foursquare®

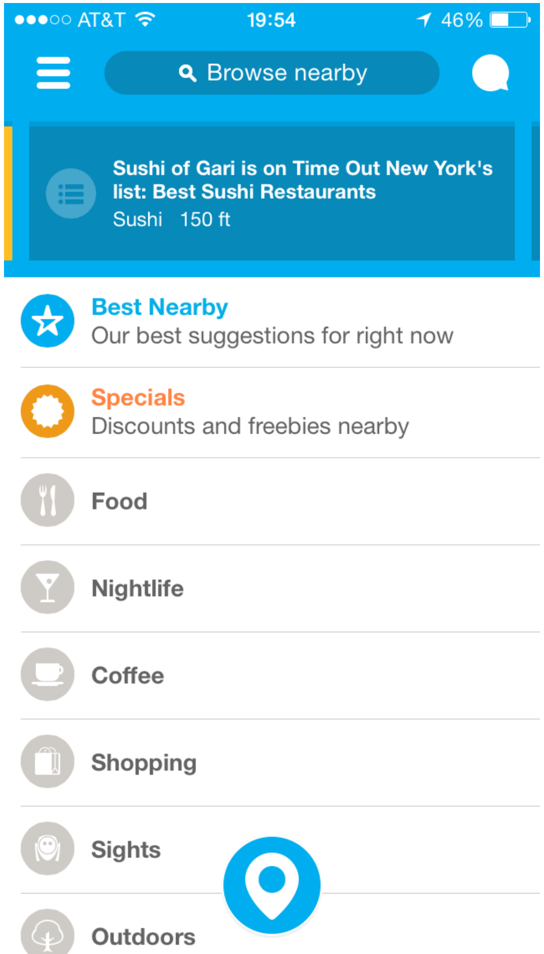
From check-ins to recommendations

Jon Hoffman @hoffrocket

QCon NYC – June 11, 2014

Swarm

About Foursquare



●●●○○ AT&T 19:54 46%

Browse nearby

Sushi of Gari is on Time Out New York's list: Best Sushi Restaurants
Sushi 150 ft

Best Nearby
Our best suggestions for right now

Specials
Discounts and freebies nearby

Food

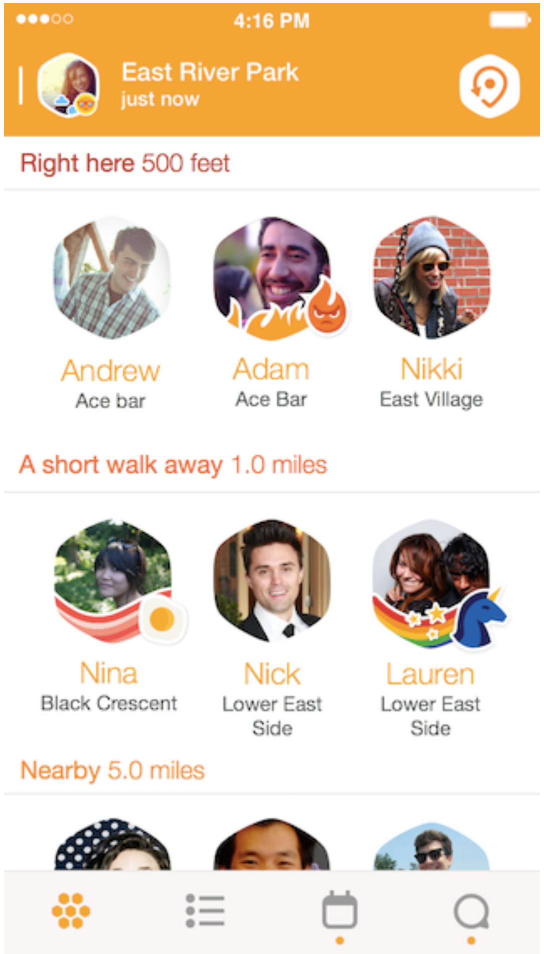
Nightlife

Coffee

Shopping

Sights

Outdoors



●●●○○ 4:16 PM

East River Park
just now

Right here 500 feet

Andrew Ace bar	Adam Ace Bar	Nikki East Village

A short walk away 1.0 miles

Nina Black Crescent	Nick Lower East Side	Lauren Lower East Side

Nearby 5.0 miles

Scaling in two parts

- Part one: data storage

- Part two: application complexity

Part 1: Data Storage

2009

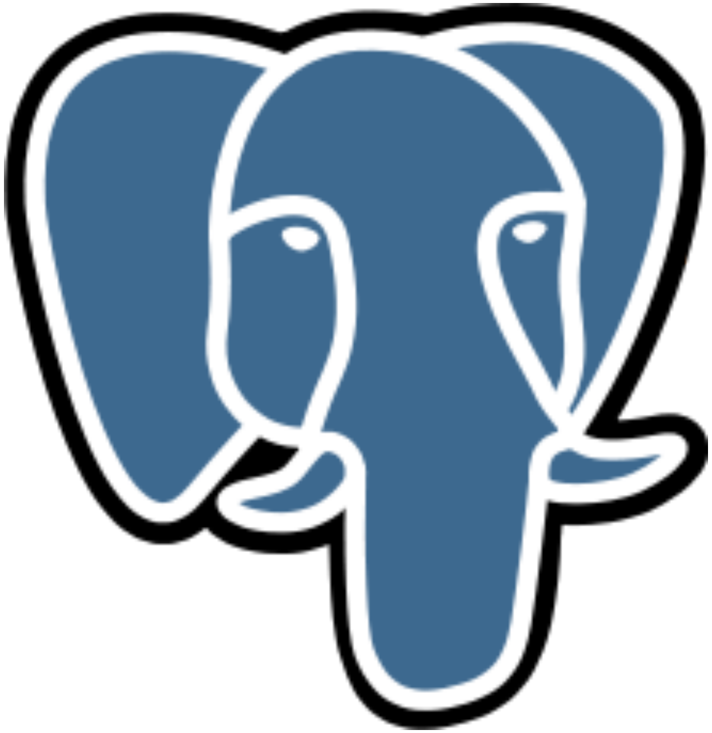
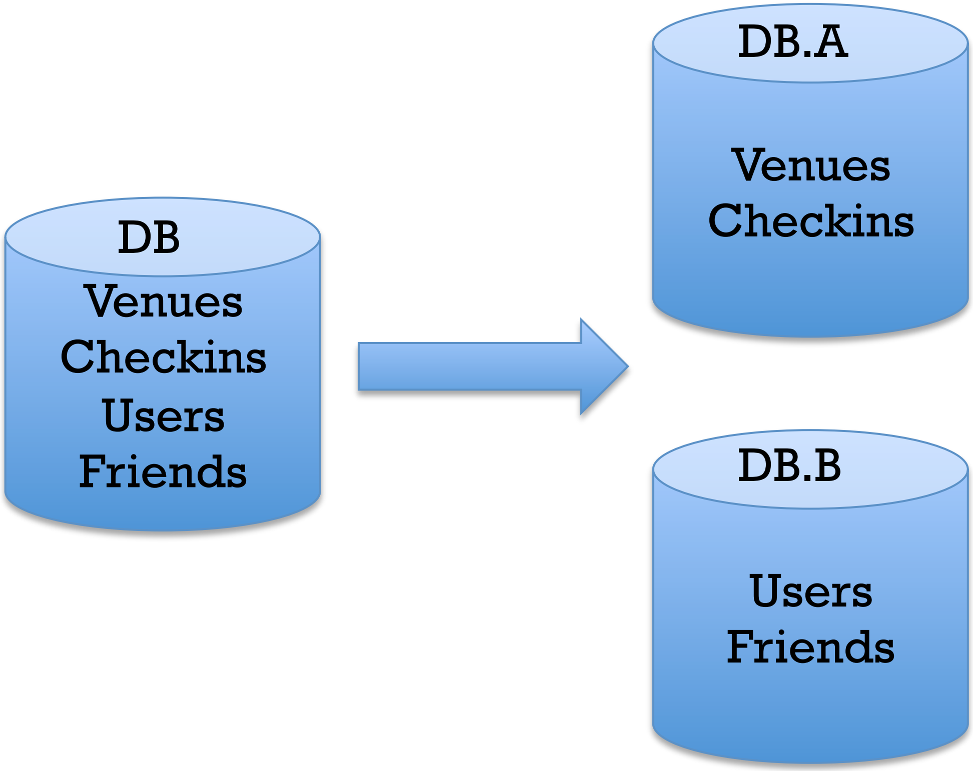
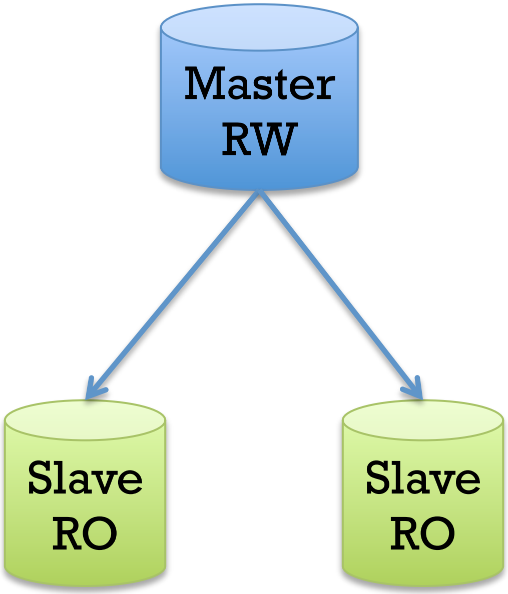


