

Closed

RCA: Repository mirroring delays

RCA: Repository mirroring delays

Summary

Redis regression could've lead to the piling up of overdue mirrors as small batches of background jobs were taking considerable time to be enqueued.

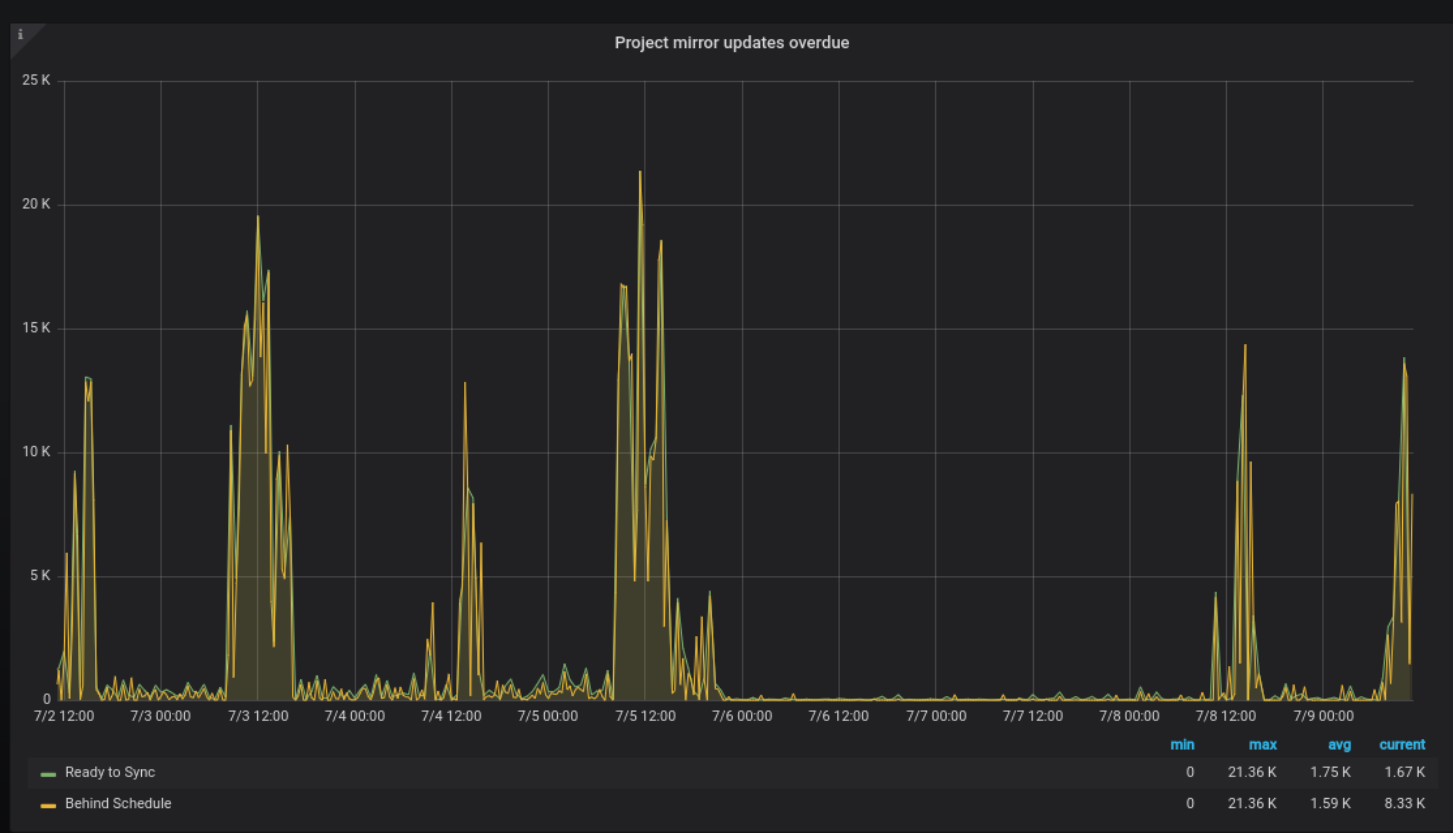
- Service(s) affected : ~"Service:Sidekiq"
- Team attribution : Infra
- Minutes downtime or degradation : 720

Impact & Metrics

Start with the following:

- What was the impact of the incident? (i.e. service outage, sub-service brown-out, exposure of sensitive data, ...)
 - Service degradation
- Who was impacted by this incident? (i.e. external customers, internal customers, specific teams, ...)
 - External customers
- How did the incident impact customers? (i.e. preventing them from doing X, incorrect display of Y, ...)
 - Delays of mirroring customers' projects
- How many attempts were made to access the impacted service/feature?
 - N/A
- How many customers were affected?
 - N/A, maximum number of delayed projects reached 13.3K project
- How many customers tried to access the impacted service/feature?
 - N/A

Provide any relevant graphs that could help understand the impact of the incident and its dynamics.



