Google Cloud

QUIC Exit

Exposing a new class of outage

August 2022

Ian Swett, Andy Sykes

Terminology

GFE, aka Google Front End, is an L7 load balancer that terminates user traffic
Google GFEs serve Google domains (google.com, youtube.com, etc)
Cloud GFEs serve Google Cloud Load Balancer domains

Uberproxy is an HTTP proxy used to access internal resources (paper)
ie: Monitoring, Incident Response, Tools to modify configs, Push jobs

QUIC is a new transport protocol build on UDP. HTTP/3 runs over QUIC.
QUIC (RFC9000), HTTP/3 (RFC9114)

gQUIC is Google's experimental protocol which inspired the standardization of IETF QUIC.

SRE is a Site Reliability Engineer, responds to pages within 5 minutes 24/7.

Summary

Query of death triggered by resumption information sent *from GFEs to clients and back to GFEs* caused GFEs to crash.

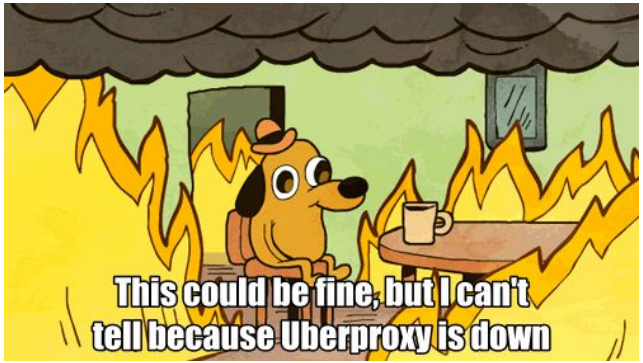
At peak around **10% of Google GFEs** were crashing, but this distribution was very uneven.

Impact was mostly limited to Europe, and to services served from datacenters.

Total outage time was **1h 44m**.

tl;dr

- **Query of death (QoD):** a nullptr deref that was introduced in 2014(!)
- **Novel QoD path:** *contagion* occurred via clients.
- **Looked terrifying:** SREs couldn't access monitoring tools because UberProxy was down, thought everything was down globally.
- **Slow to diagnose, fast to mitigate:** couldn't get to internal monitoring - diagnosis took ~1h; mitigation took 17m.
- **Incredible, barely-believable luck:** GFEs in datacenters crashed but edge GFEs almost untouched, www.google.com kept on truckin'.



One of the authors, at 9am, after being paged.

What is a Contagion Bug?

Contagion: An interaction of distributed systems

Slow rollouts identify most bugs before significant harm

- If a bug is found, roll back.

Contagion bugs are **not** fixed by rollbacks alone.

A single task could cause a global outage.

- Persistent state in another system is not rolled back.
- In the case of internet clients, cannot rollback.

Contagion: Not just crashes and clients

Almost **any** type of bug:

Incorrect responses, Memory leaks, Infinite loops, Crashes...

Any **two or more** large scale distributed systems:

