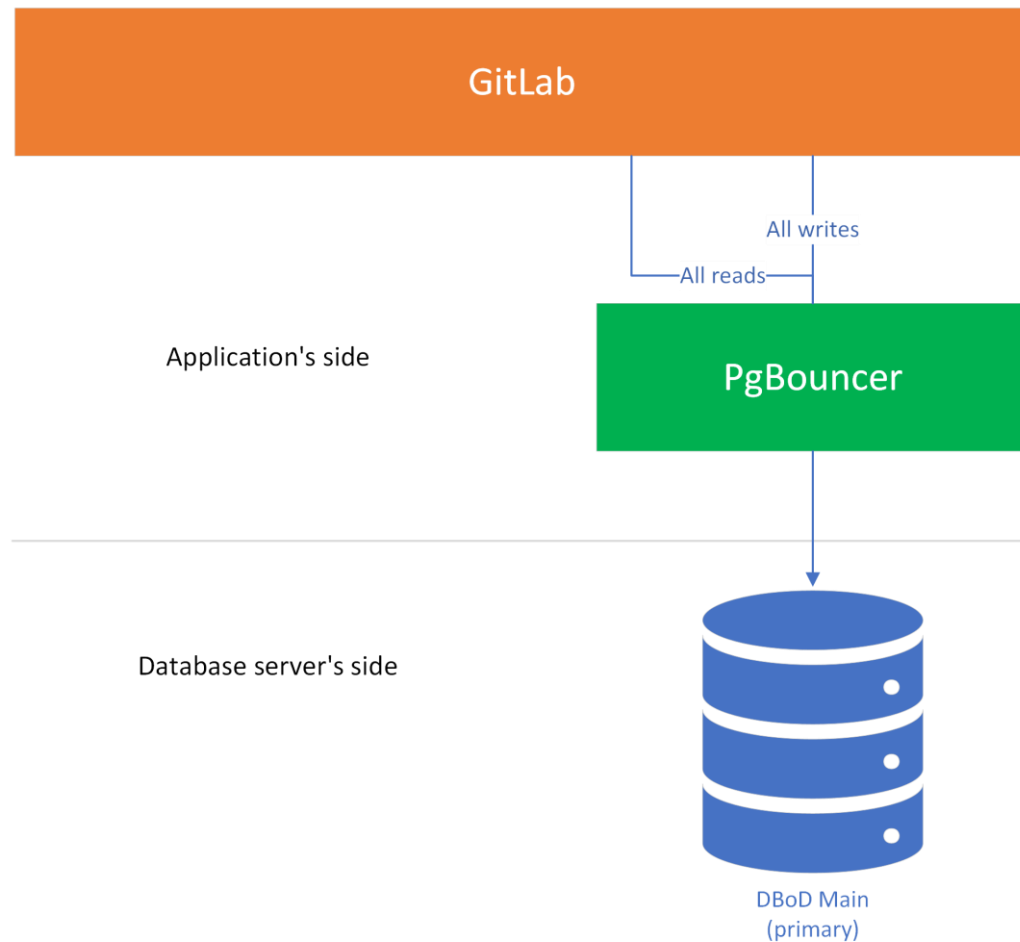


Overhead	Command	Shared Object	Symbol
47.17%	postgres	postgres	[.] GetSnapshotData
1.72%	postgres	postgres	[.] hash_search_with_hash_value
1.35%	postgres	postgres	[.] _bt_compare
1.04%	postgres	postgres	[.] AllocSetAlloc
0.99%	postgres	postgres	[.] PostgresMain
0.90%	postgres	[kernel.vmlinux]	[k] _raw_spin_lock_irqsave
0.77%	postgres	postgres	[.] LWLockRelease
0.69%	postgres	[kernel.vmlinux]	[k] _raw_spin_lock
0.66%	postgres	[kernel.vmlinux]	[k] mt...
0.64%	postgres	postgres	[.] _syscall
0.62%	postgres	[kernel.vmlinux]	[k] enqueue_task_fair
0.61%	postgres	postgres	[.] AllocSetFree
0.59%	postgres	postgres	[.] LWLockAcquire
0.58%	postgres	postgres	[.] heap_hot_search_buffer
0.53%	postgres	[vdso]	[.] __vdso_gettimeofday
0.53%	postgres	[kernel.vmlinux]	[k] sock_wfree
0.50%	postgres	libc-2.31.so	[.] __strlen_avx2
0.47%	postgres	[kernel.vmlinux]	[k] enqueue_entity
0.46%	postgres	[kernel.vmlinux]	[k] pollwake
0.46%	postgres	[kernel.vmlinux]	[k] syscall_return_via_sysret
0.41%	postgres	[kernel.vmlinux]	[k] skb_release_data
0.40%	postgres	postgres	[.] hash_seq_search
0.39%	postgres	[kernel.vmlinux]	[k] __ksize
0.38%	postgres	postgres	[.] LockAcquireExtended

