GitLab.com database incident

Feb 1, 2017 · 5 min read · Leave a comment

(/company/team/)

GitLab (/company/team/)

Update: please see our postmortem for this incident (/blog/2017/02/10/postmortem-of-database-outage-of-january-31/)

Yesterday we had a serious incident with one of our databases. We lost six hours of database data (issues, merge requests, users, comments, snippets, etc.) for GitLab.com. Git/wiki repositories and self-managed installations were not affected. Losing production data is unacceptable and in a few days we'll publish a post on why this happened and a list of measures we will implement to prevent it happening again.

Update 6:14pm UTC: GitLab.com is back online

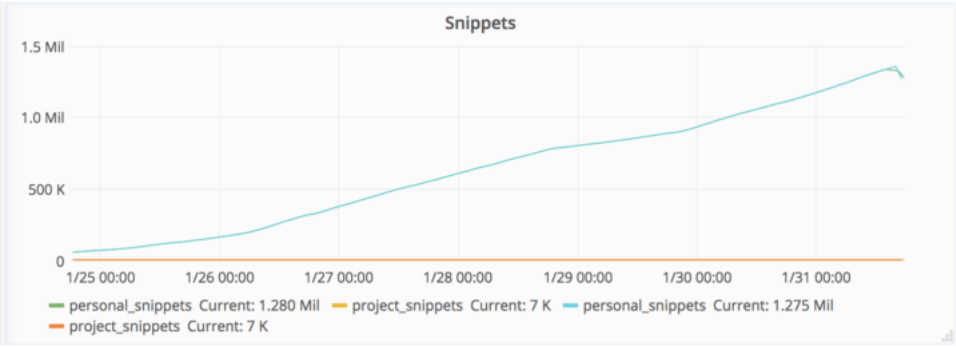
As of time of writing, we're restoring data from a six-hour-old backup of our database. This means that any data between 5:20pm UTC and 11:25pm UTC from the database (projects, issues, merge requests, users, comments, snippets, etc.) is lost by the time GitLab.com is live again.

Git data (repositories and wikis) and self-managed instances of GitLab are not affected.

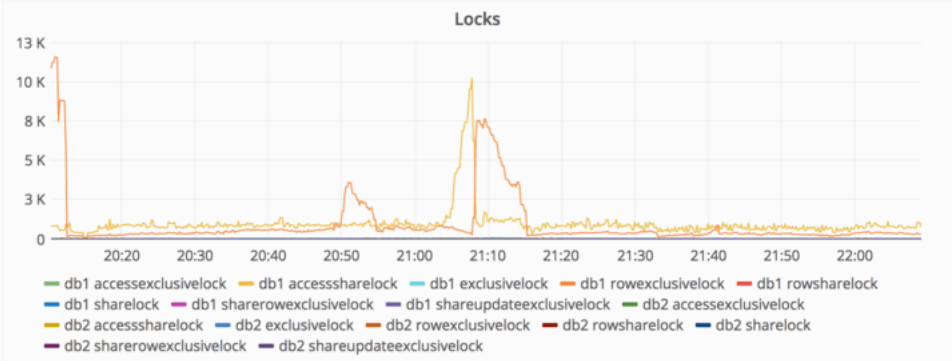
Read below for a brief summary of the events. You're also welcome to view our active postmortem doc (https://docs.google.com/document/d/1GCK53YDcBWQveod9kfzW-VCxIABGiryG7_z_6jHdVik/pub).

First incident

At 2017/01/31 6pm UTC, we detected that spammers were hammering the database by creating snippets, making it unstable. We then started troubleshooting to understand what the problem was and how to fight it.



At 2017/01/31 9pm UTC, this escalated, causing a lockup on writes on the database, which caused some downtime.