Table splits



Replication



Outgrowing our hardware

- Not enough RAM for indexes and working data set

- 100 writes/second/disk

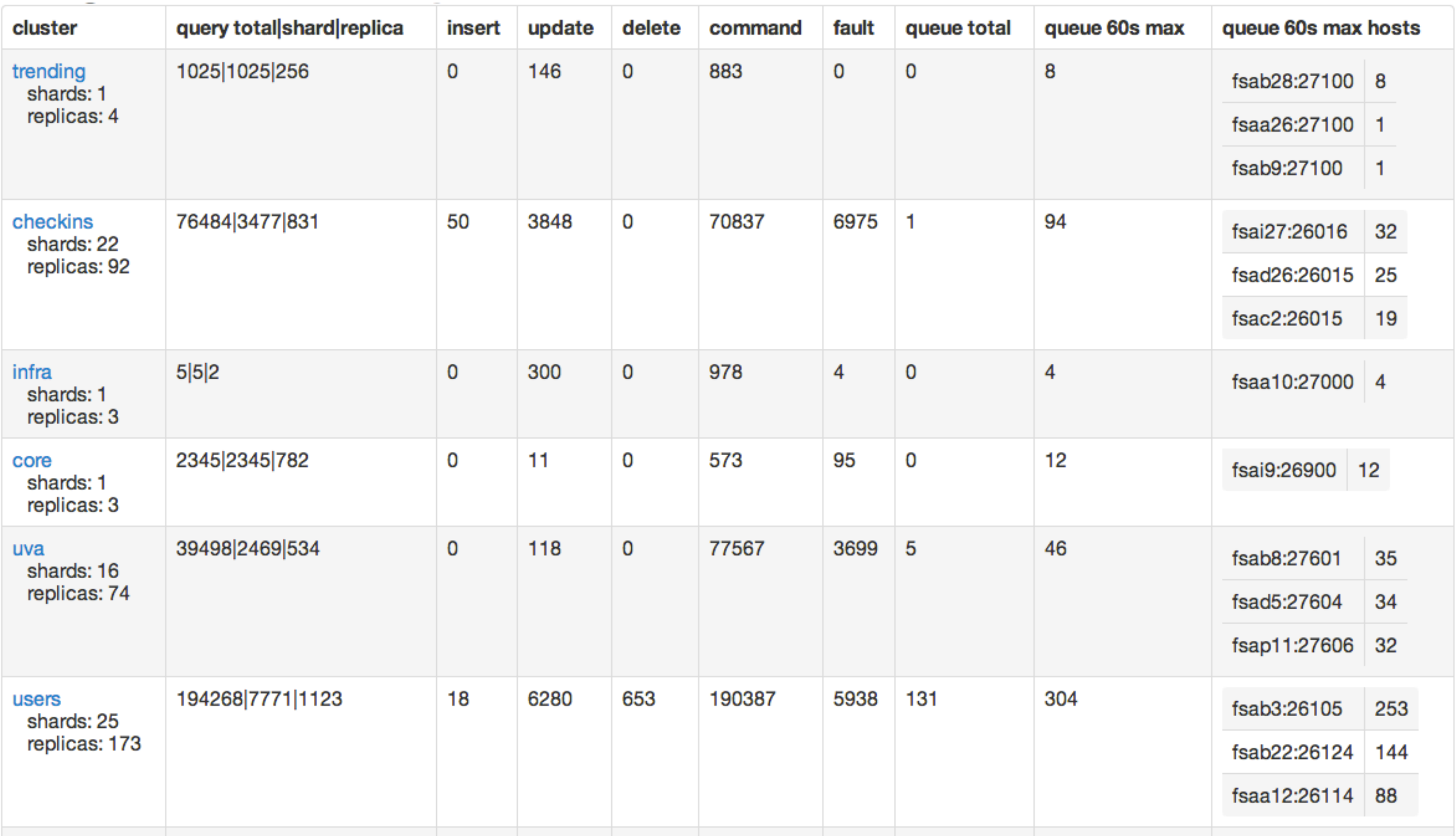
Sharding

- Manage ourselves in application code on top of postgres?

- Use something called Cassandra?

- Use something called HBase?

- Use something called Mongo?

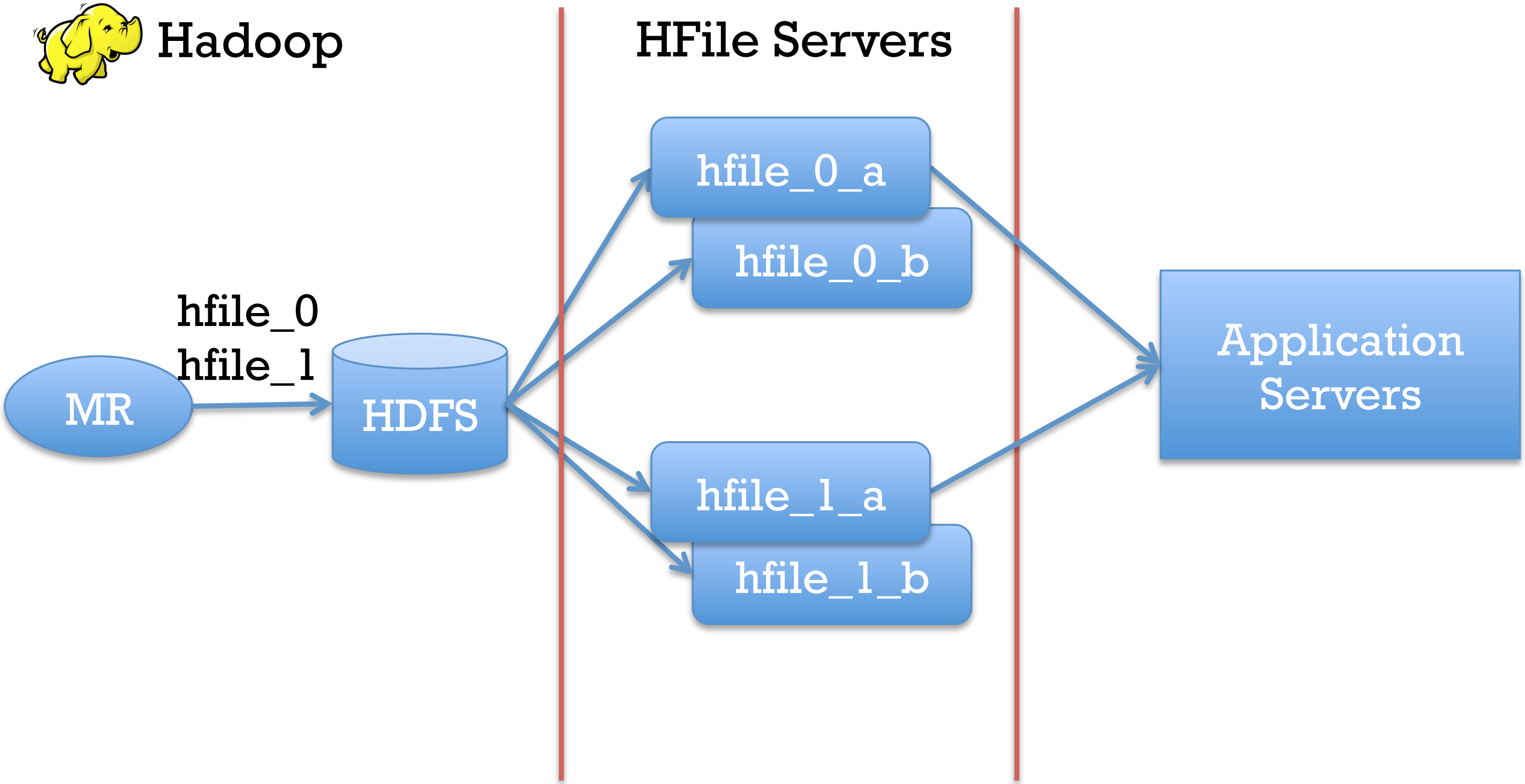


cluster	query total|shard|replica	insert	update	delete	command	fault	queue total	queue 60s max	queue 60s max hosts
trending shards: 1 replicas: 4	1025|1025|256	0	146	0	883	0	0	8	fsab28:27100 8 fsaa26:27100 1 fsab9:27100 1
checkins shards: 22 replicas: 92	76484|3477|831	50	3848	0	70837	6975	1	94	fsai27:26016 32 fsad26:26015 25 fsac2:26015 19
infra shards: 1 replicas: 3	5|5|2	0	300	0	978	4	0	4	fsaa10:27000 4
core shards: 1 replicas: 3	2345|2345|782	0	11	0	573	95	0	12	fsai9:26900 12
uva shards: 16 replicas: 74	39498|2469|534	0	118	0	77567	3699	5	46	fsab8:27601 35 fsad5:27604 34 fsap11:27606 32
users shards: 25 replicas: 173	194268|7771|1123	18	6280	653	190387	5938	131	304	fsab3:26105 253 fsab22:26124 144 fsaa12:26114 88