Tip: For hierarchical output, try: perf report --hierarchy

**Snapshot scalability issue**

# Connection scalability issues - Transaction pooling



- **PgBouncer to the rescue**
  - Lightweight connection pooler for PostgreSQL.
  - “Middleman” between app and database server.
- **Modes**
  - Session
    - Assigns 1 client connection to a dedicated session.
- **Transaction**
  - **Creates a new connection for each transaction, returning the connection to the pool when transaction is complete**
- Statement
  - Caches prepared statements, reusing them for multiple client connections.

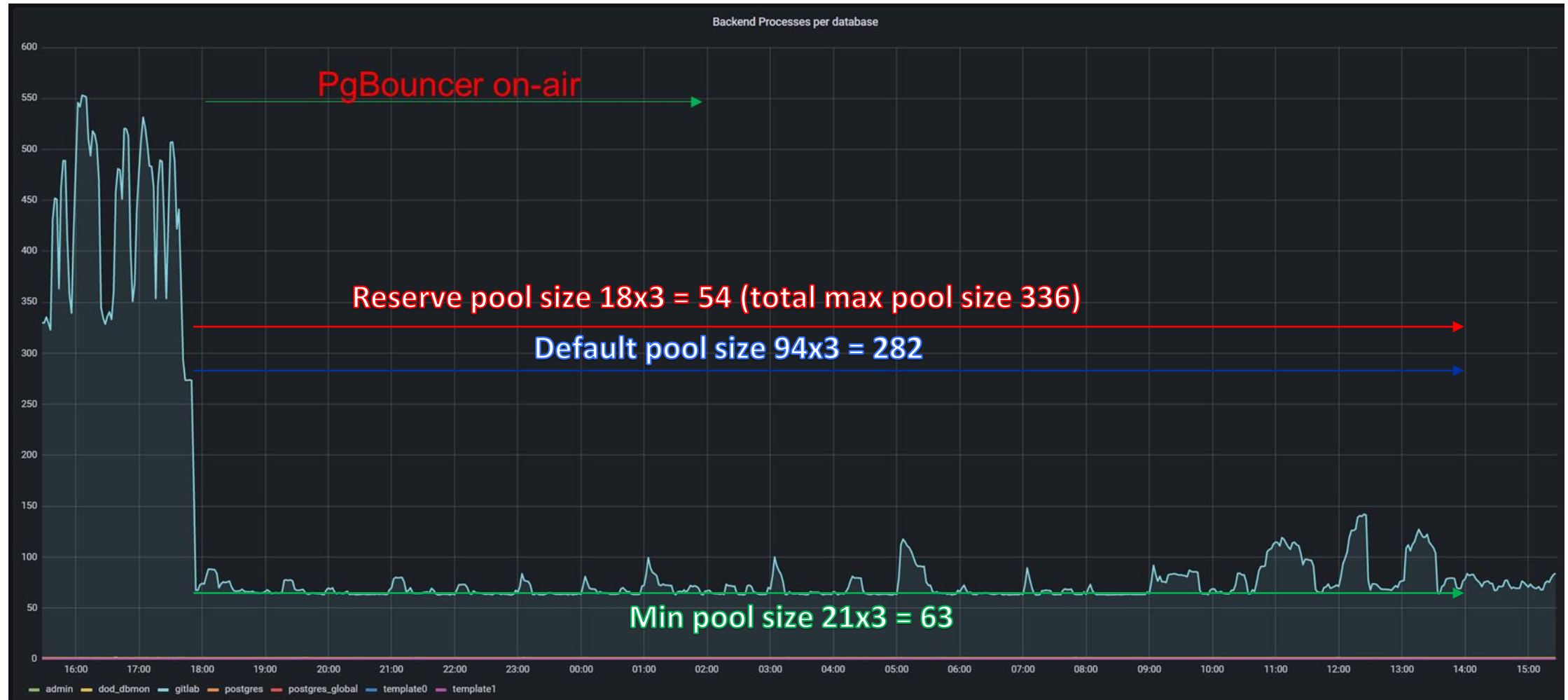
# Connection scalability issues - Transaction pooling

