
SOUNDCLOUD

Scaling RabbitMQ at SoundCloud

Sebastian Ohm (@sohm)

Erlang User Conference 2013

soundcloud

audio platform
~12 hours of audio/min.
reach 200 mil./month
(8% of internet)


Like
Share
Disclosure
Latch ft. Sam Smith
Buy on iTunes
4.02
2,651,584
44,496
5,451
1,944
SOUNDCLOUD


YOUR WEEKLY ADDRESS
AUDIO
The White House
2 days
Weekly Address: Time to Pass Commonsense Immigration Reform (Jun 07, 2013)
4.19
Like
Repost
Add to set
Share
Download
946
18
3
2


Record
Share Your Sounds
REC
Available on the App Store
Get it on Google play

transcoding

user generated content
audio processing
image generation

transcoding (2)

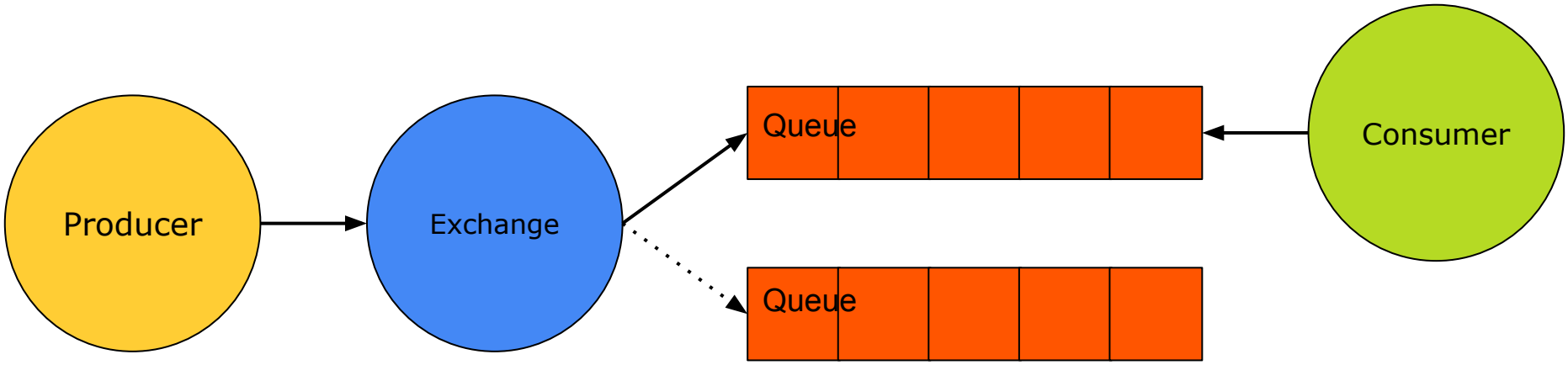
media stored in s3
worker pool on ec2
coordination?

AMQP

the model (0.9)

primitives

messages
exchanges
routing keys
queues


Producer
Exchange
Queue
Queue
Consumer

key benefit

producers
consumers
decoupled (scalable)

transcoding

rails to cloud and back

ruby, clojure, scala,
go, java, c, c++,
javascript, coffeescript
objective-c, python,
erlang, haskell

```
class Transcoding < ActiveRecord::Base
  def queue
    ex = declare_exchange('media')
    ex.publish('media.uploaded', {
      :uid => uid
    })
  end
end
```

```
class Transcoder
  def subscribe
    ex = declare_exchange('media')
    qu = declare_queue('media.uploaded')
    qu.bind(ex)
    qu.subscribe do |headers, message|
      process(message)
      headers.ack
    end
  end
end
```

