DNS issues in Kubernetes. Public postmortem #1

Amet Umerov | Follow
May 4 · 4 min read

In this article, I'll share our experience with postmortems at Preply.

Here's an example of one of our latest incidents with DNS on production in the form of a postmortem. The article could be helpful for those who want to know more about postmortems or want to prevent DNS issues in the future.

My name is Amet Umerov and I'm a DevOps Engineer at Preply.com. Let's get started!

. . .

A little bit about postmortem and processes at Preply

> A postmortem describes a production outage or paging event including a timeline, description of user impact, root cause, action items, and lessons learned.
>
> Seeking SRE. By David N. Blank-Edelman

On weekly dev meetings with pizza, we share information between technical teams. One of the important parts of these meetings is postmortem sharing. Sometimes it's accompanied by an additional presentation with slides and a more detailed analysis of the incident. Of course, we have blameless culture, but we don't clap for postmortem :) The main reason why write and present postmortems to the team is because we believe sharing knowledge can prevent problems in the future.

> The individuals involved in a post mortem must feel that they can give this detailed account without fear of punishment or retribution. No finger pointing! Writing a post mortem is not a punishment — it is a learning opportunity for the entire company.
>
> Keep CALMS & DevOps: S is for Sharing

DNS issues in Kubernetes cluster Postmortem

Date: 2020-02-28

Authors: Amet U., Andrii S., Igor K., Oleksii P.

Status: Complete

Summary: Partial DNS outage (TTM is 26 min) for some services inside the Kubernetes cluster

Impact: 15000 events dropped for services A, B, and C

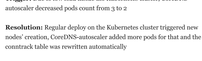
Root causes: Kube-proxy didn't successfully delete an old row from conntrack table, and some services were still routed to the nonexistent pods

```
E0228 20:13:53.795782       1 proxier.go:638] Failed to delete kube-system/kube-dns:dns endpoint connections, error: error deleting conntrack entries for UDP peer {100.64.0.10, 100.110.33.231}, error: conntrack command returned: ...
```

Trigger: Due to low load inside the Kubernetes cluster, CoreDNS-autoscaler decreased pods count from 3 to 2

Resolution: Regular deploy on the Kubernetes cluster triggered new nodes' creation, CoreDNS-autoscaler added more pods for that and the conntrack table was rewritten automatically

Detection: Prometheus monitoring detected a high 5xx rate on services A, B and C and paged on-call



5xx rate in Kibana

Action Items

Action Item	Type	Owner	Task
Disable autoscaler for CoreDNS	prevent	Amet U.	DEVOPS-695
DNS cache server on prod	mitigate	Max V.	DEVOPS-665
Conntrack usage monitoring	prevent	Amet U.	DEVOPS-674

Lessons learned

What went well:

- Monitoring detection was quick. The reaction was rapid and organized
- We didn't reach any limits on nodes

What went wrong:

- We still don't know the real root cause, it seems to be a specific bug with conntrack
- All action items only fixed consequences, not the root cause
- We knew possible issues with DNS but didn't prioritize it

Where we got lucky:

- Another deploy has triggered CoreDNS-autoscaler, and conntrack table was rewritten
- The issue didn't affect all services

Timeline (EET)

Time	Action
22:13	CoreDNS-autoscaler scaled-down pods 3->2
22:18	On-call engineers started to get calls from VictorOps
22:21	On-call engineers started troubleshooting
22:39	On-call engineers reverted latest release of the service
22:40	5xx errors had gone, situation becomes stable

- **Time to Detect:** 4 min
- **Time to Engage:** 21 min
- **Time to Fix:** 1 min

Supporting information

- CoreDNS logs:

```
I0228 20:13:53.507788       1 event.go:221]
Event(v1.ObjectReference{Kind:"ConfigMap", Namespace:"kube-system",
Name:"coredns", UID:"2493ab55-3e3b-11ea-b3a2-42004ff8c23f",
APIVersion:"apps/v1", ResourceVersion:"13209808", FieldPath:""}):
type: 'Normal' reason: 'ScalingReplicaSet' Scaled down replica set
coredns-6c00604c9 to 2
```

- Kibana link (redacted), Grafana link (redacted)
- Where Linux conntrack is no longer your friend
- kube-proxy Subtleties: Debugging an Intermittent Connection Reset
- Racy conntrack and DNS lookup timeouts

To minimize CPU utilization, the Linux kernel uses features like conntrack. Basically, it's a utility that contains a list of NAT records in the table. When the next packet comes from the same source pod to the same destination pod, it won't be translated again for the CPU economy, the destination IP address will be taken from the conntrack table.

How the conntrack works

Summary

So, it was an example of one of our postmortems with some useful links, hope it will be helpful for you.

In this particular case, the information we shared can be useful for other companies. That's why we're not afraid of making mistakes and that's why we decided to make one of our postmortems public. Here are a few more interesting public postmortems:

- GitLab: Postmortem of database outage of January 31
- Dropbox: Outage post-mortem
- Spotify: Spotify's Love/Hate Relationship with DNS
- Many others from this GitHub paste and Kubernetes Failure Stories's repo
- And an example of public postmortem with great structure from Google's SRE book

Subscribe to the Preply Engineering Blog for more interesting articles about engineering at Preply. Stay tuned!

Thanks to Andrii Dvoiak.

DNS | Postmortem | DevOps | Kubernetes | Conntrack

WRITTEN BY
Amet Umerov
DevOps Engineer at preply.com
Follow

Preply Engineering Blog
We are hiring in Barcelona/Kyiv
https://preply.com/en/careers
Follow