- **Lack of an official PgBouncer Helm chart provided by GitLab**
  - Created our own
    - [pgbouncer / PG Bouncer Helm Chart · GitLab \(cern.ch\)](#)
    - [pgbouncer / PG Bouncer Image · GitLab \(cern.ch\)](#)
  - **Contribution to GitLab (foreseen for v16.1)**
    - [Add PgBouncer Helm Chart \(!2973\) · Merge requests · GitLab.org / charts / GitLab Chart · GitLab](#)
  - **“Click-and-go” solution for Kubernetes**
    - Includes monitoring

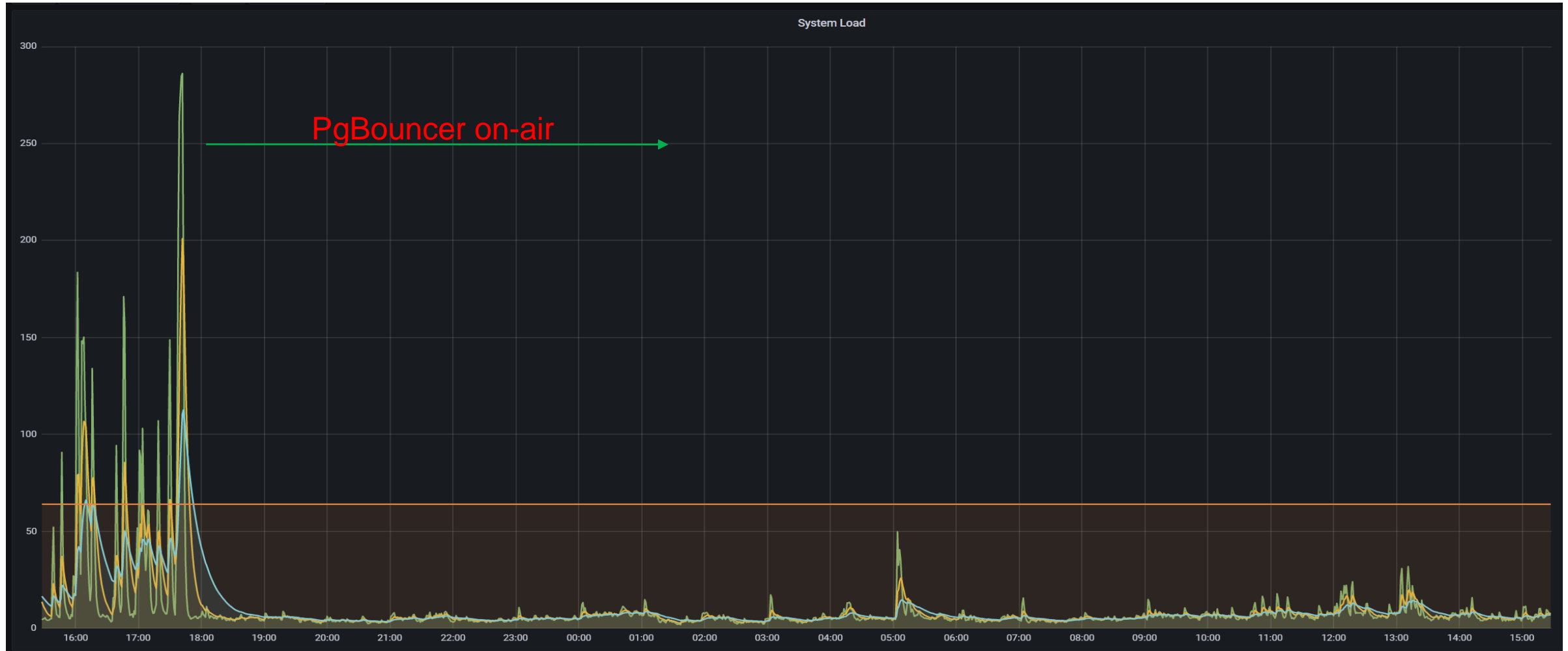




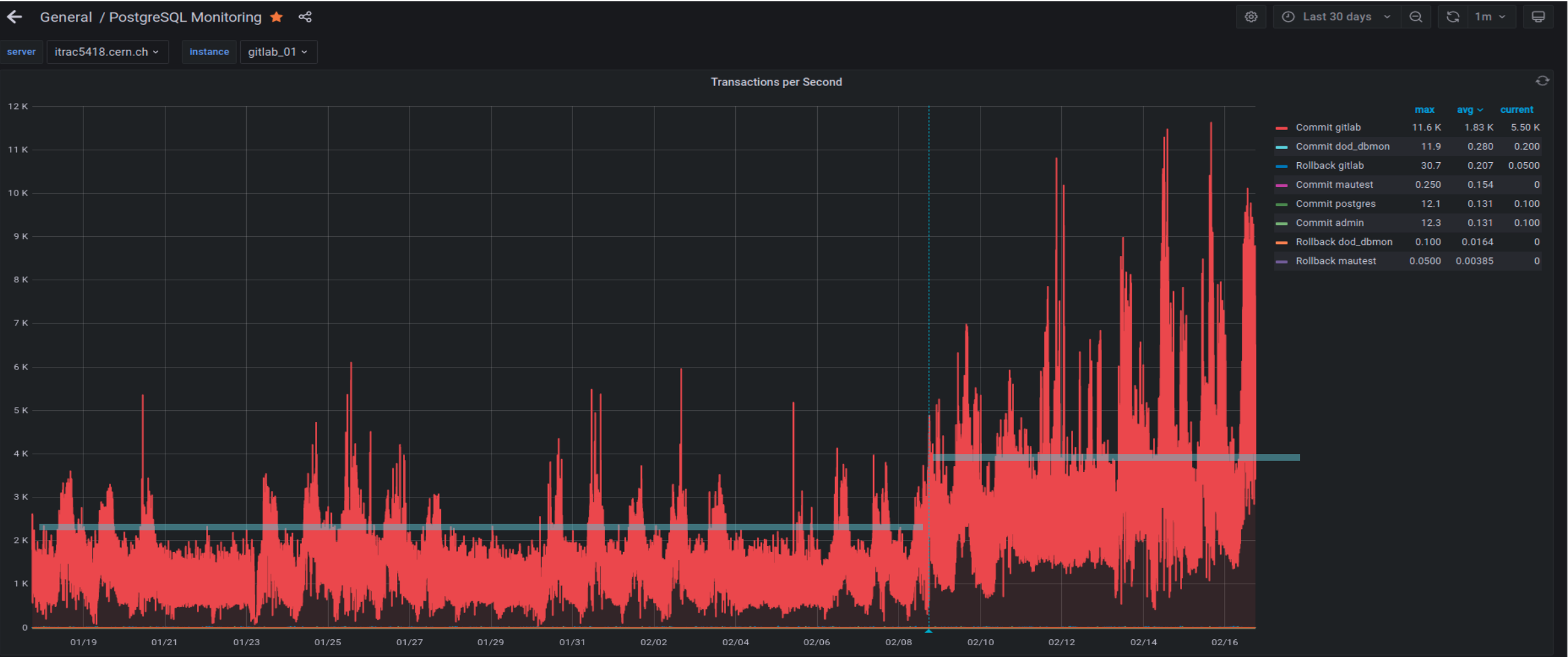
# Connection scalability issues - Transaction pooling