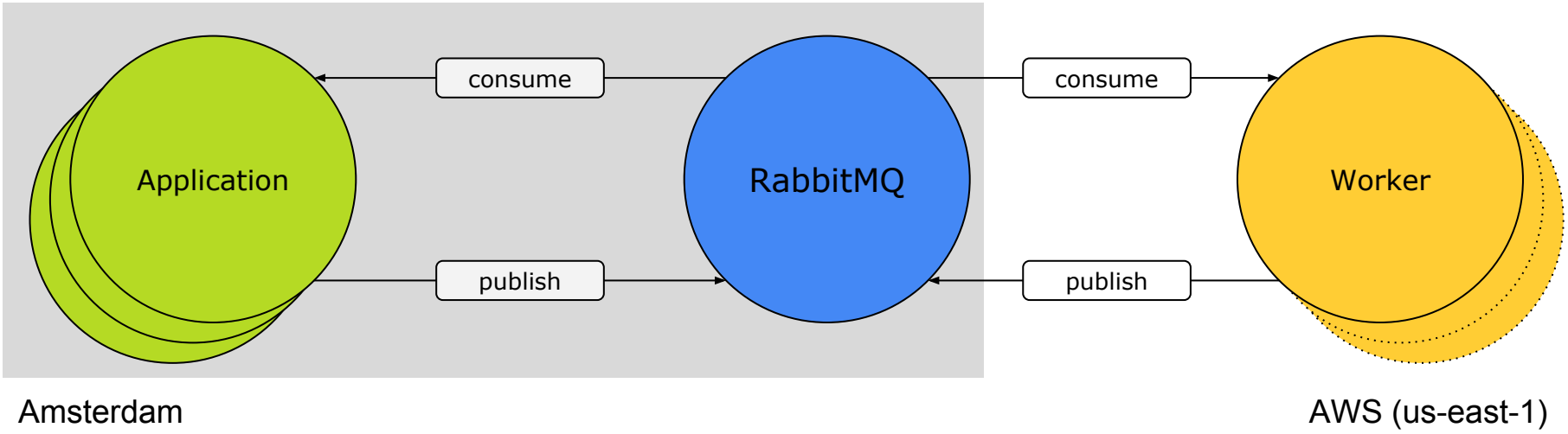
```
class Transcoder
  def process(message)
    uid = message[:uid]

    # do some work

    ex = declare_exchage('media')
    ex.publish('media.finished', {
      :uid =>  uid,
      :mp3 => 's3://sc-media/uid.mp3'
    })
  end
end
```