Browser to Server

Server to Server

Could take time to notice and not be caught in Canary

Example: TLS or QUIC Resumption

- TLS resumption
- QUIC source address tokens
- gQUIC server configs

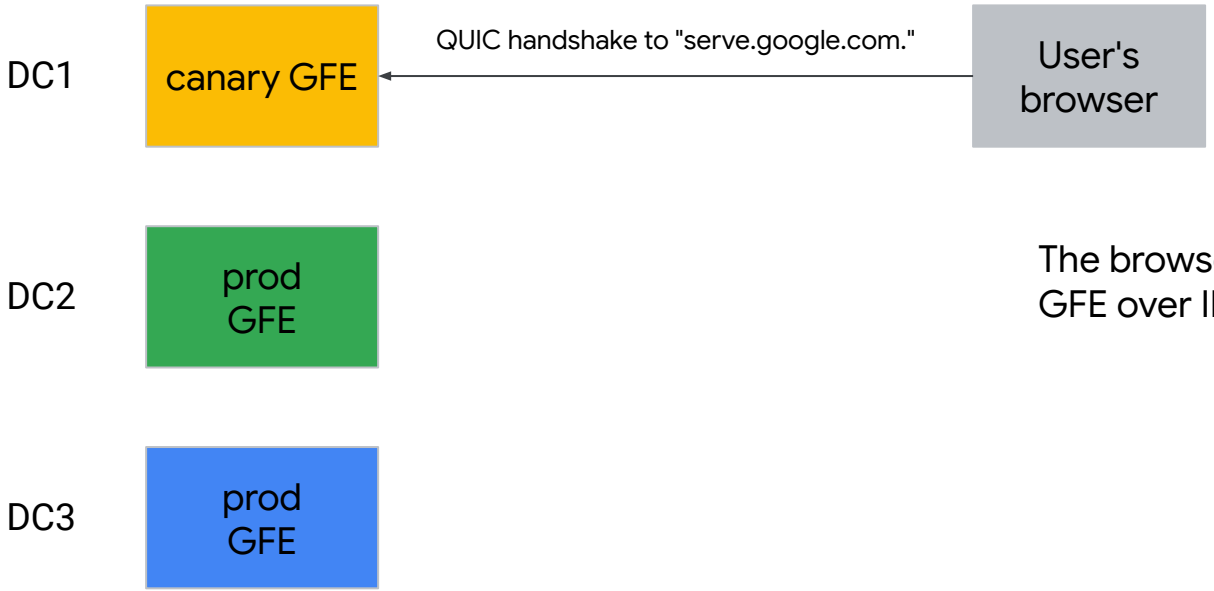
One GFE gives the client information for a future connection

Another GFE parses it later and something goes wrong.

What happened at Google in November 2021

Google Cloud

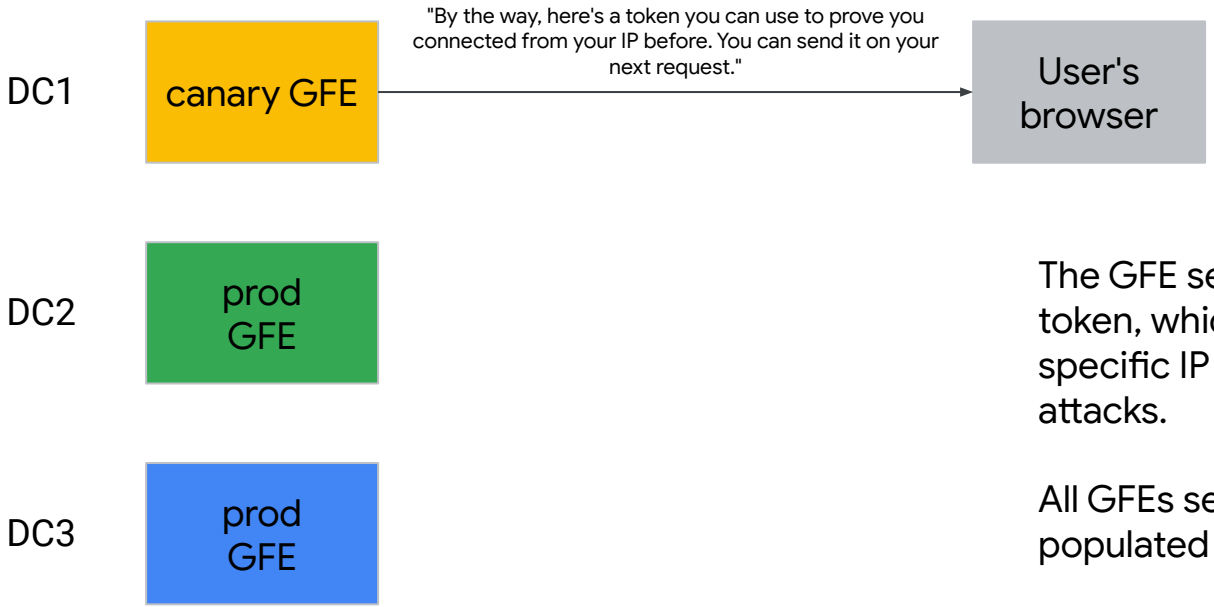
Mechanism of action



The browser completes a handshake to a GFE over IETF QUIC.