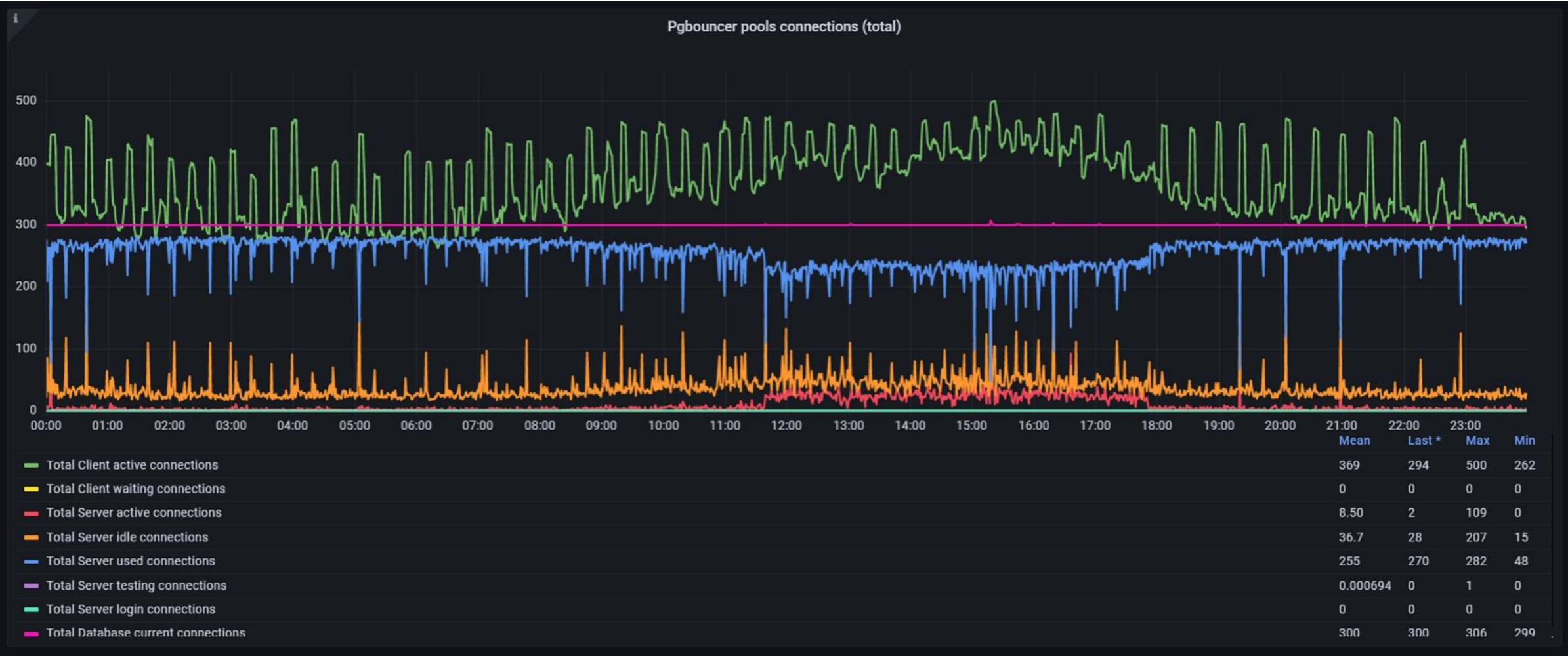
# Connection scalability issues - Transaction pooling



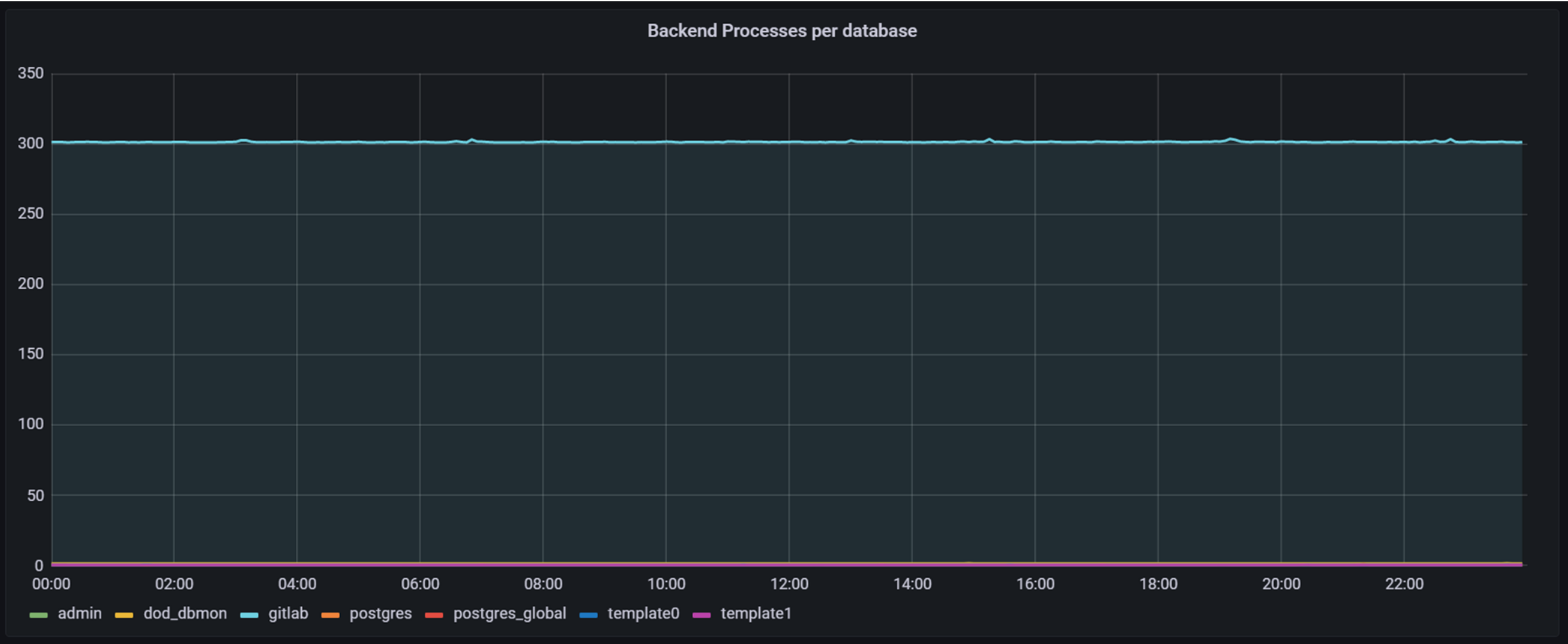
# Connection scalability issues - Transaction pooling