Besides Mongo

- Memcache
- Elastic search
  - nearby venue search
  - user search
- Custom data services
  - Read only key value server
  - in memory cache with business logic

HFile Service: Read only KV Store



Hadoop

HFile Servers

hfile_0

hfile_1

MR

HDFS

hfile_0_a

hfile_0_b

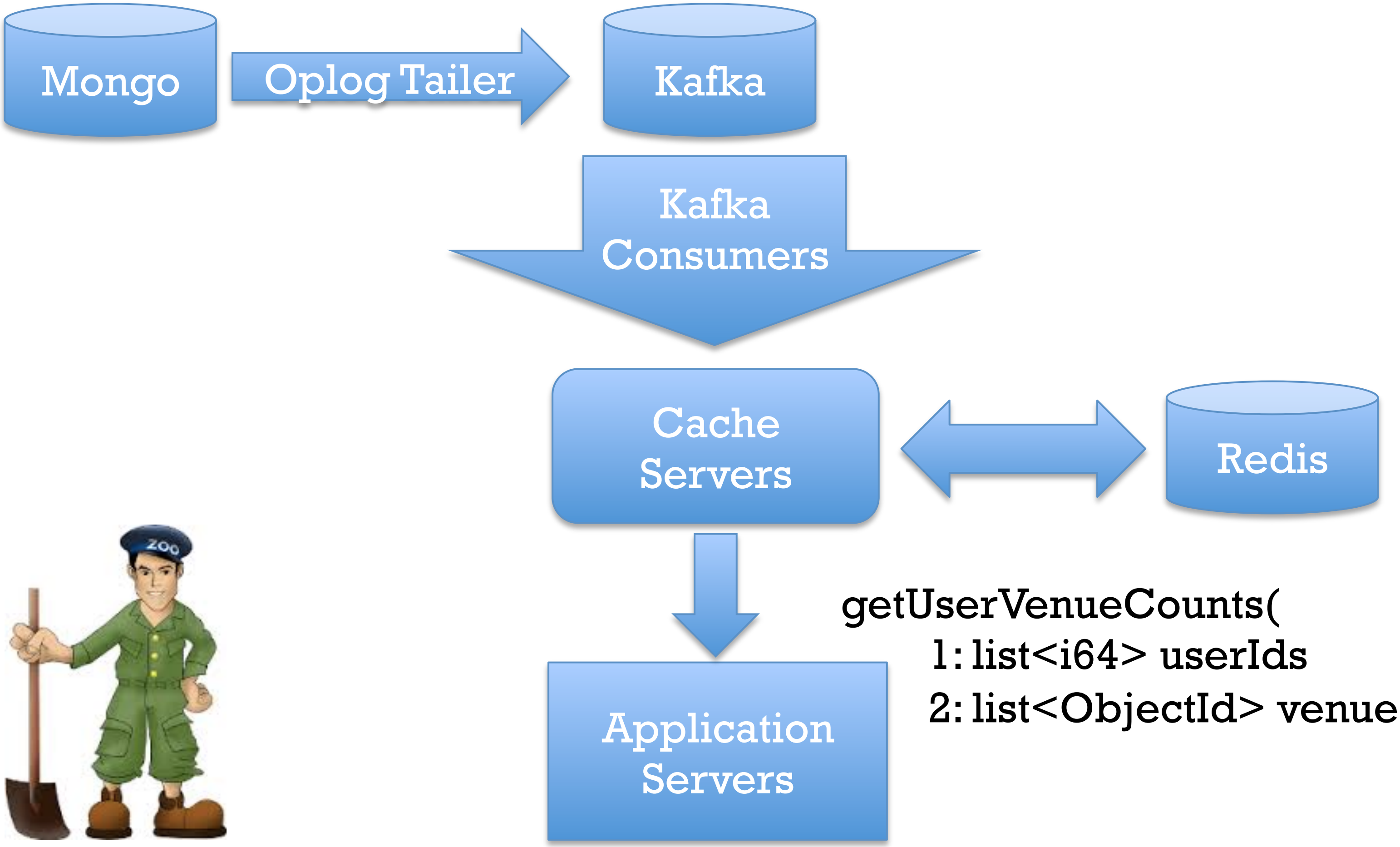
hfile_1_a

hfile_1_b

Application Servers

Zookeeper:
- data type to machine mapping
- key hash to shard mapping

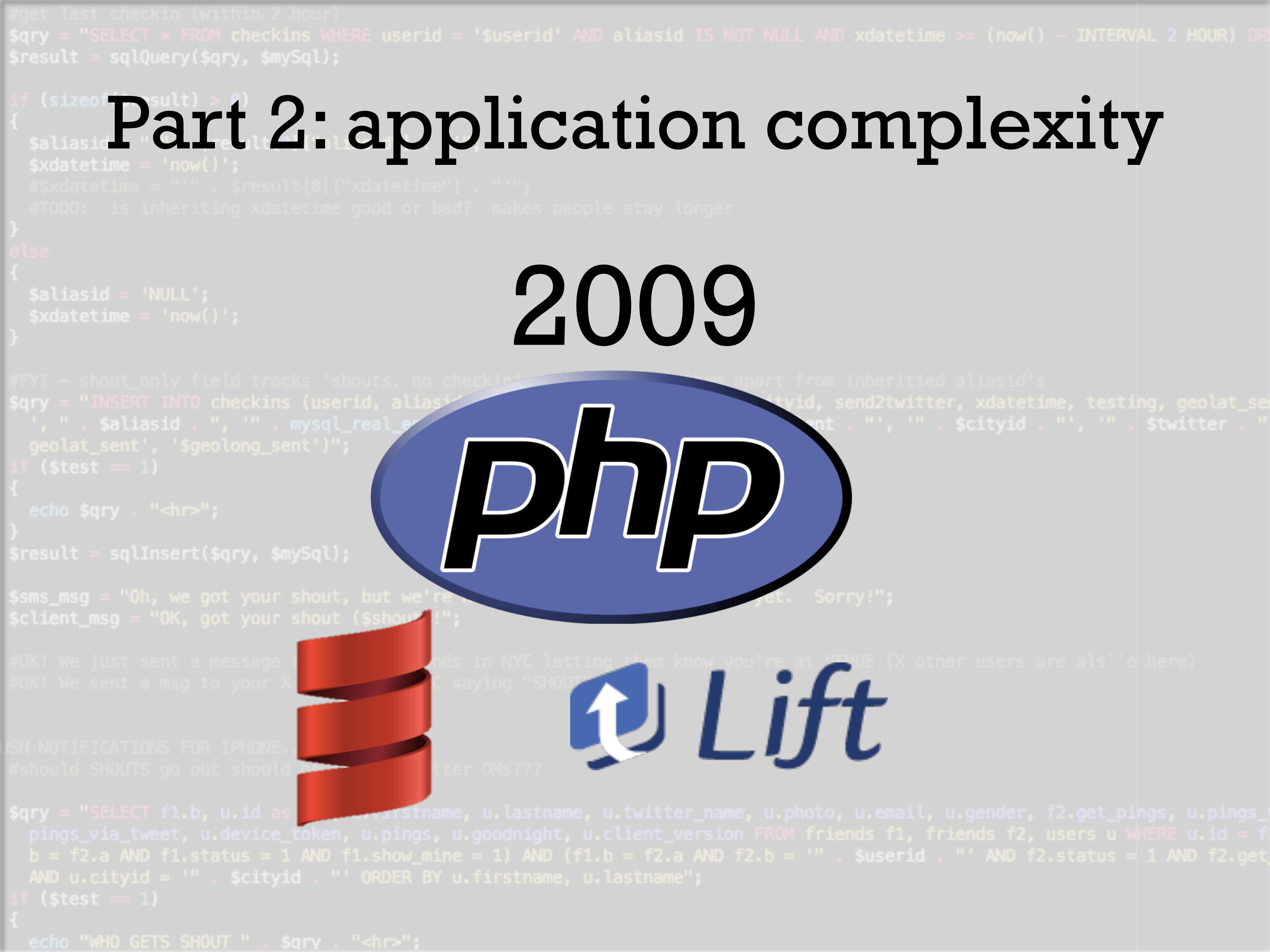
Caching Services



```
getUserVenueCounts(
        1: list<i64> userIds
        2: list<ObjectId> venues)
```

