Gancho Tenev

employer:

Apple Inc

ats committer known for:

cachekey, s3_auth_v4, debugging + fixing

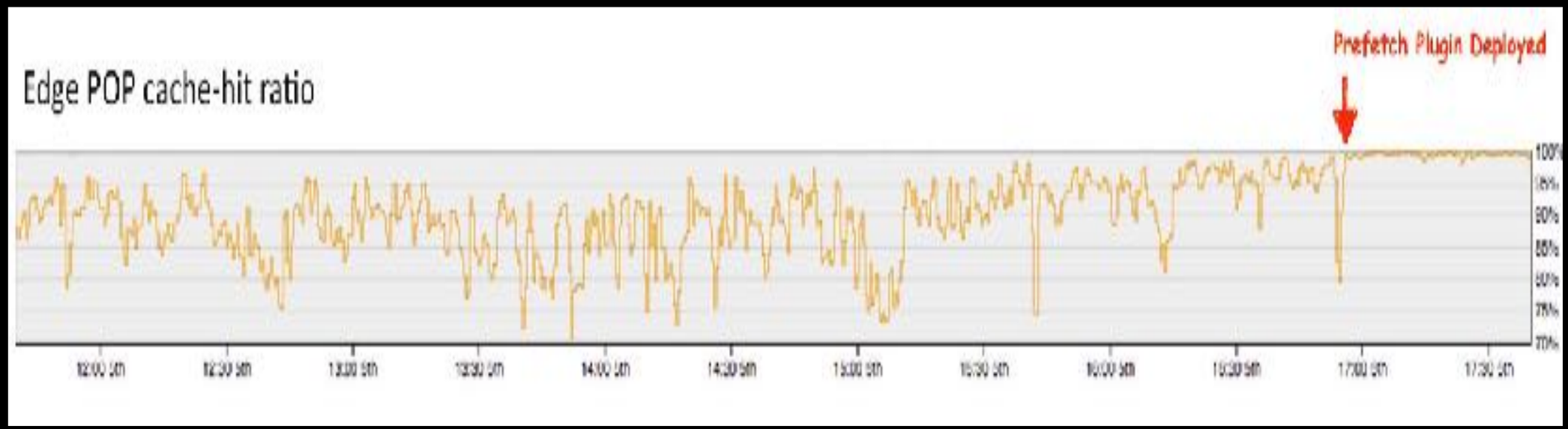
contact:

gancho@apache.org

Prefetch plugin

Apache Traffic Server plugin to prefetch content based on predefined URI patterns

Why?



This is what a failing seeding process looks like, cache-hit ratio dropped far from the targeted 100% while periodically seeding a huge collection of assets. Possible seeder software bugs, seeding process problems, unexplained 5xx errors from ATS.

Prefetch plugin was meant to compliment seeding and act as a safety net.

Generic prefetch? YES!

- Identify URI patterns and configure a series of prefetch-definitions which are matched against each request URI
- Ability to schedule multiple consecutive prefetches on each request
- The primary use-case is multi-tiered setup where a consistent URI hashing is used for parent selection but a single-tier use case is also supported

next = n+1 is simple right? not really

- Addressed lots of concerns about the feasibility of the solution, worries about the prefetch overhead, scalability and operation
- Based design and implementation decisions on real traffic analyses and lots of experimentation
- Performed real traffic simulations and significant stress testing before production
- Great results straight from the first version. Problem-free deployment and operation for almost 2 years now.

How to keep it sane ?

- cancel unnecessary prefetches at every chance we get
  - cancellation points: current object and next object cache lookup, prefetch scheduling and prefetch execution, on both the front and the backend tiers, always deduplicate prefetches.

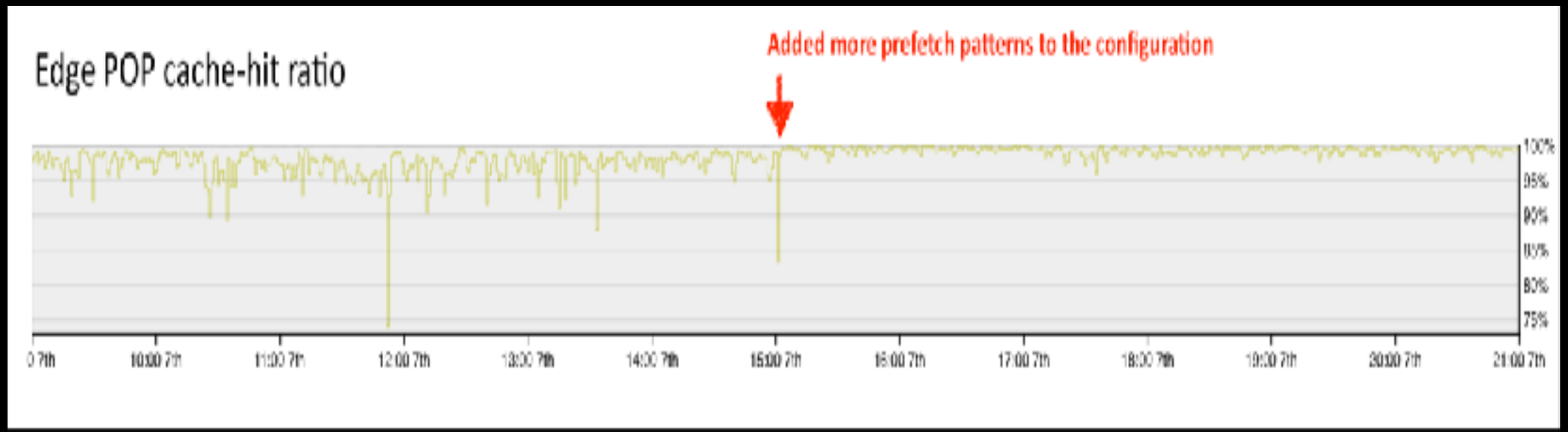
- predefined prefetch policies can be configured and enhanced if necessary in the future

- avoid expensive cache lookups by maintaining a special LRU-like structure

- cache key is always used for lookups and not URI (use cachekey plugin!)