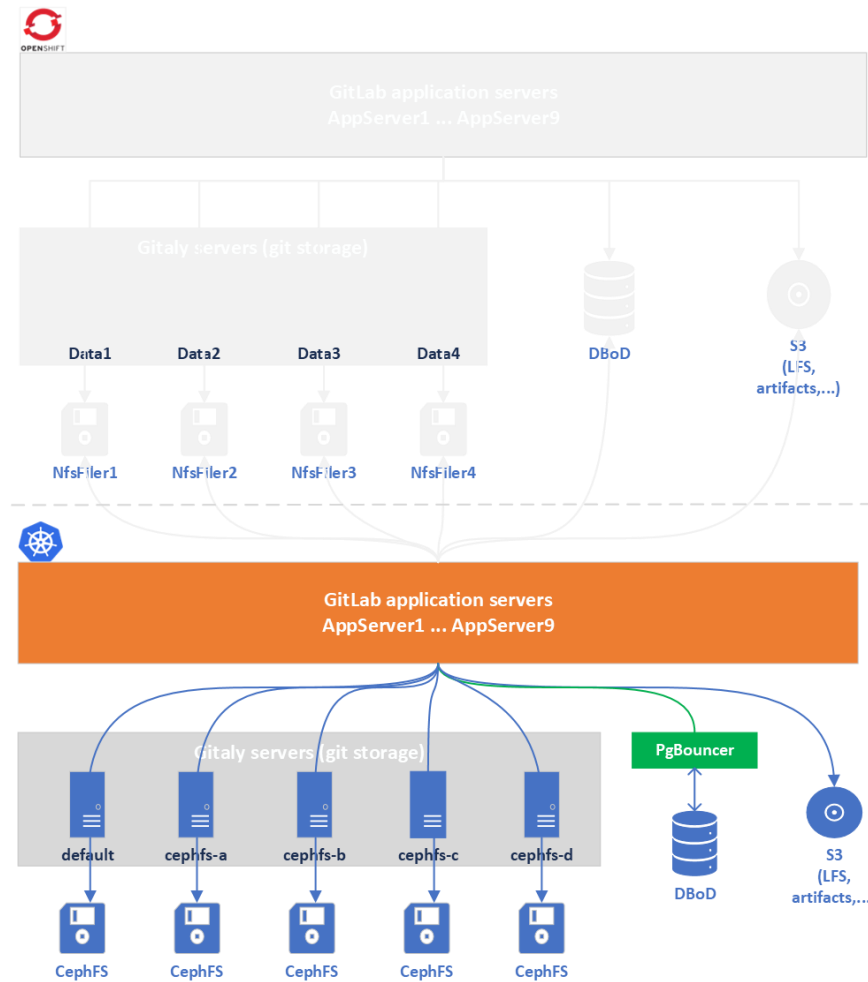
# Connection scalability issues - Transaction pooling