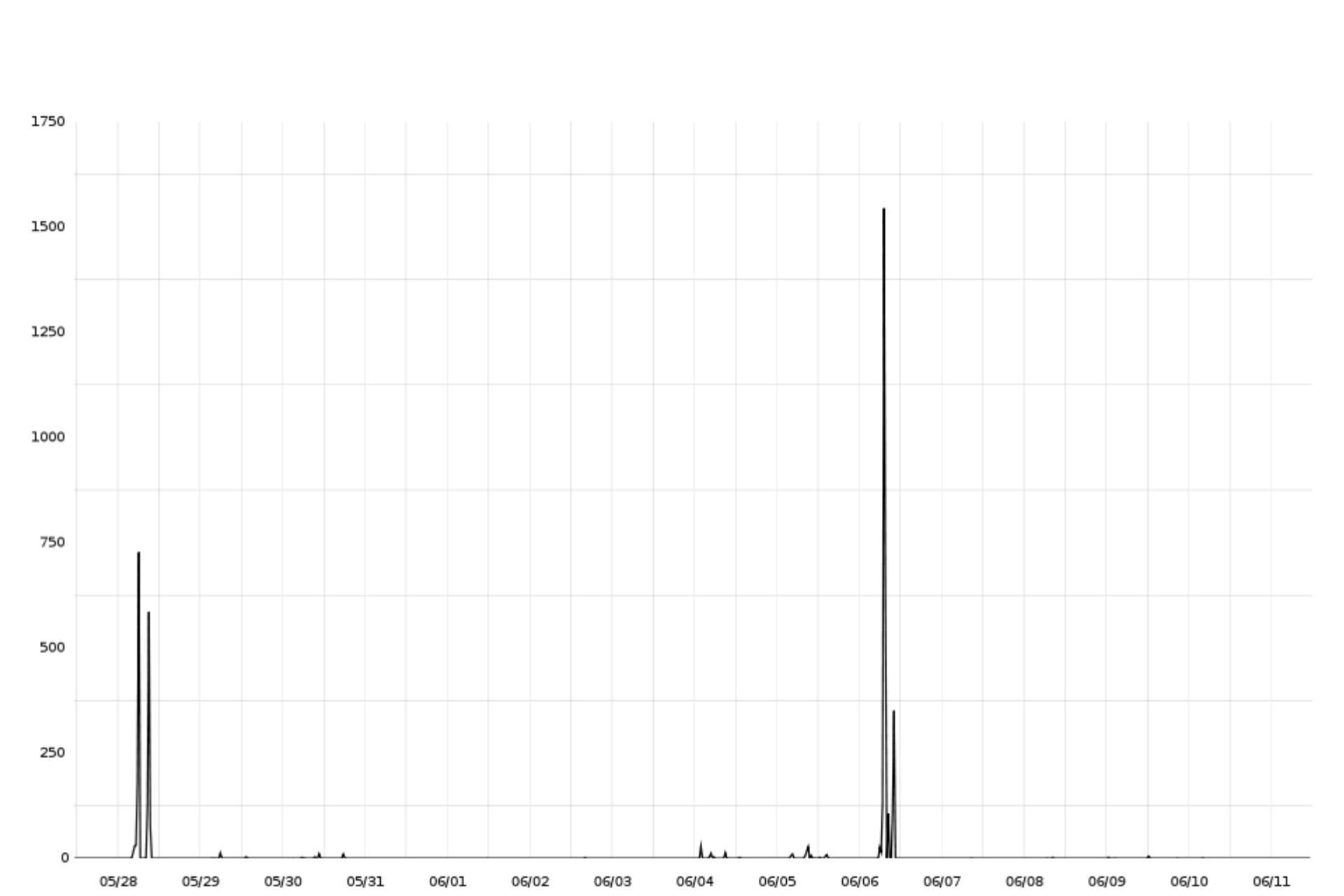
broker

rabbitmq - erlang
rock stable
amqp 0.9.1
(black box)


consume
consume
Application
RabbitMQ
Worker
publish
publish
Amsterdam
AWS (us-east-1)

transcoding, solved

autoscaling worker pool
scales quickly
handles spikes
solid broker impl.


1750
1500
1250
1000
750
500
250
0
05/28
05/29
05/30
05/31
06/01
06/02
06/03
06/04
06/05
06/06
06/07
06/08
06/09
06/10
06/11

this works really well!

let's use it for everything!!!

deferred processing

avoid runtime limitations
scale slow actions
quick HTTP responses

transcoding-like services

classification
tagging
content identification

environments

production.live.model.create
test.development.model.create

activities

activity feeds
materialized for every user
(stored in cassandra)

Stream | Explore | Search

TheEconomist
Protests spread in Turkey
17 hours
news
7.07
Like | Repost | Add to set | Share | Download
26,033 | 10 | 2

Py. Natalie
Polyethers
18 hours
electronic
3.37
Like | Repost | Add to set | Share | Buy from iTunes
17,454 | 760 | 181 | 43

Kleerup Natalie
Until We Bleed (with Lykke Li)
21 hours
4.24
Like | Repost | Add to set | Share | Buy
101,213 | 1,577 | 326 | 47

STRFKR (starfucker) Natalie
Beach Monster
22 hours
2.09
Like | Repost | Add to set | Share | Buy on iTunes
48,002 | 502 | 87 | 33

SUMMER HEART Natalie
Beat of Your Heart
23 hours
Summerwave
3.25
Like | Repost | Add to set | Share
3,045 | 256 | 72 | 21

activities (2)

1. observe changes in domain models
2. determine subscribers
3. write to storage

```ruby
module ModelBroadcast
  def self.included(base)
    base.after_create do |m|
      publish('model.create', m.attributes)
    end
  end
end

class Comment < ActiveRecord::Base
  include ModelBroadcast
end
```

```
Track.create({:user_id => Skrillex.id})
```

```
SELECT fan_id FROM followers WHERE user_id = 123;
```

```
ex = declare_exchange('activities.fanout')
fan_ids.each do |fan_id|
  ex.publish('activities.track', {
    :creator_id => Skrillex.id,
    :fan_id     => fan_id
  })
end
```

so. much. stuff.