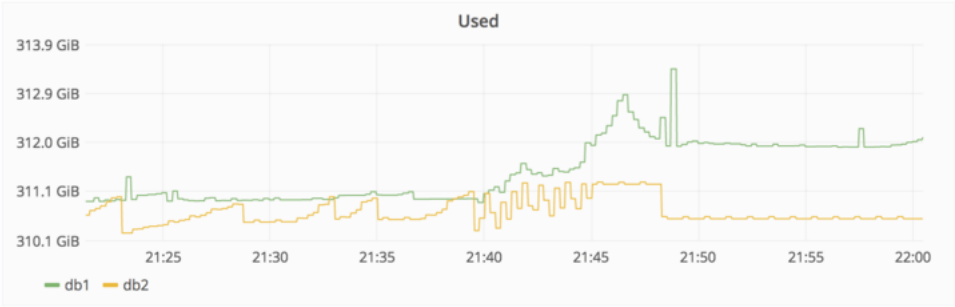


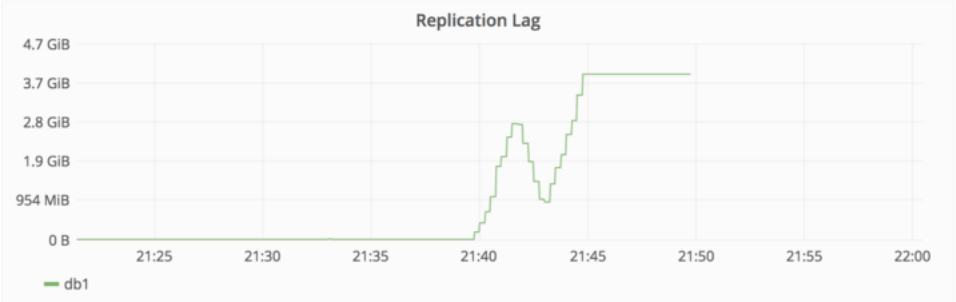
Actions taken

- We blocked the spammers based on IP address
- We removed a user for using a repository as some form of CDN, resulting in 47 000 IPs signing in using the same account (causing high DB load)
- We removed users for spamming (by creating snippets)

Second incident

At 2017/01/31 10pm UTC, we got paged because DB Replication lagged too far behind, effectively stopping. This happened because there was a spike in writes that were not processed ontime by the secondary database.





Actions taken

- Attempt to fix `db2`, it's lagging behind by about 4 GB at this point
- `db2.cluster` refuses to replicate, `/var/opt/gitlab/postgresql/data` is wiped to ensure a clean replication
- `db2.cluster` refuses to connect to `db1`, complaining about `max_wal_senders` being too low. This setting is used to limit the number of `WAL (= replication)` clients
- *Team-member-1* adjusts `max_wal_senders` to `32` on `db1`, restarts PostgreSQL
- PostgreSQL complains about too many semaphores being open, refusing to start

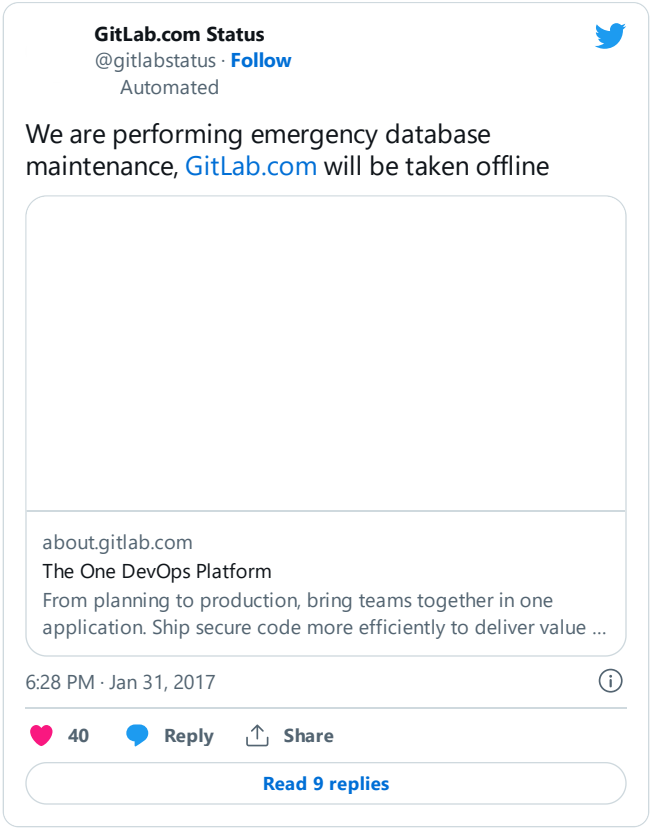
- *Team-member-1* adjusts `max_connections` to `2000` from `8000`, PostgreSQL starts again (despite `8000` having been used for almost a year)
- `db2.cluster` still refuses to replicate, though it no longer complains about connections; instead it just hangs there not doing anything
- At this point frustration begins to kick in. Earlier this night *team-member-1* explicitly mentioned he was going to sign off as it was getting late (23:00 or so local time), but didn't due to the replication problems popping up all of a sudden.