# Connection scalability issues - Transaction pooling



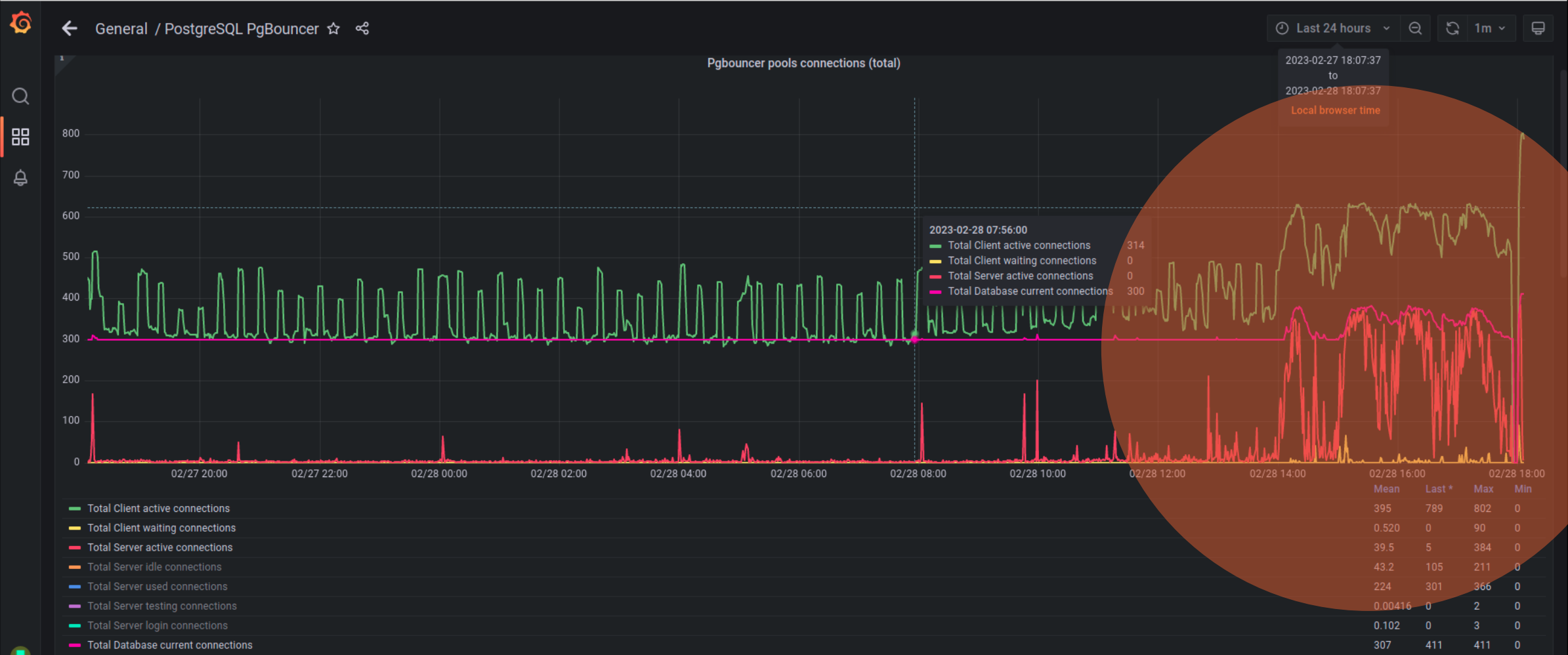
# Connection scalability issues



# Connection scalability issues

- **But ...**

# Connection scalability issues - Transaction pooling





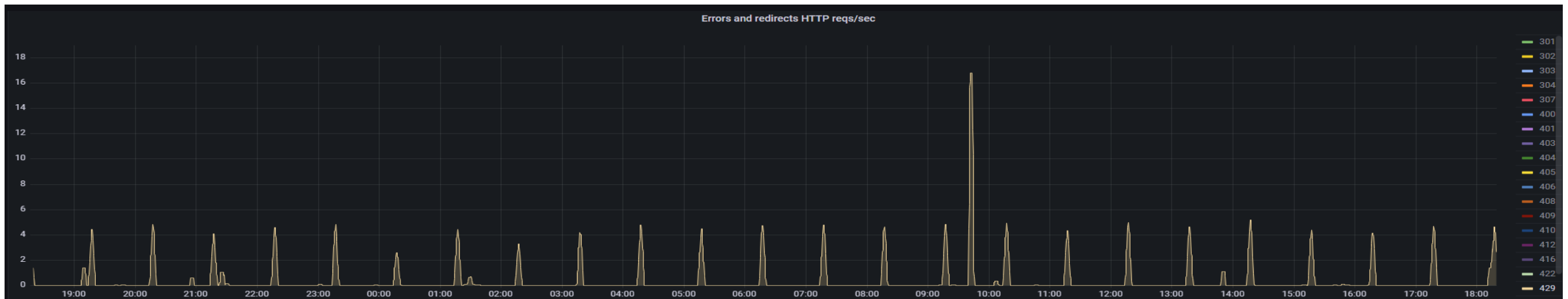
# Connection scalability issues - Transaction pooling