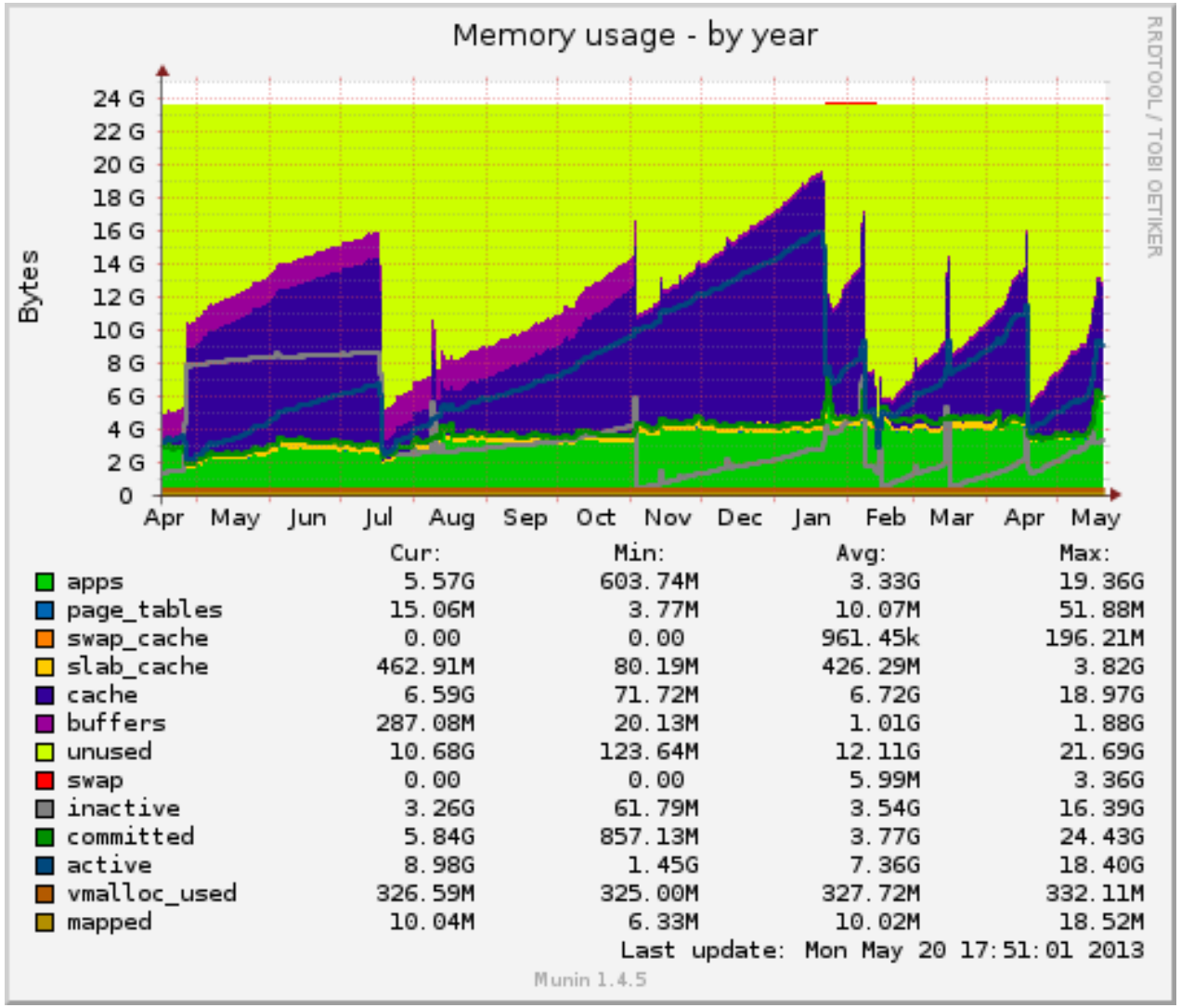
what could possibly go wrong...

the broker is down...

... we're down

frequent downtimes

single, shared broker
steadily increasing volume
diurnal cycle, bursts


Memory usage - by year
RRDTOOL / TOBI OETIKER
Bytes
24 G
22 G
20 G
18 G
16 G
14 G
12 G
10 G
8 G
6 G
4 G
2 G
0
Apr May Jun Jul Aug Sep Oct Nov Dec Jan Feb Mar Apr May
Cur: Min: Avg: Max:
apps 5.57G 603.74M 3.33G 19.36G
page_tables 15.06M 3.77M 10.07M 51.88M
swap_cache 0.00 0.00 961.45k 196.21M
slab_cache 462.91M 80.19M 426.29M 3.82G
cache 6.59G 71.72M 6.72G 18.97G
buffers 287.08M 20.13M 1.01G 1.88G
unused 10.68G 123.64M 12.11G 21.69G
swap 0.00 0.00 5.99M 3.36G
inactive 3.26G 61.79M 3.54G 16.39G
committed 5.84G 857.13M 3.77G 24.43G
active 8.98G 1.45G 7.36G 18.40G
vmalloc_used 326.59M 325.00M 327.72M 332.11M
mapped 10.04M 6.33M 10.02M 18.52M
Last update: Mon May 20 17:51:01 2013
Munin 1.4.5

partition workload

add another broker
use for activities only
use more queues
breathing room

the broker is down...