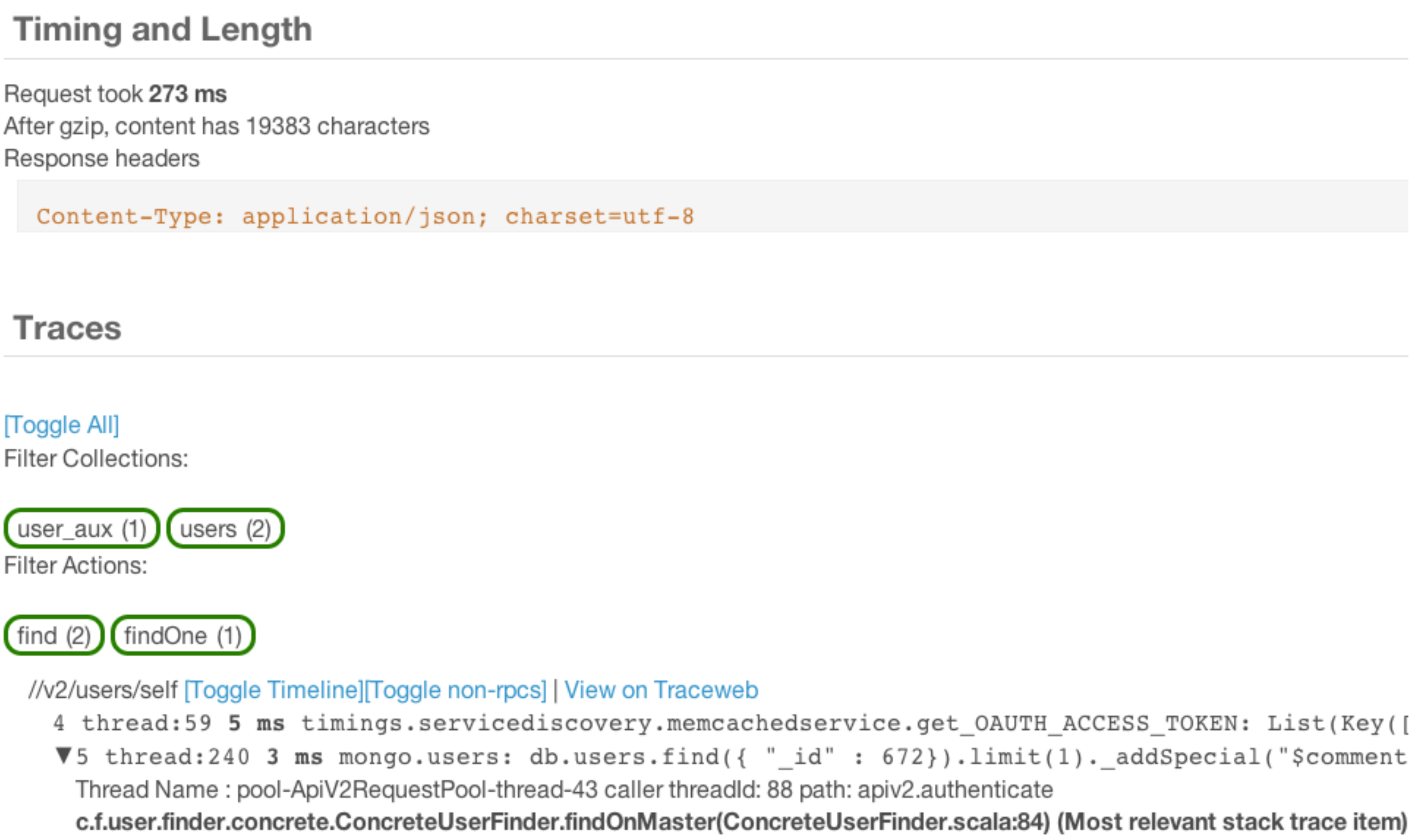
Part 2: application complexity

2009



```
#get last checkin (within 2 hour)
$qry = "SELECT * FROM checkins WHERE userid = '$userid' AND aliasid IS NOT NULL AND xdatetime >= (now() - INTERVAL 2 HOUR) OR
$result = sqlQuery($qry, $mySql);

if (sizeof(  sult) > 0)
{
  $aliasid
  $xdatetime = 'now()';
  #$xdatetime = "'" . $result[0]["xdatetime"] . "'";
  #TODO:  is inheriting xdatetime good or bad?  makes people stay longer
}
else
{
  $aliasid = 'NULL';
  $xdatetime = 'now()';
}

#FYI - shout_only field tracks 'shouts, no checkin                apart from inheritied aliasid's
$qry = "INSERT INTO checkins (userid, alias                        tyid, send2twitter, xdatetime, testing, geolat_se
  ', " . $aliasid . ", '" . mysql_real_e                           nt . "', '" . $cityid . "', '" . $twitter . "
  geolat_sent', '$geolong_sent')";
if ($test == 1)
{
  echo $qry . "<hr>";
}
$result = sqlInsert($qry, $mySql);

$sms_msg = "Oh, we got your shout, but we're                        yet.  Sorry!";
$client_msg = "OK, got your shout ($shou  !";

#OK! We just sent a message         nds in NYC letting them know you're at VENUE (X other users are als``o here)
#OK! We sent a msg to your X         C saying "SHOU

USH NOTIFICATIONS FOR IPHONE..
#should SHOUTS go out should         tter DMs???

$qry = "SELECT f1.b, u.id as       irstname, u.lastname, u.twitter_name, u.photo, u.email, u.gender, f2.get_pings, u.pings_
  pings_via_tweet, u.device_token, u.pings, u.goodnight, u.client_version FROM friends f1, friends f2, users u WHERE u.id = f
  b = f2.a AND f1.status = 1 AND f1.show_mine = 1) AND (f1.b = f2.a AND f2.b = '" . $userid . "' AND f2.status = 1 AND f2.get
  AND u.cityid = '" . $cityid . "' ORDER BY u.firstname, u.lastname";
if ($test == 1)
{
  echo "WHO GETS SHOUT " . $qry . "<hr>";
```

RPC Tracing



Timing and Length

Request took **273 ms**
After gzip, content has 19383 characters
Response headers

```
Content-Type: application/json; charset=utf-8
```

Traces

[Toggle All]
Filter Collections:

user_aux (1) users (2)
Filter Actions:

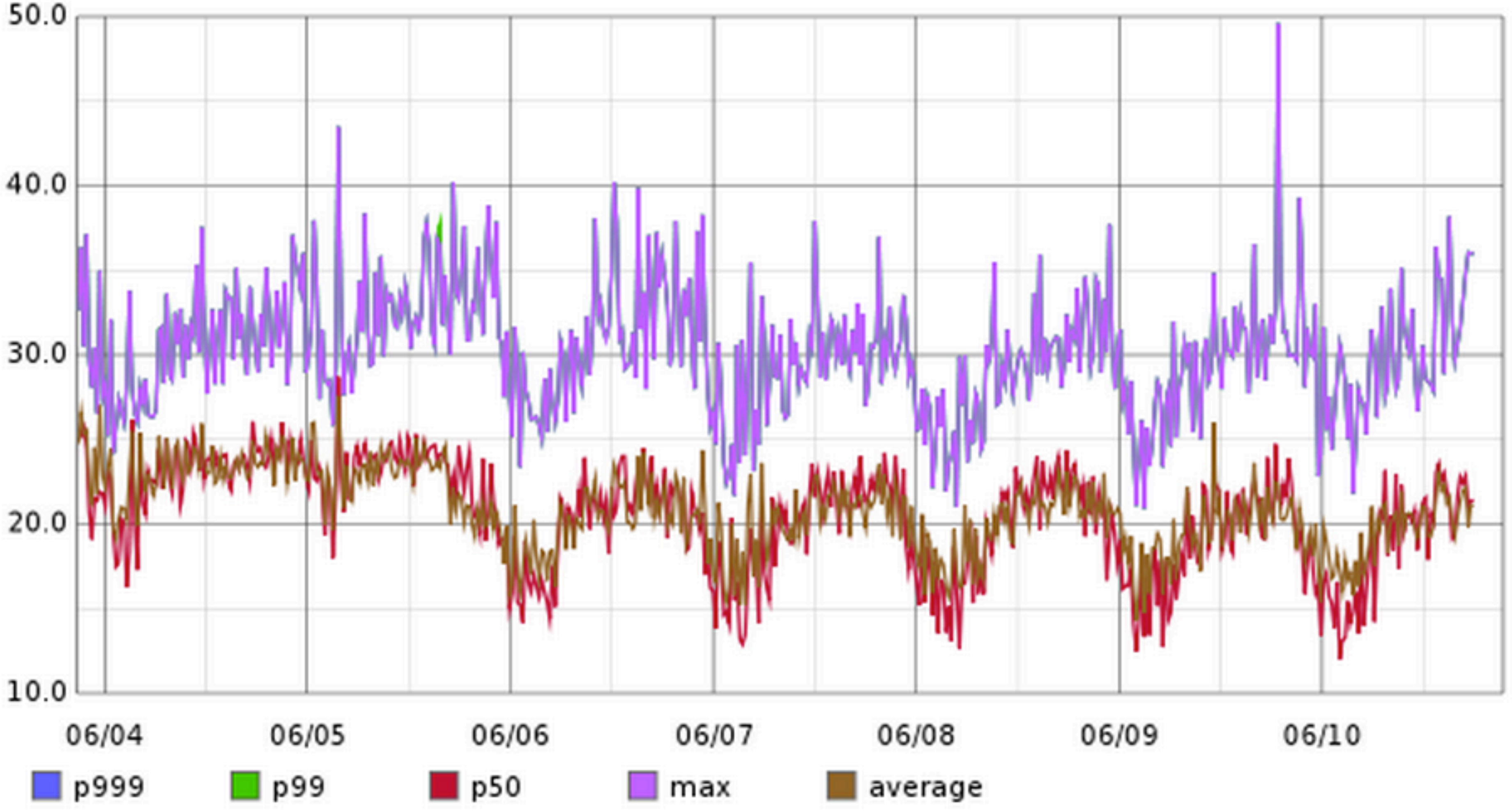
find (2) findOne (1)