Third incident

At 2017/01/31 11pm-ish UTC, *team-member-1* thinks that perhaps `pg_basebackup` is refusing to work due to the PostgreSQL data directory being present (despite being empty), decides to remove the directory. After a second or two he notices he ran it on `db1.cluster.gitlab.com`, instead of `db2.cluster.gitlab.com`.

At 2017/01/31 11:27pm UTC, *team-member-1* - terminates the removal, but it's too late. Of around 300 GB only about 4.5 GB is left.

We had to bring GitLab.com down and shared this information on Twitter:



Problems encountered

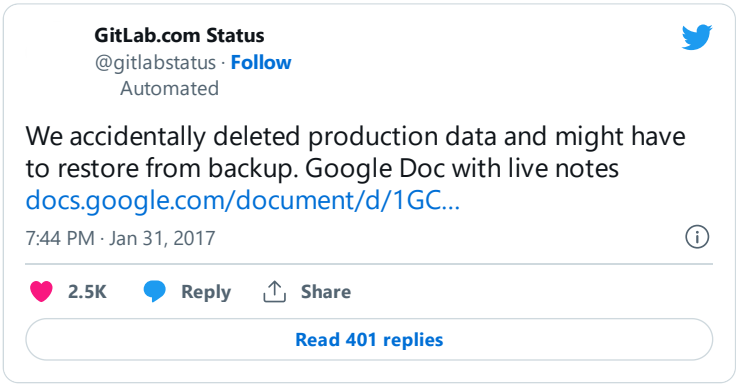
- LVM snapshots are by default only taken once every 24 hours. *Team-member-1* happened to run one manually about six hours prior to the outage because he was working in load balancing for the database.
- Regular backups seem to also only be taken once per 24 hours, though *team-member-1* has not yet been able to figure out where they are stored. According to *team-member-2* these don't appear to be working, producing files only a few bytes in size.
- *Team-member-3*: It looks like `pg_dump` may be failing because PostgreSQL 9.2 binaries are being run instead of 9.6 binaries. This happens because omnibus only uses Pg 9.6 if data/PG_VERSION is set to 9.6, but on workers this file does not exist. As a result it defaults to 9.2, failing silently. No SQL dumps were made as a result. Fog gem may have cleaned out older backups.

- Disk snapshots in Azure are enabled for the NFS server, but not for the DB servers.

- Our backups to S3 apparently don't work either: the bucket is empty
- So in other words, out of five backup/replication techniques deployed none are working reliably or set up in the first place. We ended up restoring a six-hour-old backup.
- pg_basebackup will silently wait for a master to initiate the replication progress, according to another production engineer this can take up to 10 minutes. This can lead to one thinking the process is stuck somehow. Running the process using "strace" provided no useful information about what might be going on.

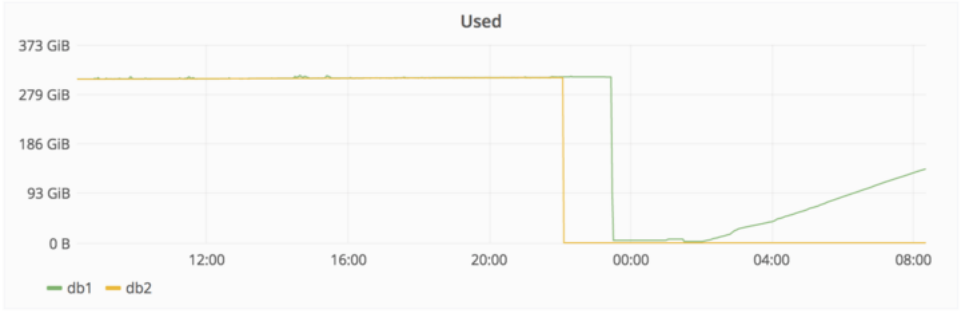
Recovery

We're working on recovering right now by using a backup of the database from a staging database.