- multi-tiered setup - only prefetch signaling between the tiers instead of actual object fetches.

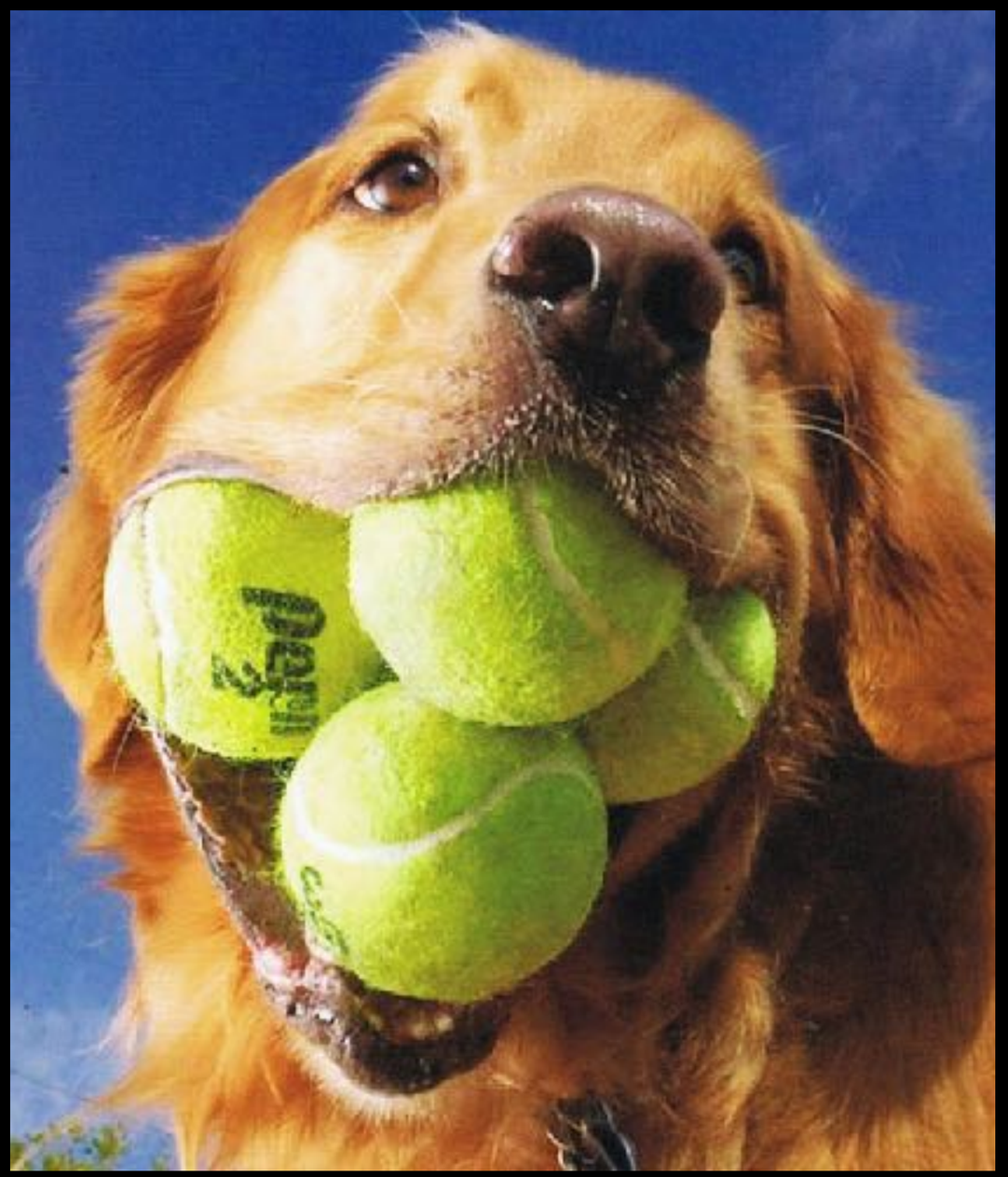
Can it run without seeding? YES!



- Tested on the worst POP
  - fresh installation (empty cache)
  - no seeding
  - low traffic, no obvious popular content
  - worst upstream network connectivity
- Made it the best POP!
  - worked well from the start
  - small code tweaks, identified and configured more prefetch patterns
  - squeezed out almost 100% cache-hit ratio (what was left was mainly traffic with no obvious URI pattern)

Don’t seed, just prefetch

- Disabled seeding on all Edge POPs and left prefetch plugin on its own.
- Observed no regression.
- Lower CPU usage, reduced cache churn, and lower bandwidth usage



Future?

- New ways to configure and define patterns. Currently using regex capture/replace + simple addition/subtraction, may be using LUA next?
- Get prefetch hints from UA or the Origin. May be use “Web Linking” headers (RFC 5988)?
- Why not automatically identify patterns?

More info?

- Documentation: https://docs.trafficserver.apache.org/en/8.0.x/admin-guide/plugins/prefetch.en.html

- Source code: https://github.com/apache/trafficserver/tree/master/plugins/experimental/access_control

- Contact: gancho@apache.org