Detection & Response

Start with the following:

- How was the incident detected?
 - Through a PagerDuty alert
- Did alarming work as expected?
 - Yes
- How long did it take from the start of the incident to its detection?
 - 10 minutes
- How long did it take from detection to remediation?

- On the first day, 7 hours. On the second day, 2 hours. In both cases, solutions applied take some time to see its effect.
- V

Closed

RCA: Repository mirroring delays

Not a member of the team that created this page? Sign up now.

 - On the first day there was another incident in-flight

Timeline

See [production#937 \(closed\)](#).

Root Cause Analysis

~~The number of mirrored projects is increasing over time (12275 new projects since last month), our mirroring setup, however, has been stayed the same.~~

~~We had a maximum mirroring capacity of 960, which means that we mirror at a given a maximum of 960 projects. Assuming all 960 are finishing within one minute (the interval at which we schedule new mirroring jobs), it would still mean we won't process all projects that are due at this particular minute. For example, at 2019-07-04 16:08:00 UTC we would need to process 4889 projects, with a maximum capacity of 960 you get an overdue of 3929, add to that 5588 that are due for the next minute and you start accumulating projects very quickly. If mirroring for some projects starts lingering over one minute, you can accumulate more.~~

~~We didn't notice an improvement until we increased the maximum mirroring capacity to higher numbers and added more Sidekiq resources. Unfortunately, such improvements doesn't come for free. More Sidekiq resources can add stress to pgbouncer which can affect neighboring client connections. Increased capacity means sending larger requests to Redis (e.g. LPUSH with thousands of items) which would stress Redis as well.~~

Qr ۷ǫǦǦ B۷ N̄ Zr Āy ĞŸ ùr N̄ N̄ GŌǺŭ Ğŕ ỹ

Problem: "Project mirror updates overdue" kept rising and wouldn't go down

1. **Why?** New mirrors were becoming "ready to update" at a higher rate than we were updating mirrors that were already ready to update
2. **Why?** We were updating mirrors at a lower rate than our pullmirror nodes are capable of
3. **Why?** There were too few RepositoryUpdateMirrorWorker jobs in the repository_update_mirror queue, leaving pullmirror nodes sitting idly by when they could be updating mirrors
4. **Why?** UpdateAllMirrorsWorker couldn't schedule mirror updates fast enough to keep up with pullmirror nodes becoming ready for new work
5. **Why?** The LPUSH Redis operation used to push new jobs onto the queue took significantly more time to complete than it would and has under healthier Redis conditions

https://gitlab.com/gitlab-org/gitlab-ee/merge_requests/14573 improves things at step 4, but we are still limited by the performance of the LPUSH operation and thus the health of our Redis cluster.

What went well

Start with the following:

- Alerting was spot-on

What can be improved

- The runbook actions
- The ability to add more Sidekiq resources (constraint by pgbouncer/Postgres and to a [lesser extent, Redis](#))

Corrective actions

- [gitlab-com/www-gitlab-com#4747 \(closed\)](#).
- [gitlab-com/runbooks!1197 \(merged\)](#).
- https://gitlab.com/gitlab-org/gitlab-ee/merge_requests/14573
- <https://gitlab.com/gitlab-org/gitlab-ee/issues/12698>

Guidelines

- [Blameless RCA Guideline](#)
- [5 whys](#)

Edited 11 months ago by [Ahmad Sherif](#)

Linked issues 

 2

Relates to

Closed RCA: Repository mirroring delays

2019-07-03 & -04 & -05: Repository mirroring delays

production#937

Related merge requests 1

Expand large-pull-mirror-queue runbook

gitlab-com/runbooks!1197


✓

- Ahmad Sherif @ahmadsherif

 added [IncidentReview](#) label 1 year ago
- Ahmad Sherif @ahmadsherif

 mentioned in issue [production#937 \(closed\)](#) 1 year ago
- Christopher Lefelhocz @clefelhocz1

 mentioned in issue [gitlab-com/www-gitlab-com#4753](#) 11 months ago



Douwe Maan @DouweM · 11 months ago

Developer

5 Whys analysis over 2019-07-05's events and findings:

Problem: "Project mirror updates overdue" kept rising and wouldn't go down

1. **Why?** New mirrors were becoming "ready to update" at a higher rate than we were updating mirrors that were already ready to update