Source: Lorem ipsum dolor sit amet, consectetur adipiscing elit. Duis non erat sem

Mechanism of action

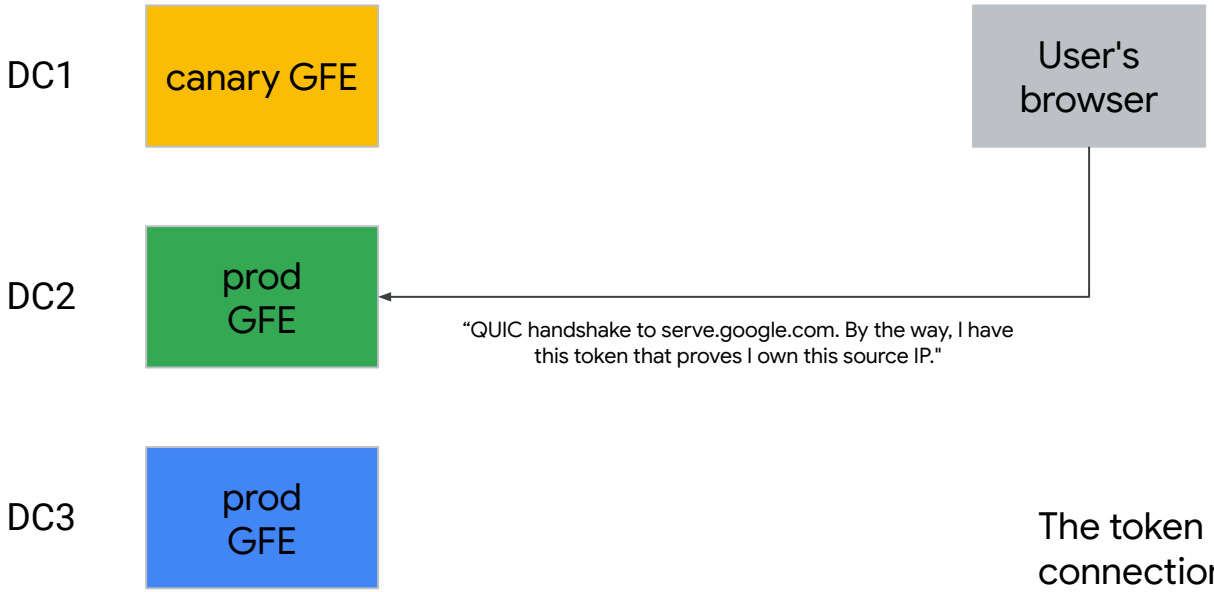


The GFE sends the browser an encrypted token, which proves a client owns a specific IP address, limiting amplification attacks.

All GFEs send this token, but canary jobs populated a new field.

Source: Lorem ipsum dolor sit amet, consectetur adipiscing elit. Duis non erat sem