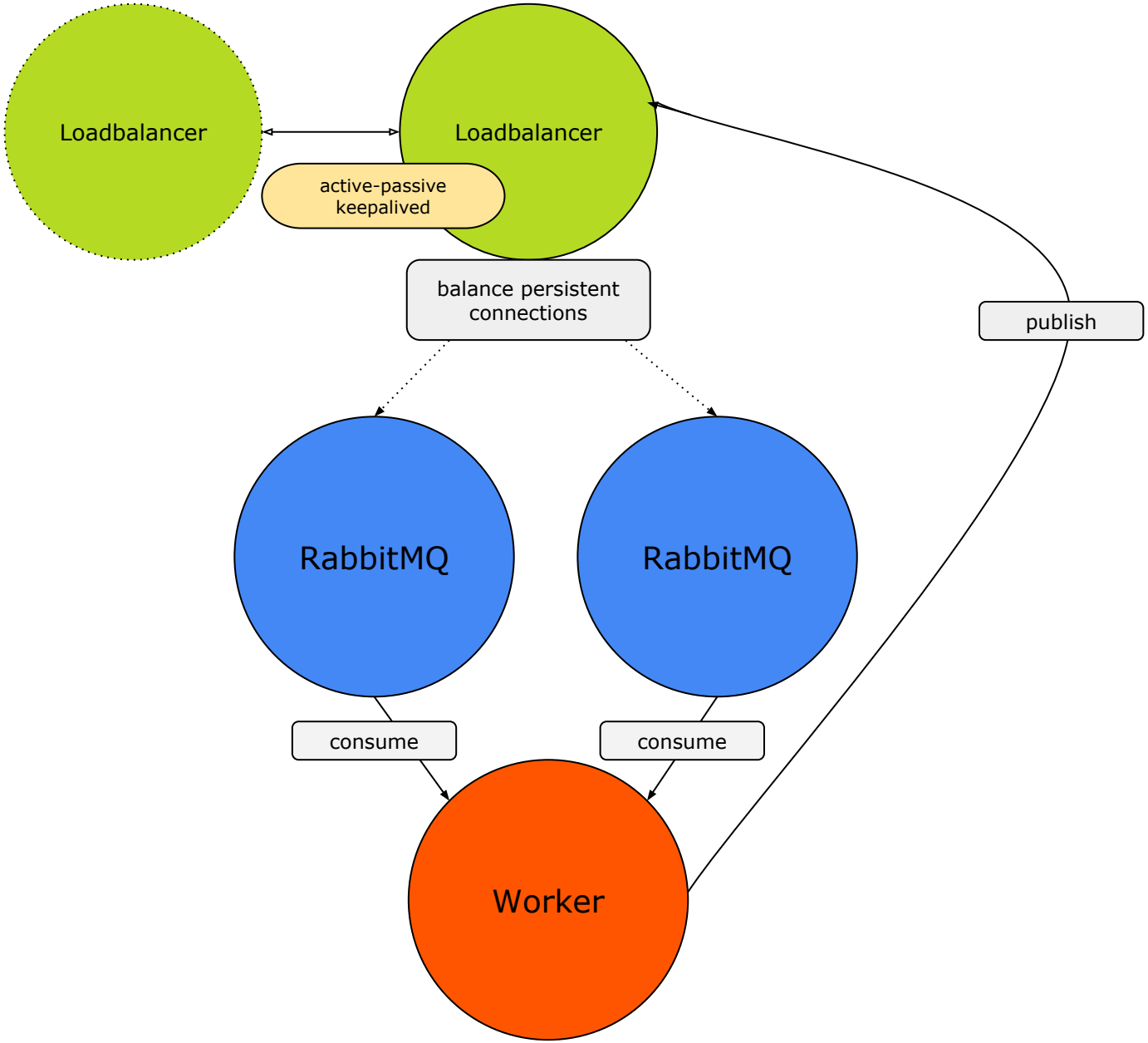
... we're down - part 2

high(er) availability

add scalability...

