Main page
Recent changes
Server admin log (Prod)
Server admin log (RelEng)
Deployments
SRE/Operations Help
Incident status

Cloud VPS & Toolforge
Cloud VPS documentation
Toolforge documentation
Request Cloud VPS project
Server admin log (Cloud VPS)

Tools
What links here
Related changes
Special pages
Permanent link
Page information
Cite this page

Print/export
Create a book
Download as PDF
Printable version

Incident documentation/20200505-wdqs-deploy

< Incident documentation

document status: in-review

Summary

On May 5th, 2020, all Wikidata Query Service instances were down for about 5 minutes due to a failed deployment. The Failed deployment was a result of two factors - an incorrectly executed deployment procedure and a bug in the code. Deployment that day was itself a result of a failed deployment the day before (May 4th), which in turn was caused by double jars present in the deploy repo.

Bug in the code that ultimately brought down the service (Blazegraph) was the removal of serialVersionUID field from WKTSerializer class. It turned out to be needed by Blazegraph, but that that need wasn't caught by standard integration/unit tests. Effect on the instances updated with faulty code was that Blazegraph (WDQS's backend) couldn't start.

Deployment was stopped after some number of instances started to fail and the previous version was rolled back. Because of how WDQS metrics are presented (all of the errors are rolled into one metric, even "standard" ones like user throttling), we don't know the exact number of queries affected, but metrics do not show significant impact. In any case, visibility here is problematic and needs to be addressed).

Contents [hide]

Timeline

SAL: https://sal.toolforge.org/production?p=0&q=&d=2020-05-05q=synchronized&d=2012-01-01

All times in UTC.

- 07:36 Deployment starts (set with the incorrect option for which groups to deploy) (https://sal.toolforge.org/log/Sc7D43EBLkHzneNN4wiv)
- 07:44 Nodes start to fail **OUTAGE BEGINS**
- 07:50 Deployment is stopped and nodes are starting to be rolled back (https://sal.toolforge.org/log/K2HQ43EBj_Bg1xd3Reki)
- 07:54 All nodes are rolled back and restarted - **OUTAGE ENDS**

Keep in mind that not all nodes failed, so the outage was partial.

Detection

The issue was detected by icinga and the alert was posted on #wikimedia-operations.

```
PROBLEM - Check systemd state on wdqs1003 is CRITICAL: CRITICAL - degraded: The
system is operational but one or more units failed.
https://wikitech.wikimedia.org/wiki/Monitoring/check_systemd_state
09:43 PROBLEM - Query Service HTTP Port on wdqs2003 is CRITICAL: HTTP CRITICAL:
HTTP/1.1 500 Server Error - 9597 bytes in 0.006 second response time
https://wikitech.wikimedia.org/wiki/Wikidata_query_service
09:43 PROBLEM - Query Service HTTP Port on wdqs2001 is CRITICAL: HTTP CRITICAL:
HTTP/1.1 500 Server Error - 9597 bytes in 0.008 second response time
https://wikitech.wikimedia.org/wiki/Wikidata_query_service
09:44 PROBLEM - Check systemd state on wdqs2003 is CRITICAL: CRITICAL -
degraded: The system is operational but one or more units failed.
https://wikitech.wikimedia.org/wiki/Monitoring/check_systemd_state
09:45 PROBLEM - PyBal backends health check on lvs2010 is CRITICAL: PYBAL
CRITICAL - CRITICAL - wdqs-heavy-queries_8888: Servers wdqs2002.codfw.wmnet are
marked down but pooled: wdqs-ssl_443: Servers wdqs2002.codfw.wmnet are marked
down but pooled: wdqs-internal_80: Servers wdqs2004.codfw.wmnet are marked down
but pooled: wdqs_80: Servers wdqs2002.codfw.wmnet are marked down but pooled
https://wikitech.wikimedia.org/wiki/PyBal
09:45 PROBLEM - LVS HTTP codfw IPv4 #page on wdqs.svc.codfw.wmnet is CRITICAL:
HTTP CRITICAL: HTTP/1.1 500 Server Error - 9597 bytes in 0.078 second response
time https://wikitech.wikimedia.org/wiki/LVS%23Diagnosing_problems
09:45 PROBLEM - Check systemd state on wdqs2001 is CRITICAL: CRITICAL -
degraded: The system is operational but one or more units failed.
https://wikitech.wikimedia.org/wiki/Monitoring/check_systemd_state
```

Fortunately, preexisting warnings were enough to stop deployment and rollback servers already updated with the faulty version.

As for the faulty version itself, since manual tests weren't run after the canary (what should've happen), we didn't actually know that service was faulty. There are no automatic tests during deployment that showed us that that service didn't work properly.

Conclusions

Two main things that happened during the deployment:

- the issue with the faulty version wasn't noticed before the deployment

The problem here is that the version of WDQS was faulty from the start - it wouldn't run properly after updating on any server. Unfortunately, we do not have CI/CD test servers that would have allowed us to spot the issue sooner. It seems that having automated deployments of each master version would be beneficial - we could get information if something fails much faster, e.g. by having a set of smoke tests executed there (we already have such a suite). It is worth it to point out that we already have a server (wdqs1009) that is being updated automatically, but only with the deploy repo version and we tend to deploy immediately after updating the version on that one. Version update with the master version would be more beneficial in scenarios like this one.

- tests were not run after canary deployment

During the deployment there is a step when the process halts - after deploying a canary (single server). The deployer should at this point run the manual tests to verify correctness - that didn't happen. That was caused by human error, but there is no reason why tests - which are normally executed by a person - couldn't be executed as a part of the deployment process.

After the outage, in turn, it was hard to precisely assess the impact. While graphs (dashboard) show error rates, they also count in errors that normally happen (like rejections because of throttling).

What went well?

- issue detection
- fast reaction made the incident's impact low

What went poorly?

- deployment process itself
- unknown issue with the version

Where did we get lucky?

- low impact - could've been much worse - thanks to fast response from gehel, not all instances went down. Also - while difficult to be precise with current visibility (see action item) - the graphs show that the impact

wasn't substantial.

How many people were involved in the remediation?

- 1 SRE, 1 developer and 1 engineering manager

Links to relevant documentation

Production deployment guide -
https://wikitech.wikimedia.org/wiki/Wikidata_query_service#Production_Deployment

Actionables

- Update documentation to describe manual testing procedure after canary deployment DONE
- Create automatically updated CI test environment (phab:T252503)
- Include smoke tests as a part of deployment procedure (phab:T252504)
- Improve outage visibility to see actual impact of an outage (phab:T252508)

Categories: Incident documentation | Incident documentation drafts