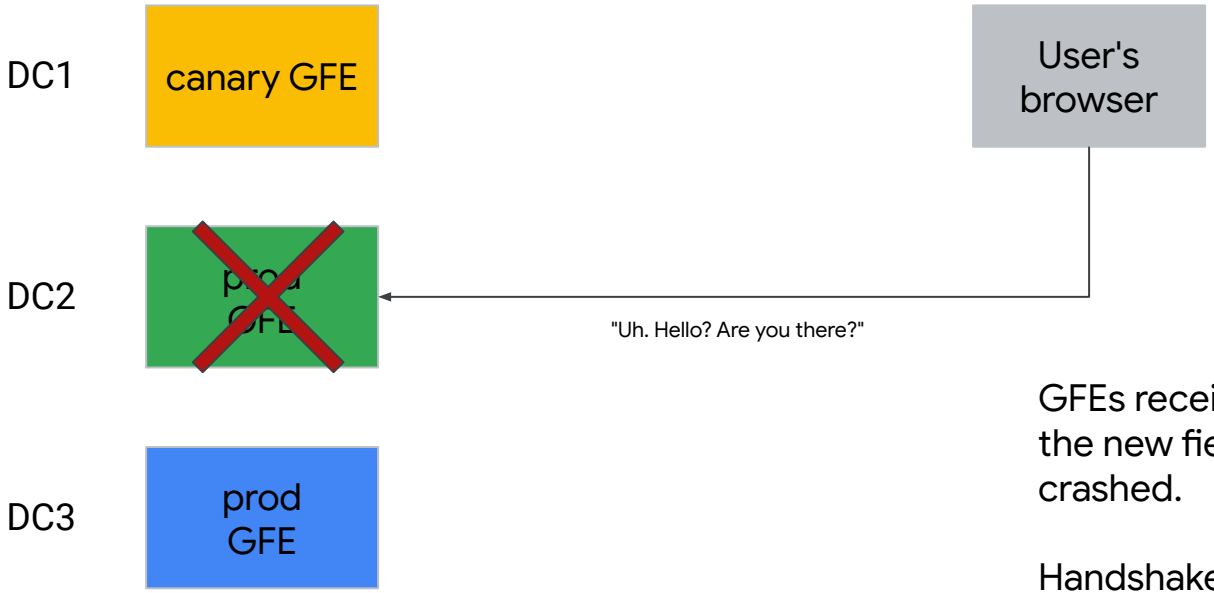
Mechanism of action



The token is sent by the client on the next connection; after a handshake, the token *should* be cleared and was if the handshake completed.

Source: Lorem ipsum dolor sit amet, consectetur adipiscing elit. Duis non erat sem