//v2/users/self [Toggle Timeline][Toggle non-rpcs] | View on Traceweb

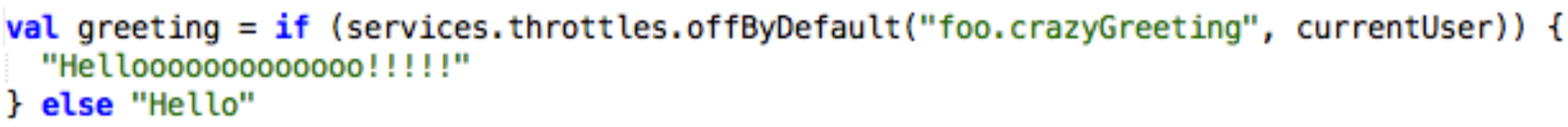
```
 4 thread:59 5 ms timings.servicediscovery.memcachedservice.get_OAUTH_ACCESS_TOKEN: List(Key([
▼5 thread:240 3 ms mongo.users: db.users.find({ "_id" : 672}).limit(1)._addSpecial("$comment
```

Thread Name : pool-ApiV2RequestPool-thread-43 caller threadId: 88 path: apiv2.authenticate
c.f.user.finder.concrete.ConcreteUserFinder.findOnMaster(ConcreteUserFinder.scala:84) (Most relevant stack trace item)

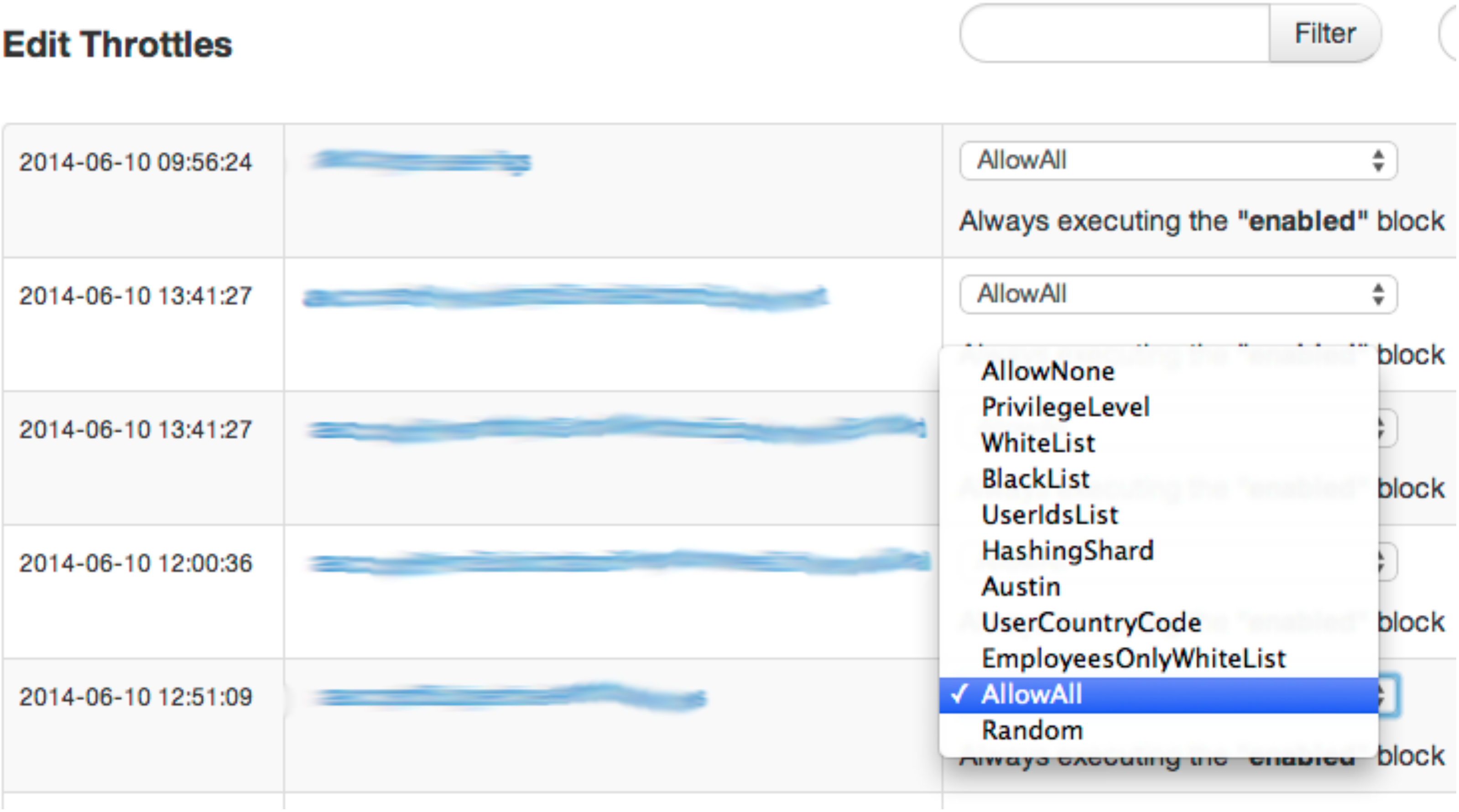
RPC Counts Past Week



Throttles



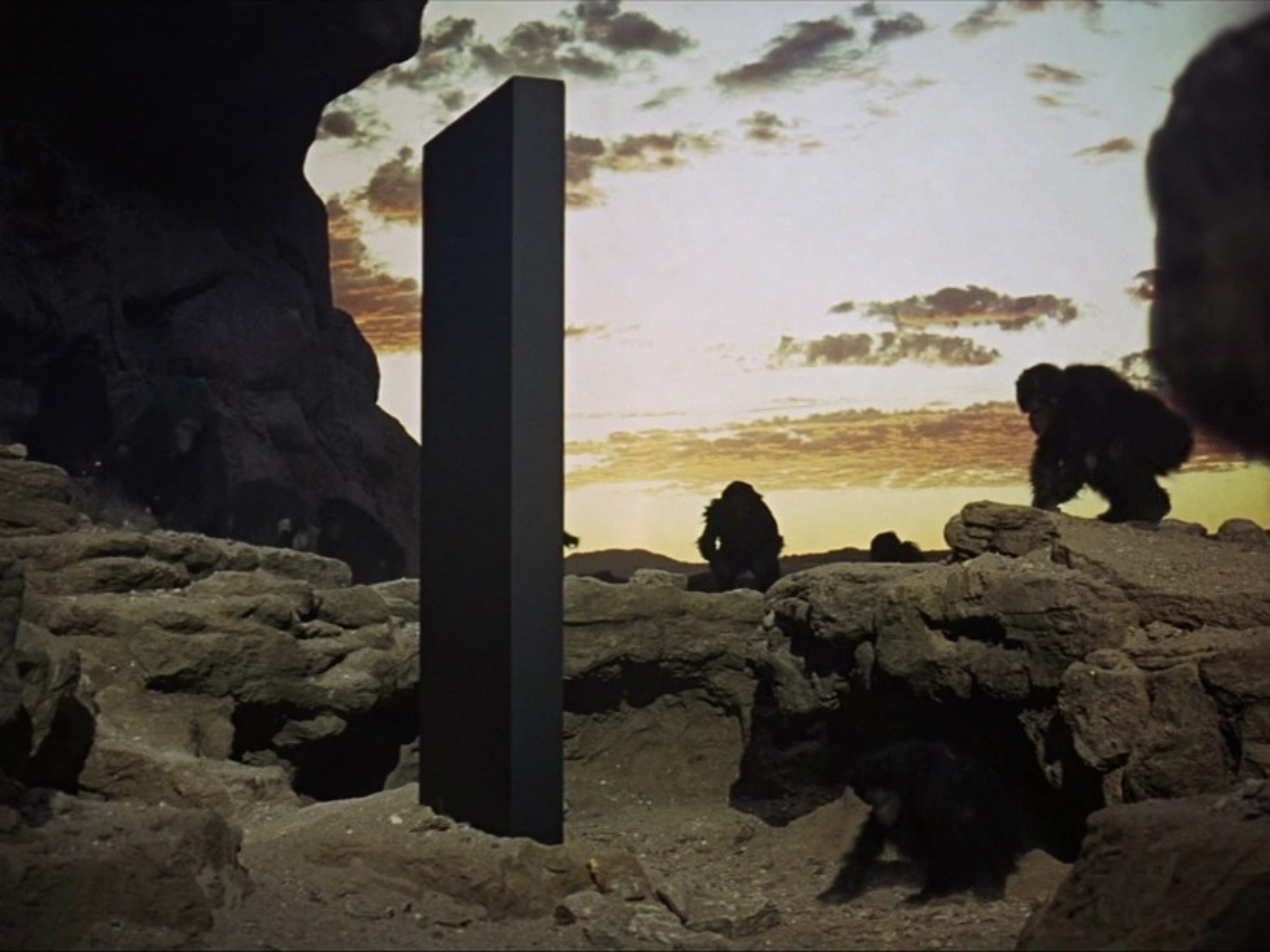
```
val greeting = if (services.throttles.offByDefault("foo.crazyGreeting", currentUser)) {
  "Hellooooooooooooo!!!!!"
} else "Hello"
```



Edit Throttles

2014-06-10 09:56:24	[redacted]	AllowAll Always executing the "**enabled**" block
2014-06-10 13:41:27	[redacted]	AllowAll block
2014-06-10 13:41:27	[redacted]	block
2014-06-10 12:00:36	[redacted]	block
2014-06-10 12:51:09	[redacted]	block

Remember the goats?



