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/20140716-DNS

< Incident documentation

Contents [hide]

Summary

Due to the combination of an incorrectly-deployed DNS change and a bug in our DNS server software, a small fraction of users could not reliably resolve our site hostnames for a period of several hours (intermittently and of decreasing prevalence over time from approximately July 16 22:00 UTC to July 17 13:40 UTC).

Timeline

- 2014-07-16 21:36 - Deployed this DNS change: https://gerrit.wikimedia.org/r/#/c/140149/
    - This change was the removal of several service+site -specific hostnames that have not been referenced by us for several months.
    - This didn't turn out to be the real problem in this incident, it's just peripherally involved.
- 2014-07-16 22:00 - Deployed this DNS change: https://gerrit.wikimedia.org/r/#/c/140391/
    - This changes a ton of CNAME entries to point at a new canonical name "text-lb.wikimedia.org" and also adds that new canonical name in the same change
    - This caused the actual problem, as detailed below
- 2014-07-16 ~23:00 - A handful of isolated reports of DNS issues came in via IRC
    - Although traffic stats confirmed the majority were unaffected, still troubling and warranted investiation/response
- 2014-07-17 00:15 - Reverted the first change above in https://gerrit.wikimedia.org/r/#/c/146990/ as a possible fix
    - It seemed the more likely of the two to cause issues at the time when the analysis below wasn't yet clear.
    - Some reporters indicated their problem was solved after this change, but that was likely just a coincidence.
    - I don't believe that change was actually causing issues, but it has been left in the reverted state for now.
- 2014-07-17 12:02 - A user filed a bugzilla bug reporting DNS problems from an ISP in Germany (https://bugzilla.wikimedia.org/show_bug.cgi?id=68162).
    - Solid evidence that the problem was still not resolved in all cases.
    - This lead directly to the re-analysis and decision to revert below.
    - The user put us in touch with admins at his ISP (Hetzner.de), which helped greatly with later post-analysis by providing technical details on their DNS cache software and its behavior.
- 2014-07-17 12:40 - Mostly-reverted the second change above in https://gerrit.wikimedia.org/r/#/c/147095/ , as it was now clear that this change was the real problem.
    - The reason for the incomplete revert was that, at this point, it was better to leave the new text-lb.wm.org hostname in place to avoid confusing caches further, while still reverting all the CNAMEs that pointed at it.

Conclusions

The specific problem in the faulty change (140391) was that it created a temporary DNS race condition. Ultimately, we cannot deploy a DNS change that involves such a two-step dependency as if it were a single "transaction", because our DNS servers receive constant traffic and the process of updating our DNS servers is slightly (on the order of a few seconds) asynchronous. The race condition happens as follows:

- Most of our production DNS resolution involves a pair of CNAMEs which cross zone boundaries (e.g. en.wikipedia.org -> wikipedia-lb.wikimedia.org -> text-lb.esams.wikimedia.org -> $actual_ip)
- When a CNAME chain crosses a zone boundary like that, a resolving cache has to resolve it in two steps with two separate queries, and may use different, arbitrary authoritative servers of ours for each query.
- Therefore in a change like this, during a very brief window while the change is rolling out, the first nameserver affected may start serving "en.wp.org -> text-lb.wm.org", while the second nameserver still doesn't have the update that creates "text-lb.wm.org", and so it's possible for the querying cache to end up with an NXDOMAIN during this brief period.

Even in a case like this where we've created a race condition resulting in NXDOMAIN, there are generally two mitigating factors working in our favor:

- The race-window is very small relative to the TTLs that matter. The TTLs that matter in this case are 1-hour long, and the race window was a few seconds. Therefore any given cache only has ~1/1000 chance of being caught in the race and getting the faulty NXDOMAIN response.
- For those caches caught in the race, the faulty NXDOMAIN should have expired and self-corrected for each cache within at most 10 minutes (the duration of wikimedia.org's negative caching TTL).

However, in our case some smaller subset of the caches continued to cache the NXDOMAIN for hours instead of minutes. This ultimately turned out to be a bug in our authoritative server (not a true code-level bug, but more an "author didn't understand the specification" bug). This resulted in our authservers communicating a negative cache TTL of 24 hours instead of 10 minutes, thus exacerbating the impact.

Impact Scope/Depth

We can only speak in terms of rough approximations here, as we don't have direct insight into client DNS caches and client results. Consider everything here approximations to about an order-of-magnitude based on analysis:

- The vast majority of users were unaffected, as indicated by no significant long-term loss of traffic in our LVS statistics, etc.
- Up to ~0.08% of caches, and thus up to ~0.08% of users potentially initially affected (based on the math of the race-condition window timing of the deployment process vs TTL) received an NXDOMAIN for most of our major domainnames for some variable length of time beginning around 2014-07-16 22:00.
- I suspect if that approximation is off, it is on the high side, and our traffic stats seem to agree. There are various minor factors working in our favor (e.g. that caches aren't always going to expire both related records at the same instant, locality preference among NS lists, etc).
- The most common case for the end time of the problem for arbitrary affected caches/users would be 3 hours later at 2014-07-17 01:00, because the most common cache implementation on the internet (BIND) limits negative cache TTLs to 3 hours by default.
- Some caches may have been manually fixed relatively quickly by administrators - especially larger ISP/shared caches would likely have received user reports about the issue and the admins would have cleared the related cache entries to fix.
- Depending on cache software used and local configuration, the end time for less-common caches without active intervention is highly-variable, with a maximum limit at approximately 2014-07-17 13:40 (when we deployed the revert + up to 1 hour positive TTL).

Actionables

- Status: 🟧 **on-going** - Clearly, the core issue here was a faulty change that shouldn't have passed review. It was part of a series of changes that started review a month ago, and had been vetted for similar TTL-related issues resulting in some changes and commit-splits to avoid these things, but we missed one.
- Status: ⬜ **Declined** - We should simplify the structure of our key production DNS data, avoiding the double queries via domain-crossing CNAME chains and simply returning the final address directly. The simplification at the zonefile level would make it harder to make mistakes in editing our DNS to begin with, and the lack of double-queries greatly reduces the chances of races of this nature. It would also reduce our DNS load and any DNS-specific part of our site's initial page load latency. The series of changes that triggered this outage were already moving in the direction of simplification, and I think in the future we should go further. There may be some debate to be had on this topic, as switching our public resolution to returning direct addresses removes some debugging information from the responses (those debugging DNS queries would no longer see the loadbalancer and site name textually in the response data, but could figure it out from the IP address).
    - specific steps?

- Status: 🟩 **Done** - There's a real bug in our authserver implementation causing large negative cache TTLs in practice. It's the worst kind of bug: one that only bites you when you make a mistake. I've already committed a fix for this upstream, and we'll be packaging and deploying a release with that fix early next week. -- Update: this was upgraded to apply the fix during the week of July 21.

Category: Incident documentation