Mechanism of action

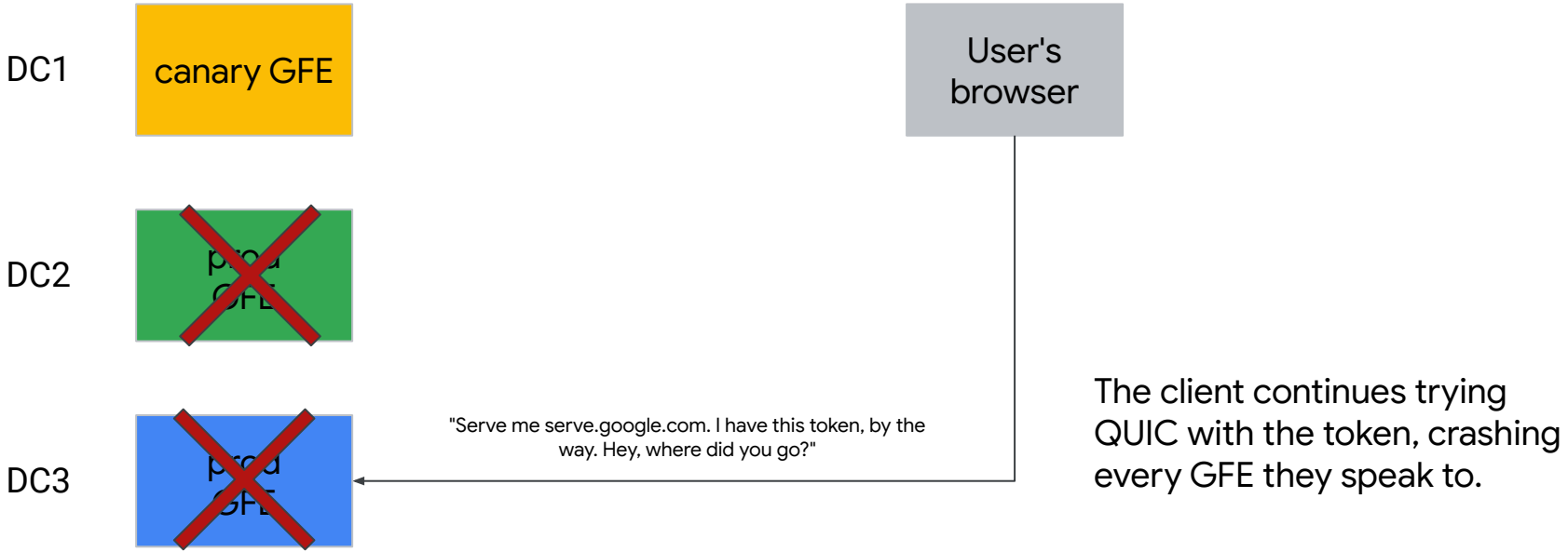


GFEs receive the IETF QUIC token with the new field, dereferenced a nullptr, and crashed.

Handshake doesn't complete, so due to a bug, the client keeps using the "poison" token.

Source: Lorem ipsum dolor sit amet, consectetur adipiscing elit. Duis non erat sem

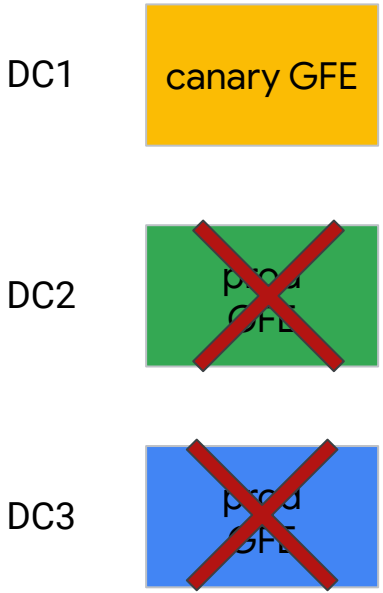
Mechanism of action

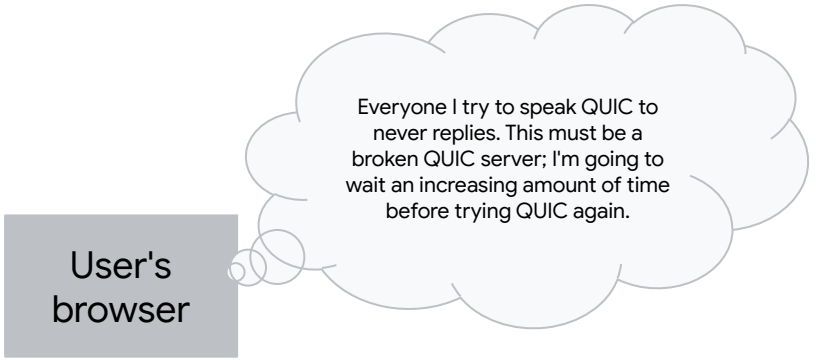


The client continues trying QUIC with the token, crashing every GFE they speak to.

Source: Lorem ipsum dolor sit amet, consectetur adipiscing elit. Duis non erat sem

Mechanism of action





When Chromium clients see a handshake failure, they mark QUIC is "broken", and go into exponential backoff. 5 minutes... 10 minutes...

Source: Lorem ipsum dolor sit amet, consectetur adipiscing elit. Duis non erat sem

