# Learning From Our Miztakes

# Helpful or Evasive?

We're currently experiencing degraded performance issues with [redacted]. Our team is currently working to restore normal performance levels. We apologize for any inconvenience. Users may be affected.

Aug 22, 2021

**A subset of customers in Australia are unable to access** [redacted]

**Resolved** - We received reports and resolved the issue causing a subset of customers in Australia unable to access [redacted]

At 15:56 PM PT, we received the reports of a subset of Australian customers unable to access [redacted]
At 16:18 PM PT, our engineering team resolved this issue.
Aug 22, 16:28 PDT

"We continue to investigate the issue affecting data services," a [redacted] spokesman said. "Many of our services are starting to restore. We're sorry for any issues caused and will provide an update when we know more."

The company is still investigating the cause of the outage.

# Why are these reports so evasive?

- Do the service providers see outage reports as an admission of some form of liability?
- Do they think that an open and direct appraisal of faults in their service will cause them reputational harm in the eyes of their customers?
- Or do they think that describing the causes of an outage somehow compounds their potential liabilities?
- What are they covering up here?

# Helpful or Evasive?

Early today [redacted] was down or unreachable for many of you for approximately 2.5 hours. This is the worst outage we've had in over four years, and we wanted to first of all apologize for it. We also wanted to provide much more technical detail on what happened and share one big lesson learned.

The key flaw that caused this outage to be so severe was an unfortunate handling of an error condition. An automated system for verifying configuration values ended up causing much more damage than it fixed.

The intent of the automated system is to check for configuration values that are invalid in the cache and replace them with updated values from the persistent store. This works well for a transient problem with the cache, but it doesn't work when the persistent store is invalid.

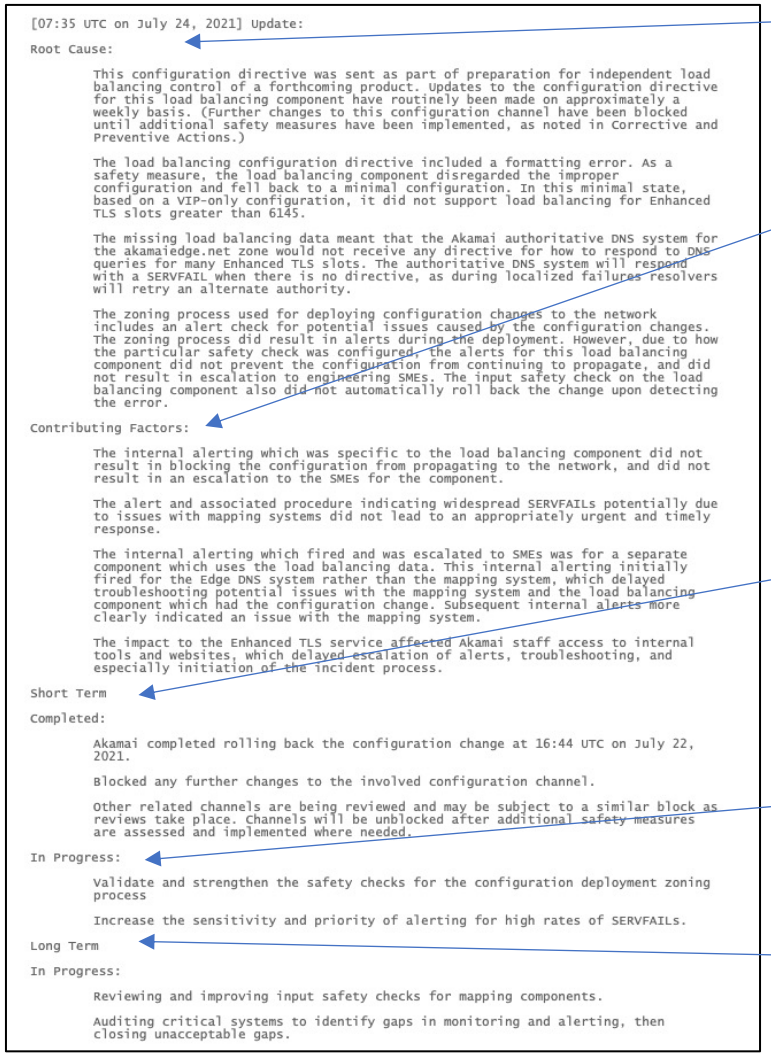
Today we made a change to the persistent copy of a configuration value that was interpreted as invalid. This meant that every single client saw the invalid value and attempted to fix it. Because the fix involves making a query to a cluster of databases, that cluster was quickly overwhelmed by hundreds of thousands of queries a second.