# Throttling the infrastructure

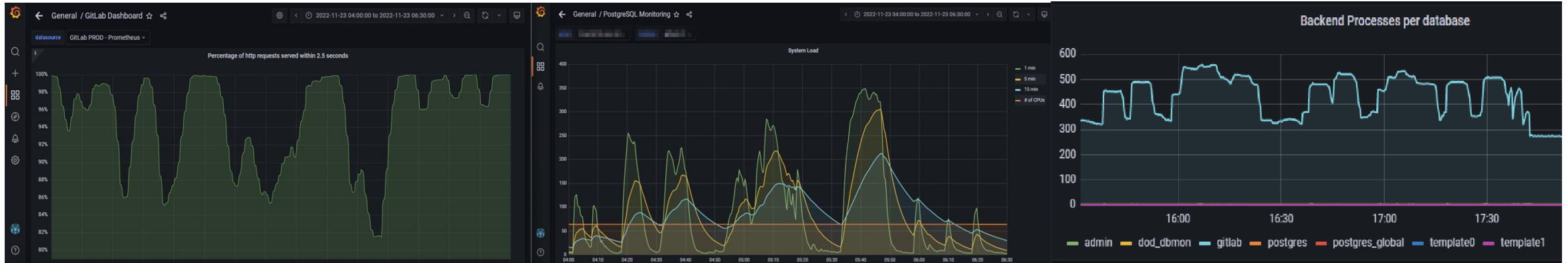
- **Throttling – Rate limits**

- Misuse and/or abuse from some users: **Too many requests – Error 429**
  - Infinite loops hammering the API
    - [Set rate limits for reqs/sec](#)
  - Huge number of jobs triggered simultaneously
    - Rate limits for the maximum number of jobs triggered per project.

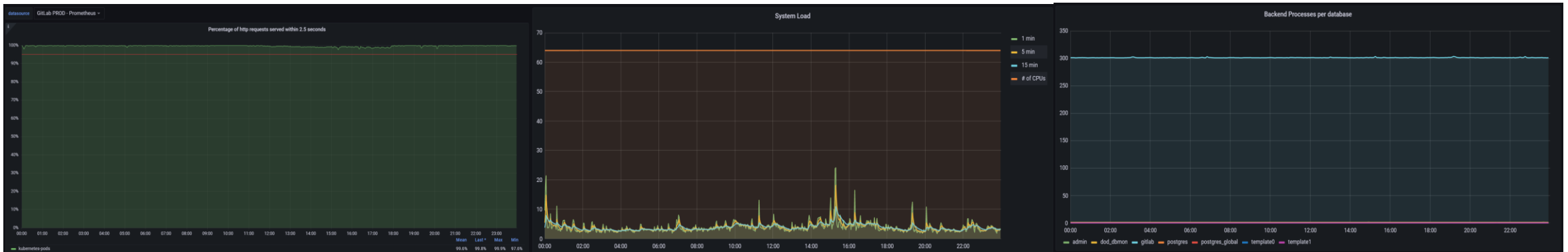


# Seeing the light at the end of the tunnel...

- Long journey from...

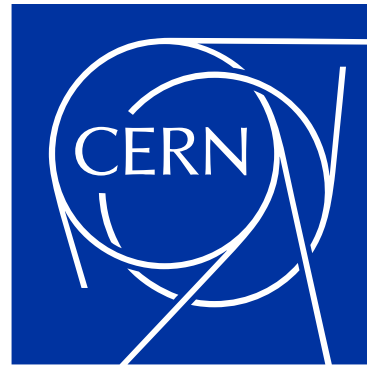


- ...to



# Conclusion

- **Our Version control system performs extremely well now.** All the effort put in place within all investigations, is translated into a more reliable, efficient and performing infrastructure CERN benefits from...
- ...but we as users need to use it responsibly, follow best practices, do not leave unused stuff behind and make sure we do not abuse of resources.
- All in all, in harmony, this will allow us to get ready for current and future challenges and further improvements of our beloved GitLab infrastructure.



[home.cern](https://home.cern)