From canary to resolution

00:27 PST

4 Canary GFEs receive updated flags

GFEs in Europe begin crashing

00:31 PST

Probers fail and SREs alerted by pages

Canary judge automatically rolls back flags after 4 minutes

00:42 PST

European GFEs continue crashing

London SREs realize all monitoring tools, including crashlogs, are inaccessible

01:38 PST

London SREs learn it's not a global outage

India, NZ SREs reroute all European UberProxy traffic

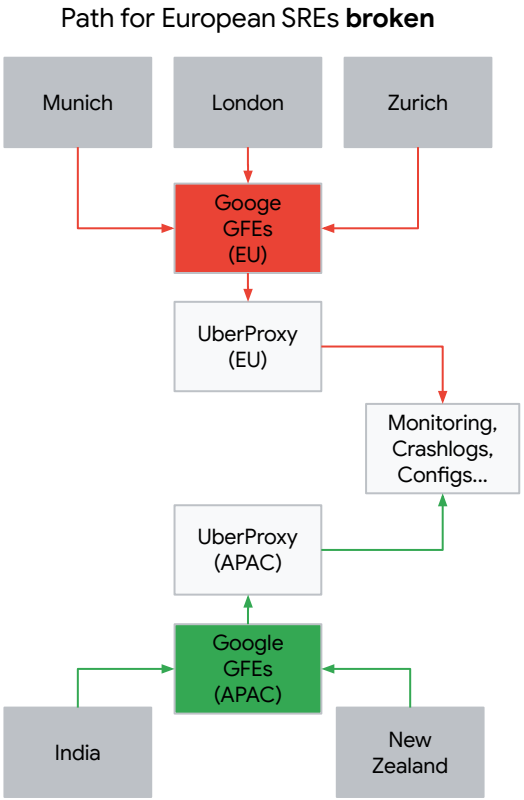
01:51 PST

SREs disable QUIC

Page me at 6am to figure out what happened

Why did it take so long to mitigate?

- UberProxy is only served from datacenters; European SREs terminate on Google GFEs in Europe, but these were effectively down due to the crash rate, so UberProxy was barely accessible.
- With no crashlogs, the only information was that Google GFEs were crashing and probers were failing.
- We got lucky we had a few SREs in other locations awake and helping (e.g. India, New Zealand).
- We lost **more than an hour** because of UberProxy unavailability.



Google Cloud

Impact

- Crashes outside Europe were rare, so there was very little impact outside of Europe.
- Crashes almost entirely in datacenters, but most services are served from edge.
- Datacenter queries per second dropped by 0.3% globally; nothing visible in edge.
- GMail, Groups, Tasks, Calendar, and Chat were impacted (Incident Report).
- UberProxy particularly impacted because it relies upon long-lived connections.
- Made it to HN, reports in press about "Gmail down".
    - Actually referred to SMTP and IMAP, as these are only hosted in datacenters.
- Not as bad as it first looked!

What went well

- SREs were paged almost instantly.
- Fast, effective escalation from oncallers to entire team.
- Root-cause analysis was fast once stack traces were available.
- Rollback was fast, understandable and effective.
- Having Cloud GFEs separate from Google GFEs meant no impact to Google Cloud Load Balancer.

Conjecture: If UberProxy hadn't gone down, the outage would have lasted ~20m.

Incredible, barely believable luck

- The bad push was active for **FOUR MINUTES** on **FOUR TASKS** and yet managed to crash 10% of GFEs.
- Automated rollbacks caught a crash of a Canary GFE that hadn't received the flags **yet**.
- Cloud GFE canary rollout hadn't started.
- Didn't push to edge; if edge GFEs were handing out poison tokens, we'd have had a partial global outage for www.google.com, www.youtube.com ....

Other lucky breaks, including but not limited to:

- Meet kept trucking because media is not served by GFE; SREs were able to use Meet as a warroom.
- QUIC Big Red Button worked, despite not being used since November 2017.
- SREs outside Europe could use production tools to tell us what was happening.

Follow-up Action Items

Prevent programming errors

- Static analysis looking for common C++ antipatterns.
- Add extensive and targeted handshake fuzzing.

Preventing contagion

- Test new version of GFE against prior with realistic clients.

Mitigating Contagion

- Regionalize resumption.
- Canary Judge GFEs near the one being updated.

Ensure SREs can Respond

- Do not serve Uberproxy from GFE canary locations.
- Do not put canary locations close to large SRE offices.
- Improve our access-without-Uberproxy workflow.

Virtually all distributed systems could be at risk.