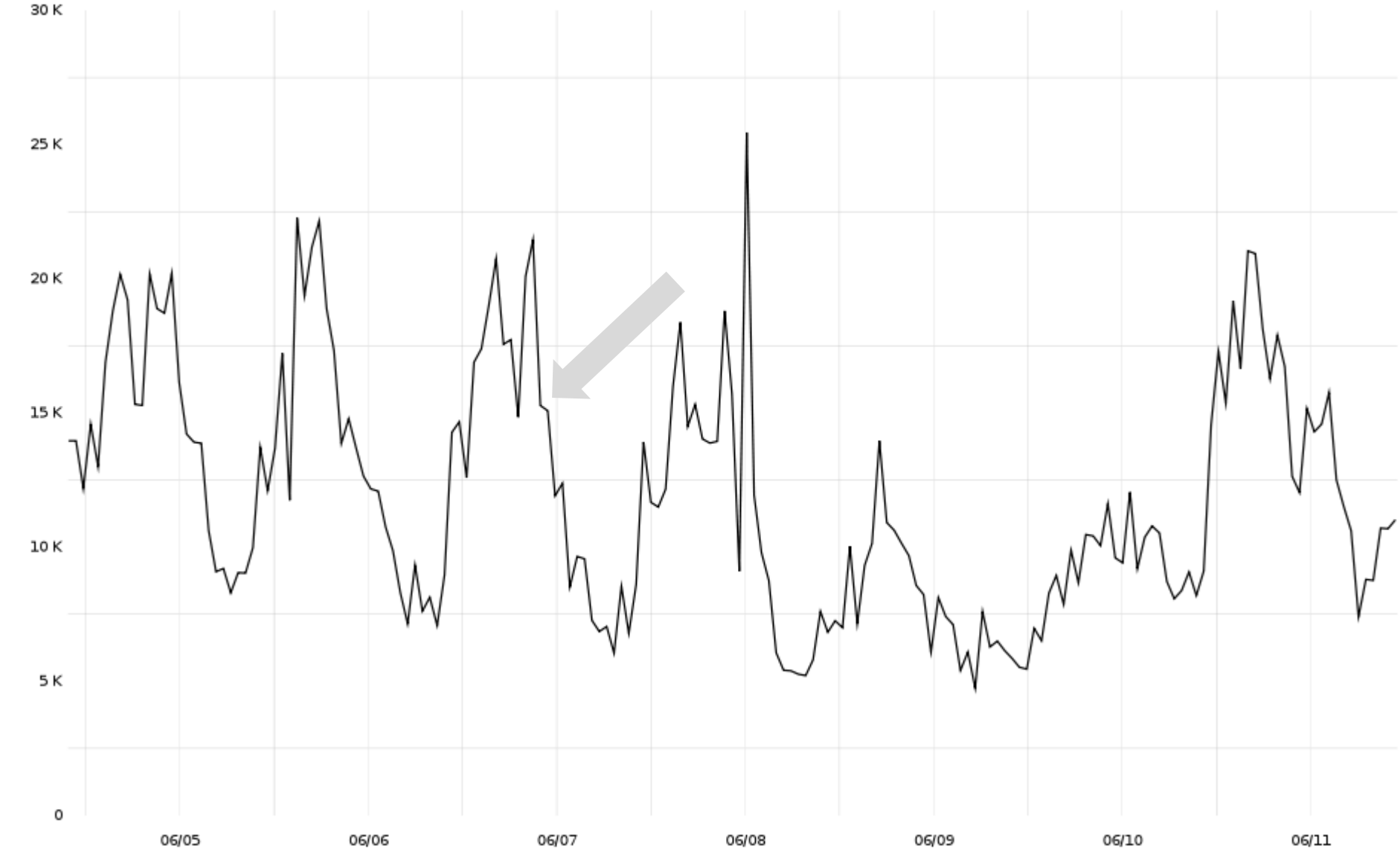
clustering rabbitmq

multiple brokers as cluster
publish one
subscribe many


Loadbalancer
Loadbalancer
active-passive
keepalived
balance persistent
connections
publish
RabbitMQ
RabbitMQ
consume
consume
Worker

client changes

tcp connection per broker
protocol heartbeats
reconnect



benefits

simple
all logic in client
availability
scalability

platforms

ruby
clojure, scala
erlang
go

further work

more clusters
commodity hardware
semantic events
discovery

Thank you!