- 2017/02/01 00:36 - Backup `db1.staging.gitlab.com` data
- 2017/02/01 00:55 - Mount `db1.staging.gitlab.com` on `db1.cluster.gitlab.com`
- Copy data from staging `/var/opt/gitlab/postgresql/data/` to production `/var/opt/gitlab/postgresql/data/`
- 2017/02/01 01:05 - `nfs-share01` server commandeered as temp storage place in `/var/opt/gitlab/db-meltdown`
- 2017/02/01 01:18 - Copy of remaining production data, including `pg_xlog` tar'ed up as `20170131-db-meltodwn-backup.tar.gz`

Below a graph showing the time of deletion and subsequent copying in of data.



Also, we'd like to thank everyone for the amazing support we've received on Twitter and elsewhere through #hugops (https://twitter.com/i/moments/826818668948549632)

u=https%3A%2F%2Fabout.gitlab.com%2Fblog%2F2022%2F10%2F07%2Fgitlab-dot-com-database-incident%2F&title=GitLab.com database incident)

Tags:

More to explore

ENGINEERING

Take advantage of Git rebase

GitLab

(/blog/2022/10/06/take-advantage-of-git-rebase/)

ENGINEERING

A visual guide to GitLab CI/CD caching

GitLab

(/blog/2022/09/12/a-visual-guide-to-gitlab-ci-caching/)

ENGINEERING

Speed up your monorepo workflow in Git

GitLab

(/blog/2022/09/06/speed-up-your-monorepo

All Blog Posts → (/blog/)

133 Comments

Login

Join the discussion…

LOG IN WITH

OR SIGN UP WITH DISQUS

Name

Sort by Best 36

codepotato • 6 years ago

Thanks for the great explanation. It's a shame it's happened, but at the same time it's great that you're being so open. Now, if we could just find a way of speeding up the restore :) Sending hugs to the engineers solving this!

132 | • Reply • Share ›

A. Felipe Cabargas codepotato • 6 years ago

Thank you! We're happy to know the community appreciates our transparency. We hope to have GitLab.com online ASAP!

21 | 1 • Reply • Share ›

Sytse Sijbrandij GitLab A. Felipe Cabargas • 6 years ago

2 hours later we were online again

1 | • Reply • Share ›

morph027 • 6 years ago

Sh*t happens, huge props for being transparent as ever! Makes waiting easier. Virtual cup of coffee for your engineers!

49 | • Reply • Share ›

morph027 ➚ morph027 • 6 years ago
Also this is the best method learning and implementing new precaution methods, as this hurts a lot (also did something similiar years ago) ;)
9 | • Reply • Share ›

Job van der Voort ➚ morph027 • 6 years ago
Thanks for the support @morph027
1 | • Reply • Share ›

Timo Witte • 6 years ago
Aw... so many classics:
- Happening near to end of working day
- Backups non restorable, due to problems with backup procedure, which hasn´t been checked for months
- Accidently deleting the other side, during DB Replication (How many times, did you accidently added "DROP TABLE XYZ;" into the binlog in a master <-> master replication ^^)
- Having to restore from a slow backup source (like local backup behind a DSL connection or something like that)
- Cold sweat when you first realize your misery
45 | • Reply • Share ›

Job van der Voort ➚ Timo Witte • 6 years ago
This was pretty painful indeed. We'll be doing a proper post-mortem at a later time soon.
3 | • Reply • Share ›

Sytse Sijbrandij GitLab ➚ Job van der Voort • 6 years ago
Post-mortem write-up issue
https://gitlab.com/gitlab-c...
2 | • Reply • Share ›

Ezequiel Pellettieri • 6 years ago
Shit happens guys. Thanks for all the updates and for streaming the incident resolution. #JeSuisTeamMember1

see more

41 | • Reply • Share ›

Job van der Voort ➚ Ezequiel Pellettieri • 6 years ago
This is hilarious, thanks!
| • Reply • Share ›

Cezariusz Marek • 6 years ago

36 | • Reply • Share ›

Job van der Voort ➚ Cezariusz Marek • 6 years ago

We can't agree more. We'll make sure to check them regularly.

