Growing pains: Learning Media

Radu Ciorba {radu@devrandom.ro}

February 25, 2015

How it all started

- Feruary 2012.
- Started behind schedule, with a hard deadline for the next shool year.
- With an existing data model.
- Initial push was for feature completness. We'll refactor after the release, they said.

Upfront decisions!

- On AWS. Using rightscale. Variable number of machines.
- Apache with mod_wsgi.
- MySQL :(
- We launched with a memcached-backed DB Queryset Cache(johnny-cache). This bought us some time.
- Celery for really slow operations.

Then there was search

- Initially on SOLR.
- Every time you see a listing on the site, it's actually a search.

Then there was search

- Initially on SOLR.
- Every time you see a listing on the site, it's actually a search.
- Index kept gowing as our content library grew.
- Single point of failure.
- Switch to ElasticSearch!
- Just a different Haystack backend, it just works.
- ES is a breeze to cluster.

Then there was search

- Initially on SOLR.
- Every time you see a listing on the site, it's actually a search.
- Index kept gowing as our content library grew.
- Single point of failure.
- Switch to ElasticSearch!
- Just a different Haystack backend, it just works.
- ES is a breeze to cluster.
- Search is slower. A lot slower.

ES, Y U No Fast?

- Log slow queries.

ES, Y U No Fast?

- Log slow queries.
- Reproduce them with CURL.

ES, Y U No Fast?

- Log slow queries.
- Reproduce them with CURL.
- Response body is massive! What gives!

ES, Y U No Fast?

- Log slow queries.
- Reproduce them with CURL.
- Response body is massive! What gives!
- Turns out haystack stores fields you asked not to be stored.

ES, Y U No Fast?

- Log slow queries.
- Reproduce them with CURL.
- Response body is massive! What gives!
- Turns out haystack stores fields you asked not to be stored.
- Fork it, fix it, write tests, send PR that gets ignored forever. YEY!

1 kid from a school in NY with an F5 key brought down the site.

- Wollow in shame.

1 kid from a school in NY with an F5 key brought down the site.

- Wollow in shame.
- Suggest they remove all F5 keys :)

1 kid from a school in NY with an F5 key brought down the site.

- Wollow in shame.
- Suggest they remove all F5 keys :)
- Write some locust test to reproduce the issue.

1 kid from a school in NY with an F5 key brought down the site.

- Wollow in shame.
- Suggest they remove all F5 keys :)
- Write some locust test to reproduce the issue.
- Very high load on ES cluster. What's that all about?

Multiple queries per search

- WTF?

Multiple queries per search

- WTF?
- ipdb to the rescue

Multiple queries per search

- WTF?
- ipdb to the rescue
- Turns out when you call count or get_facet_count on a SearchQuerySet that has not yet been evaluated, it does a search. Doesn't cache any of the results.

Page rendering is slow

- By this time we've got newrelic.
- We can see at a glance we're spending an awful lot of time in memcache operations and the DB.
- Dajngo debug toolbar quickly shows we do over 300 DB requests per page load.
- This makes the baby Jesus cry.
- No one query is very slow. Death by 1 thousand papercuts.
- Latency is a bitch.

Page rendering is slow, continued

- Large amount of data not used very often, displayed only after 1 click to expand. Just load this later via ajax if anyone really wants it.
- select_related / prefetch_related.
- Cache rendered html, not just ORM access.

Reports

- Did I mention we have a really crappy data model?
- Lots of common report queries are to slow on our schema, so we have denormalized tables we update once per night.
- Starts to take so long night turns in to day and you have a grumpy content team.
- All the more annoying because it's holding locks on the primary data tables.

Reports, continued

- Break it up in to several smaller transactions.
- Set transaction isolation level to READ UNCOMMITTED.
- Then, while no longer looking at performance you see a join on a string field, deep down in some stored procedure. You switch it to join on indexed int fields you already had available, and the whole report generation now takes 30 minutes.

Some advice

- Have some form of log aggregation and analysis.
- newrelic is awesome (and expensive).
- Do load tests from time to time. (locust is really nice).
- Look at your data access patterns. Most sites have lots of mostly-read content. Ideal for caching.
- CACHE ALL THE THINGS! (jhonny-cache for ORM caching).
- Look at your web server config. How large is your worker pool?