To make matters worse, every time a client got an error attempting to query one of the databases it interpreted it as an invalid value, and deleted the corresponding cache key. This meant that even after the original problem had been fixed, the stream of queries continued. As long as the databases failed to service some of the requests, they were causing even more requests to themselves. We had entered a feedback loop that didn't allow the databases to recover.

The way to stop the feedback cycle was quite painful - we had to stop all traffic to this database cluster, which meant turning off the site. Once the databases had recovered and the root cause had been fixed, we slowly allowed more people back onto the site.

This got the site back up and running today, and for now we've turned off the system that attempts to correct configuration values. We're exploring new designs for this configuration system following design patterns of other systems at [redacted] that deal more gracefully with feedback loops and transient spikes.

# Helpful or Evasive?



[07:35 UTC on July 24, 2021] Update:

Root Cause:

This configuration directive was sent as part of preparation for independent load balancing control of a forthcoming product. Updates to the configuration directive for this load balancing component have routinely been made on approximately a weekly basis. (Further changes to this configuration channel have been blocked until additional safety measures have been implemented, as noted in Corrective and Preventive Actions.)

The load balancing configuration directive included a formatting error. As a safety measure, the load balancing component disregarded the improper configuration and fell back to a minimal configuration. In this minimal state, based on a VIP-only configuration, it did not support load balancing for Enhanced TLS slots greater than 6145.

The missing load balancing data meant that the Akamai authoritative DNS system for the akamaiedge.net zone would not receive any directive for how to respond to DNS queries for many Enhanced TLS slots. The authoritative DNS system will respond with a SERVFAIL when there is no directive, as during localized failures resolvers will retry an alternate authority.

The zoning process used for deploying configuration changes to the network includes an alert check for potential issues caused by the configuration changes. The zoning process did result in alerts during the deployment. However, due to how the particular safety check was configured, the alerts for this load balancing component did not prevent the configuration from continuing to propagate, and did not result in escalation to engineering SMEs. The input safety check on the load balancing component also did not automatically roll back the change upon detecting the error.

Contributing Factors:

The internal alerting which was specific to the load balancing component did not result in blocking the configuration from propagating to the network, and did not result in an escalation to the SMEs for the component.

The alert and associated procedure indicating widespread SERVFAILs potentially due to issues with mapping systems did not lead to an appropriately urgent and timely response.

The internal alerting which fired and was escalated to SMEs was for a separate component which uses the load balancing data. This internal alerting initially fired for the Edge DNS system rather than the mapping system, which delayed troubleshooting potential issues with the mapping system and the load balancing component which had the configuration change. Subsequent internal alerts more clearly indicated an issue with the mapping system.

The impact to the Enhanced TLS service affected Akamai staff access to internal tools and websites, which delayed escalation of alerts, troubleshooting, and especially initiation of the incident process.

Short Term

Completed:

Akamai completed rolling back the configuration change at 16:44 UTC on July 22, 2021.

Blocked any further changes to the involved configuration channel.

Other related channels are being reviewed and may be subject to a similar block as reviews take place. Channels will be unblocked after additional safety measures are assessed and implemented where needed.

In Progress:

Validate and strengthen the safety checks for the configuration deployment zoning process

Increase the sensitivity and priority of alerting for high rates of SERVFAILs.

Long Term

In Progress:

Reviewing and improving input safety checks for mapping components.

Auditing critical systems to identify gaps in monitoring and alerting, then closing unacceptable gaps.

Root Cause Analysis – What Happened

Contributary Factors – What exacerbated the incident?

Short Term – what we did to fix it

In Progress – what we doing right now

Long Term – how we intended to avoid repeating this outage

# Why is this important?

- The internet is now the foundation for all of our communications - from the trivial and frivolous through to vital and life saving systems
- This is now a public safety issue, and we need to move away from the handling of operational incidents as PR exercises and take steps that other industries have already embraced

# What are we doing about it?

- Time to call up Sean and Jared and have a chat