Fun fact: Did you know that Schrodinger didn't mean to _explain_ anything with his thought experiment, rather he wanted to illustrate the absurdity of the existing view of quantum mechanics?

3 | • Reply • Share ›

An0nym0usC0ward ➚ Job van der Voort • 5 years ago

Late to the game, but I wouldn't go for backups. Instead, I'd go for a monkey-plagued replication infrastructure - what netflix uses. If monkeys cannot take your replication down, chances are it will survive accidents like the one described above too.

| • Reply • Share ›

Connor Shea ➚ An0nym0usC0ward • 5 years ago

I believe the plan is to do this eventually, but right now we want to make the infrastructure solid enough to handle monkeys before adding them.

| • Reply • Share ›

Ivens • 6 years ago • edited

Oh, boy! I'm feeling bad for team-member-1. But rest assured mate, mistakes happen and you will recover from it.

Good job for being so transparent, Gitlab!

23 | • Reply • Share ›

Job van der Voort ➚ Ivens • 6 years ago

He's doing well, thanks for the support!

4 | • Reply • Share ›

Denis ➚ Job van der Voort • 6 years ago

The moment you realize the error is the worst feeling. We've all done it.

1 | • Reply • Share ›

Jarle Nygård ➚ Denis • 6 years ago

Oh yes!

| • Reply • Share ›

dr_drsh • 6 years ago

I think an important factor behind this mishap is that Team-member-1 had to deal with the problem while in need of sleep. Doctors perform their worst around the end of their shift (~5 AM) and I would think sysadmins are the same. Stay strong Team-member-1, we've all been there.

14 | • Reply • Share ›

Ender Aydin Ozek • 6 years ago

Ender Aydin Orak • 6 years ago

Gitlab is an amazing platform and I hope you will get over this quickly. Kudos for being so open and honest. Wish the best for you.

Ps. Same thing happened to Github before.
https://github.com/blog/744...

14 | 1 • Reply • Share ›

Job van der Voort ➔ Ender Aydin Orak • 6 years ago

Thanks @Ender Aydin Orak, glad to have the support for the community.

2 | • Reply • Share ›

rumurphy • 6 years ago

As a DBA, doing something to the primary DB when you think it's the backup DB is one of my top nightmare scenarios. So easy to do with fatigue and the fact that the primary and the backup (by design) look so damn similar. To help prevent that for ssh sessions, in putty you can save sessions with specific background colors. I have my primary in red, backup in blue, and dev in green. It helps me instantly know where I am.

6 | • Reply • Share ›

Curt J. Sampson ➔ rumurphy • 6 years ago

Programmers pair when writing code to reduce errors; in the same way, sysadmins should pair when doing manual operations on critical systems to help catch errors.

1 | • Reply • Share ›

Pablo C ➔ rumurphy • 6 years ago • edited

This made me think that a nice way to do this would be to check if the DB is acting as master and color the PS based on that, that way it would be easier to spot if the current DB is the master or not.

One of the main issues is that it's hard to see db1 vs db2 - even if you add the whole name, you don't know which is acting as primary because they rotate, and that's good.

https://gitlab.com/gitlab-c...

Thanks for the suggestion!

| • Reply • Share ›

Aleksey Leshchuk ➔ rumurphy • 6 years ago

nice idea, I actually also moved ssh command to different scripts: production.sh, staging.sh, and never run raw ssh ... .

| • Reply • Share ›

Pierre Untel ➔ rumurphy • 6 years ago • edited

good idea!, i will apply it to .bashrc instead

| • Reply • Share ›

webster ➔ Pierre Untel • 6 years ago

avoid green because of color blindness. use orange or purple instead

| • Reply • Share ›

jinmatt • 6 years ago

Great transparency. Can't blame team-member-1 for the mistake. But next time follow good database naming conventions instead of just naming them db1 and db2!