2. **Why?** We were updating mirrors at a lower rate than our pullmirror nodes are capable of


3. **Why?** There were too few RepositoryUpdateMirrorWorker jobs in the repository_update_mirror queue, leaving pullmirror nodes sitting idly by when they could be updating mirrors

4. **Why?** UpdateAllMirrorsWorker couldn't schedule mirror updates fast enough to keep up with pullmirror nodes becoming ready for new work

5. **Why?** The LPUSH Redis operation used to push new jobs onto the queue took significantly more time to complete than it would and has under healthier Redis conditions

https://gitlab.com/gitlab-org/gitlab-ee/merge_requests/14573 improves things at step 4, but we are still limited by the performance of the LPUSH operation and thus the health of our Redis cluster.

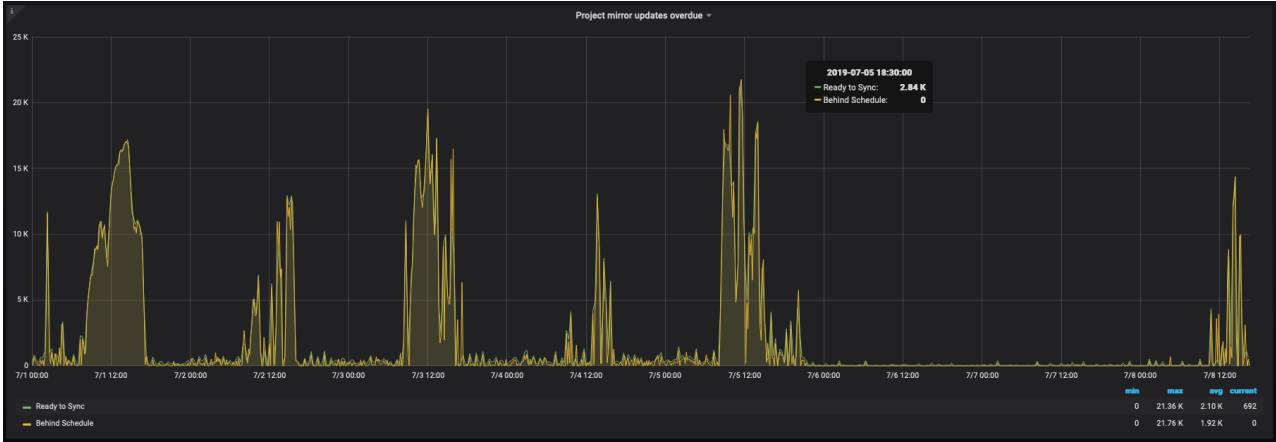
[@ahmadsherif](#) [@craigf](#) [@reprazent](#) Does this sound about right based on what we saw on Friday?



Bob Van Landuyt @reprazent · 11 months ago

Developer

Interestingly this seems to happen at about 12 UTC consistently:



[https://dashboards.gitlab.net/d/ MKRXrSmk/pull-mirrors?orgId=1&refresh=5s&fullscreen&panelId=13&from=now%2FM&to=now](https://dashboards.gitlab.net/d/MKRXrSmk/pull-mirrors?orgId=1&refresh=5s&fullscreen&panelId=13&from=now%2FM&to=now)

- Ahmad Sherif @ahmadsherif

 changed the description 11 months ago
- Ahmad Sherif @ahmadsherif

 changed the description 11 months ago



Ahmad Sherif [@ahmadsherif](#) changed the description [11 months ago](#)



Closed **RCA: Repository mirroring delays**



Ahmad Sherif [@ahmadsherif](#) changed the description [11 months ago](#)



Andrew Newdigate [@andrewn](#) mentioned in issue [#7216 \(closed\)](#) [11 months ago](#)



Anthony Sandoval [@AnthonySandoval](#) · [11 months ago](#)

Owner


[@ahmadsherif](#) please reopen if there is any additional investigation or discussion that needs to take place in this issue.




Anthony Sandoval [@AnthonySandoval](#) closed [11 months ago](#)



ops-gitlab-net  [@ops-gitlab-net](#) mentioned in issue [#7393 \(closed\)](#) [11 months ago](#)



Andrew Newdigate [@andrewn](#) marked this issue as related to [production#937 \(closed\)](#) [10 months ago](#)



Michelle Gill [@m_gill](#) mentioned in issue [gitlab-org/gitlab#12698](#) [5 months ago](#)

Please [register](#) or [sign in](#) to reply