Monolithic problems

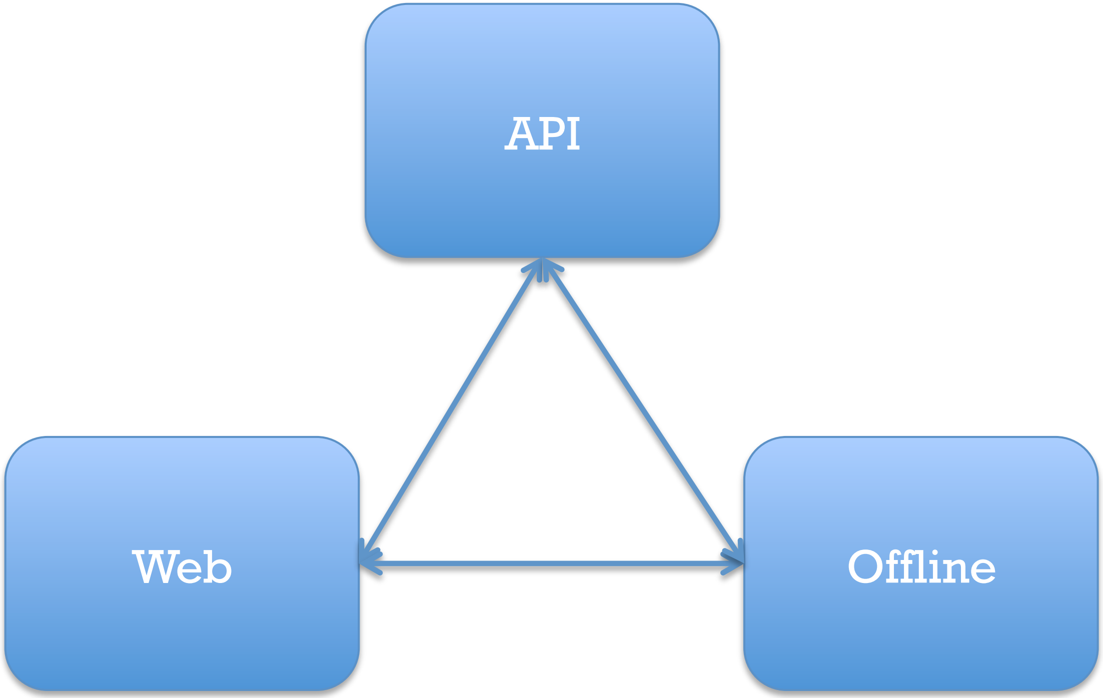
- Compiling all the code, all the time

- Deploying all the code all the time

- Hard to isolate cause of performance regressions and resource leaks

SOA Infancy

- Single codebase, Multiple builds



Finagle Era

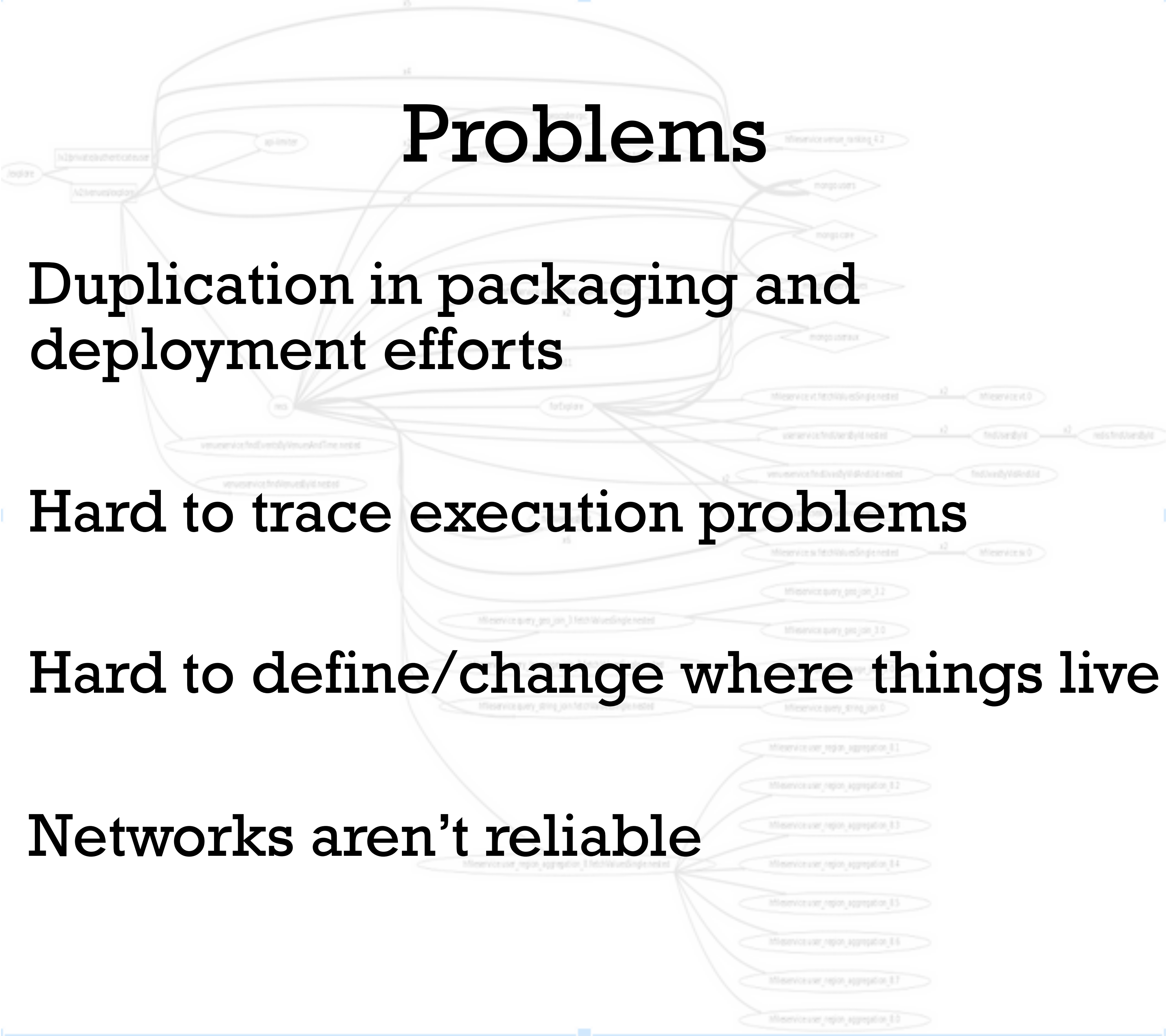
- Twitter’s scala based RPC library

```
service Geocoder {
   GeocodeResponse geocode(
      1: GeocodeRequest r
   )
}
```

Benefits

- Independent compile targets
- Fined grained control on releases and bug fixes
- Functional isolation

Problems



- Duplication in packaging and deployment efforts
- Hard to trace execution problems
- Hard to define/change where things live
- Networks aren’t reliable

Builds and deploys

- single service definition file

- consistent build packaging

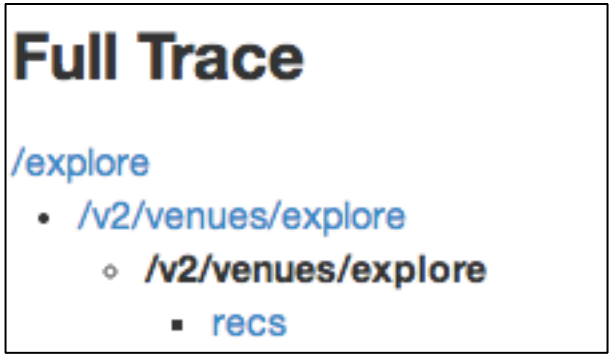
- simple deployment of canary & fleet

  ```
  ./service_releaser –j service_name
  ```

Monitoring

- healthcheck endpoint over http
- consistent metric names
- dashboard for every service