6 | • Reply • Share ›

Fred@Bootstrap • 6 years ago

A message for team-member-1: please don't let this get you down, trust me we've all been there and remember this is the first time this has happened. Gitlab, to their credit, have been very

open and responsive about it.

6 ^ | v • Reply • Share ›

ivanfeanor • 6 years ago

Been there

http://devopsreactions.tumb...

6 ^ | v • Reply • Share ›

Job van der Voort → ivanfeanor • 6 years ago

Haha, I'm sure that's how it went!

^ | v • Reply • Share ›

Zeno Popovici • 6 years ago

I'm sorry to make a discordant note here, but this is not "a shame" or a "shit happens" moment. This only shows how bad at safeguarding customer data GitLab is.

They don't do interval snapshots, only regular backups, that aren't tested for recovery every 24-48h. And because they never test restores, none of their "backups" are actually any good. They also never check to see if the backup is actually taking place ("S3 bucket is empty").

From my understanding, if "team-member-1" didn't happen to do a backup of a staging database 6h earlier, all would be lost ...

I can understand and accept human error (although, "rm -rf' should be protected on production by simple aliasing), what I can't accept is irresponsible and incompetent behaviour.

Can't believe people are praising them ...

9 ^ | v 1 • Reply • Share ›

Job van der Voort → Zeno Popovici • 6 years ago

Yes, this shouldn't have happened. We're not saying it's a shame. It's unacceptable and we're taking measures both process-wise as well as technical to make sure something like this can never happen again.

Thanks for the candor @Zeno Popovici

2 ^ | v • Reply • Share ›

Zeno Popovici → Job van der Voort • 6 years ago

Thank you for acknowledging this. I really don't want to see GitLab fail. Godspeed.

1 ^ | v • Reply • Share ›

Cesar Diaz • 6 years ago

Been there. Huge props for being so open. I'm sure team-member-1 is kicking themselves right now, but stuff happens. Learn from the experience and carry on.

5 ^ | v • Reply • Share ›

gamesbook → Cesar Diaz • 6 years ago

Please don't fire team-member-1 ... but maybe create an annual award in his name <evil grin="">

2 ^ | v • Reply • Share ›

Job van der Voort → gamesbook • 6 years ago

No worries, he won't be fired.

Thanks for the support!

abc@xyz.com

Subscribe

I would like to receive the bi-weekly newsletter via email

Having trouble viewing or submitting this form?

Platform

Why GitLab (/why-gitlab/)

Pricing (/pricing/)

The DevOps Lifecycle (/stages-devops-lifecycle/)

Features (/features/)

Releases (/releases/)

Solutions

Enterprise (/enterprise/)

Small Business (/small-business/)

Continuous Integration (/solutions/delivery-automation/)

Public Sector (/solutions/public-sector/)

Education (/solutions/education/)

Resources

Blog (/blog/)

Install (/install/)

Docs (https://docs.gitlab.com/)

Developer Portal (https://developer.gitlab.com/)

Newsletter (/company/contact/)

Support

Get help (/get-help/)

Contact Sales (/sales/)

Support (/support/)

Status (https://status.gitlab.com/)

Customers Portal (https://customers.gitlab.com/)

Community Forum (https://forum.gitlab.com/)

Company

About (/company/)

Jobs (/jobs/)

Leadership (/company/team/e-group/)

Board of Directors (/company/team/board-of-directors/)

Team (/company/team/)

Press (/press/)

Investor Relations (https://ir.gitlab.com)

Handbook (/handbook/)

Terms of Use (/terms/)

Privacy Statement (/privacy/)

Cookie Preferences

Shop (https://shop.gitlab.com/)

View **page source** — Edit in **Web IDE** — please **contribute (https://gitlab.com/gitlab-com/www-gitlab-com/blob/master/CONTRIBUTING.md)**.

(https://twitter.com/gitlab) (https://www.facebook.com/gitlab) (https://www.youtube.com/channel/UCnMGQ8QHMAnVIsI3xJrihhg) (https://www.linkedin.com/co
com)