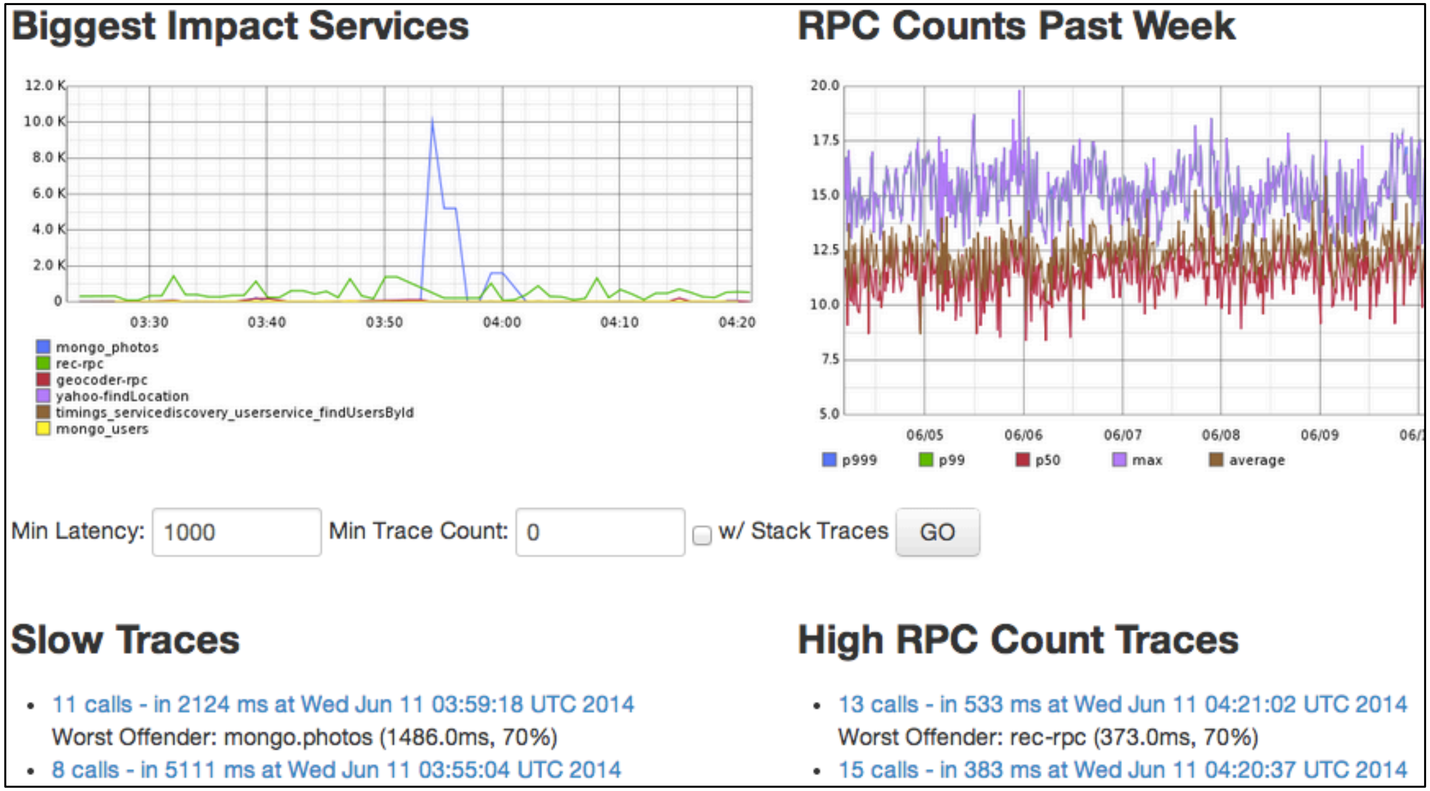
Distributed Tracing



Full Trace

/explore

- /v2/venues/explore
    - **/v2/venues/explore**
        - recs



Biggest Impact Services

RPC Counts Past Week

Min Latency: 1000 Min Trace Count: 0 □ w/ Stack Traces

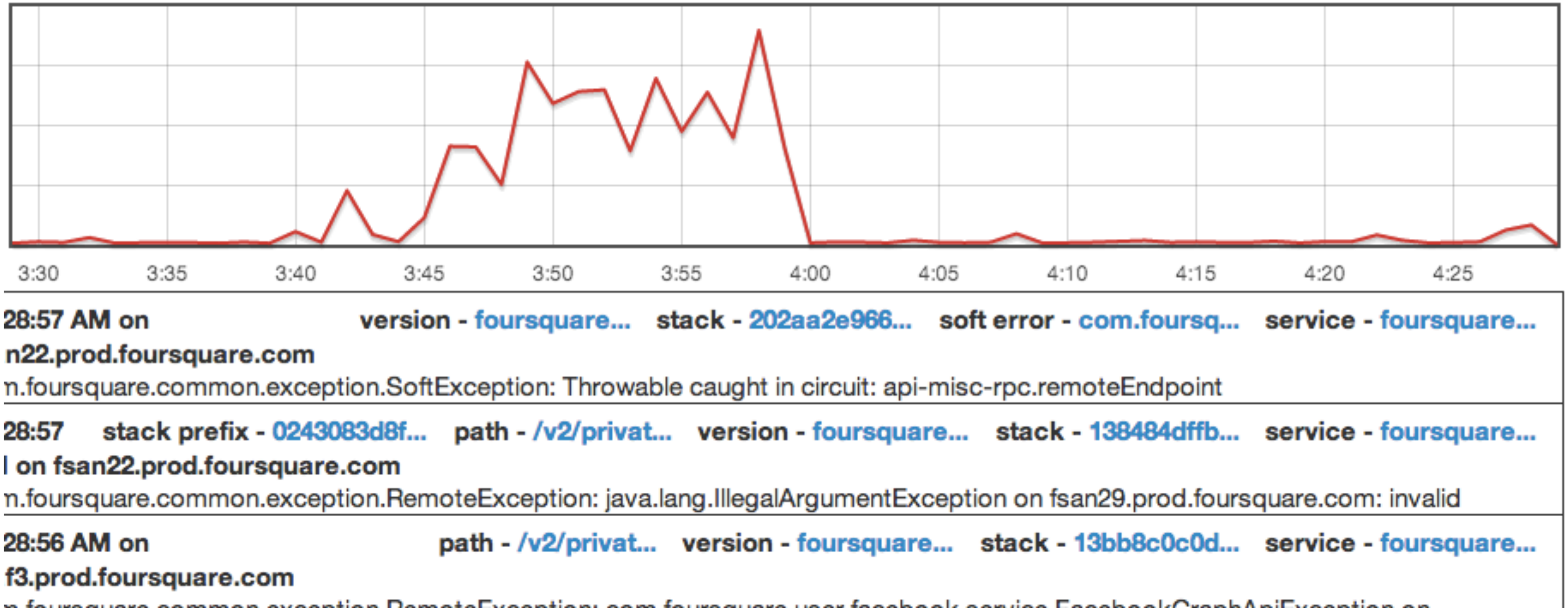
Slow Traces

- 11 calls - in 2124 ms at Wed Jun 11 03:59:18 UTC 2014
  Worst Offender: mongo.photos (1486.0ms, 70%)
- 8 calls - in 5111 ms at Wed Jun 11 03:55:04 UTC 2014

High RPC Count Traces

- 13 calls - in 533 ms at Wed Jun 11 04:21:02 UTC 2014
  Worst Offender: rec-rpc (373.0ms, 70%)
- 15 calls - in 383 ms at Wed Jun 11 04:20:37 UTC 2014

Exception Aggregation



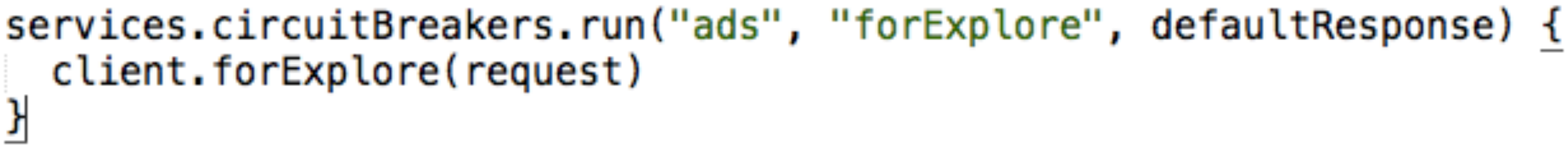
```
28:57 AM on                   version - foursquare...   stack - 202aa2e966...   soft error - com.foursq...   service - foursquare...
n22.prod.foursquare.com
n.foursquare.common.exception.SoftException: Throwable caught in circuit: api-misc-rpc.remoteEndpoint
28:57    stack prefix - 0243083d8f...   path - /v2/privat...   version - foursquare...   stack - 138484dffb...   service - foursquare...
l on fsan22.prod.foursquare.com
n.foursquare.common.exception.RemoteException: java.lang.IllegalArgumentException on fsan29.prod.foursquare.com: invalid
28:56 AM on                        path - /v2/privat...   version - foursquare...   stack - 13bb8c0c0d...   service - foursquare...
f3.prod.foursquare.com
n.foursquare.common.exception.RemoteException: com.foursquare.user.facebook.service.FacebookGraphApiException on
```

Application Discovery

- Finagle Server Sets + ZK

Circuit Breaking

- Fast failing RPC calls after some error rate threshold



```
services.circuitBreakers.run("ads", "forExplore", defaultResponse) {
  client.forExplore(request)
}
```

- Loosely based on Netflix's hystrix

SOA Problem Recap

- Duplication in packaging and deployment efforts
  - Build and deploy automation
- Hard to trace execution problems
  - Monitoring consistency
  - Distributed Tracing
  - Error aggregation
- Hard to define/change where things live
  - Application discovery with zookeeper
- Networks aren't reliable
  - Circuit breaking

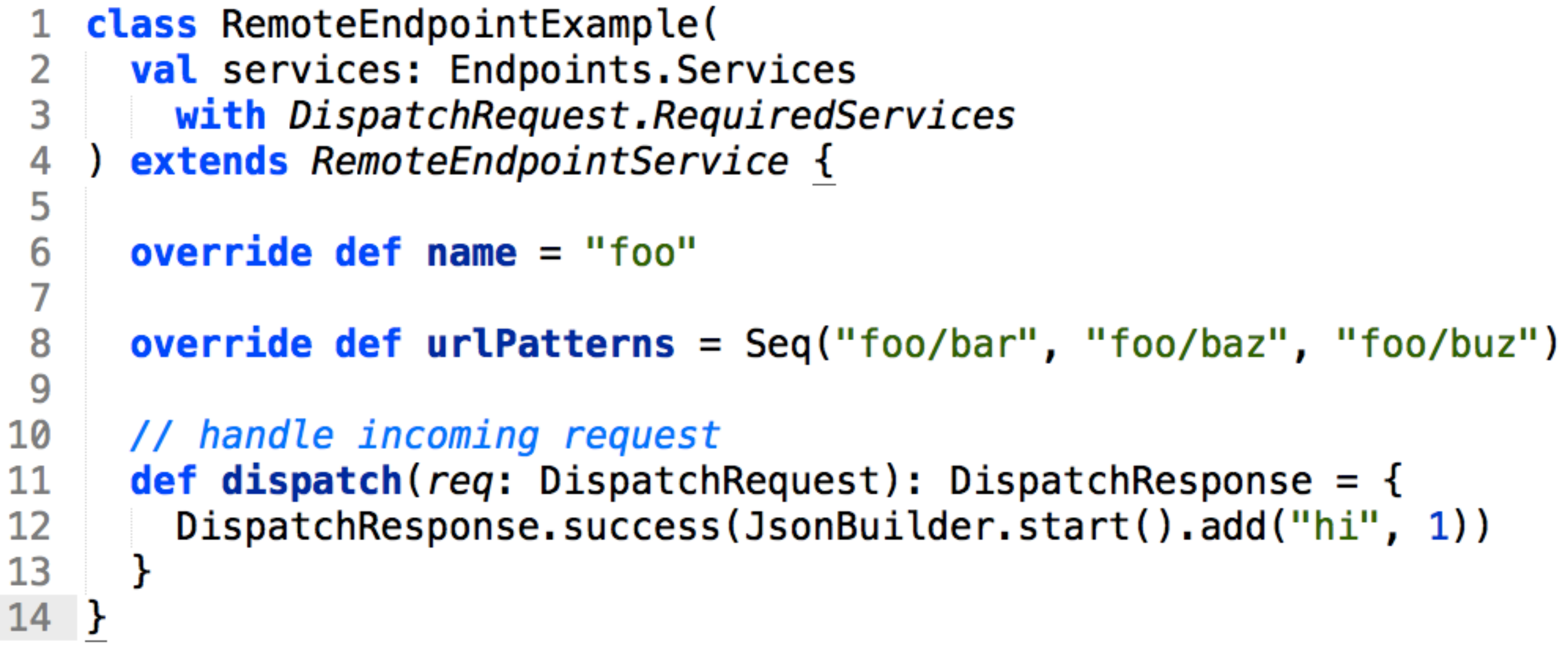
Organization

- Smaller teams owning front to back implementation of features
- Desire to have quick deploy cycles on new API endpoints

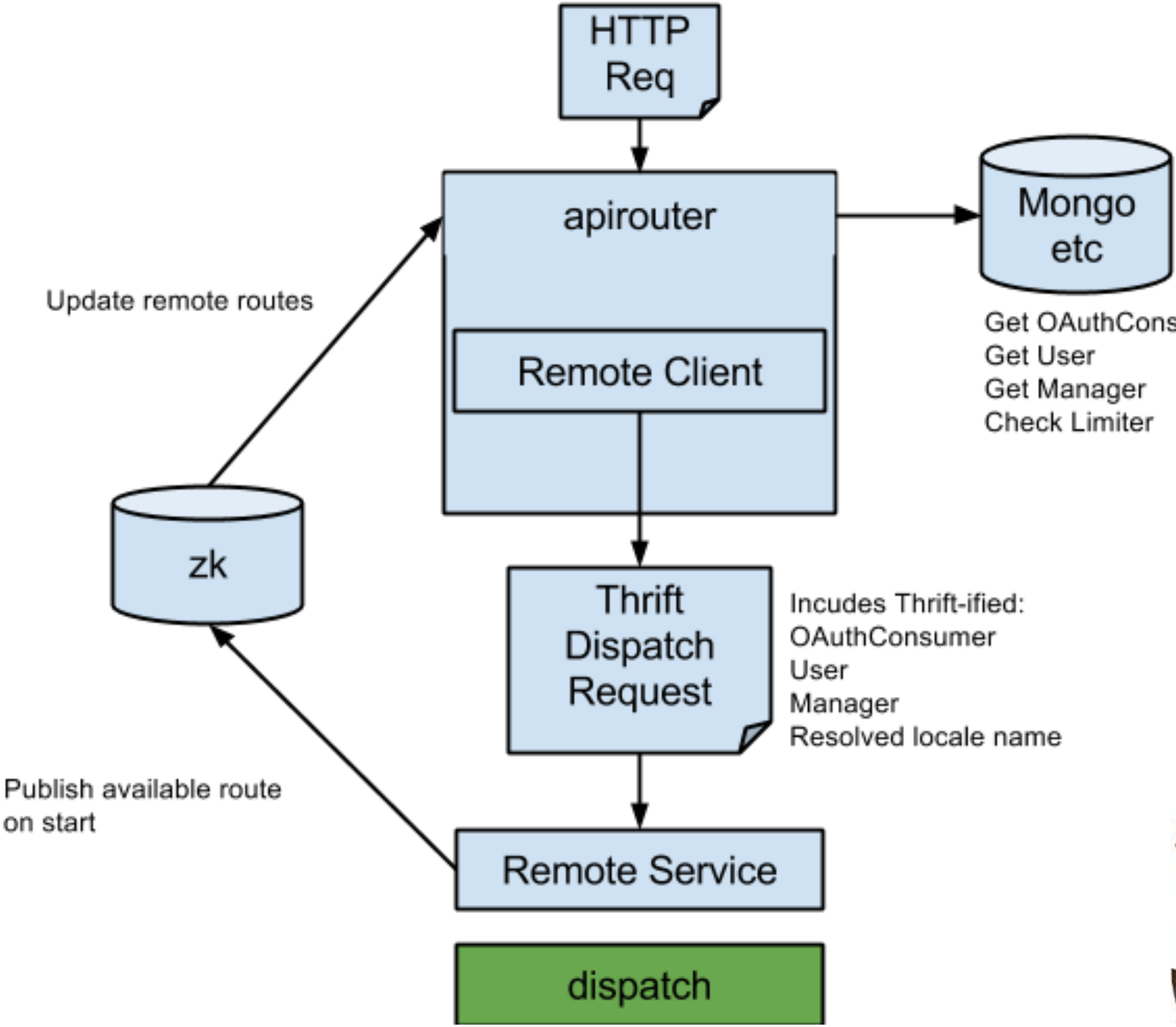
Swarm

Remote Endpoints

Wouldn't it be cool if a developer could expose a new API endpoint without redeploying our still monolithic API server?



```scala
class RemoteEndpointExample(
  val services: Endpoints.Services
    with DispatchRequest.RequiredServices
) extends RemoteEndpointService {

  override def name = "foo"

  override def urlPatterns = Seq("foo/bar", "foo/baz", "foo/buz")

  // handle incoming request
  def dispatch(req: DispatchRequest): DispatchResponse = {
    DispatchResponse.success(JsonBuilder.start().add("hi", 1))
  }
}
```



HTTP Req

apirouter

Remote Client

Mongo etc

Get OAuthConsumer
Get User
Get Manager
Check Limiter

Update remote routes

zk

Thrift Dispatch Request

Incudes Thrift-ified:
OAuthConsumer
User
Manager
Resolved locale name

Publish available route
on start

Remote Service

dispatch

Remote Endpoint Benefits

- Very easy to experiment with new endpoints
- Tight contract for service interaction
  - JSON responses
  - all http params passed along
- Clear path to breaking off more chunks from API monolith

Future work: Part 3?

- Further isolating services with independent storage layers?
- Completely automated continuous deployment
- Hybrid immutable/mutable data storage
  - mongo & hfile & cache service

Thanks!

- Want to build these things?

  https://foursquare.com/jobs

- jon@foursquare.com