
The Future of Microblogging

Jack Magiera, Jon Richelsen

Idea

The human race needs a way to share every aspect of one's life at all times.

If you're not sharing your pictures and words with the entire world, then what's even the point?

Twitter is good, but not good enough. Tweets are wordy and cumbersome.

Problem

Need a **simple, distributed, scalable** system to allow for constant microblogging and social validation.

Sharing pictures and text **at all times**

Deliver an infrastructure that can manage **high traffic usage** and scale to **large numbers of users**

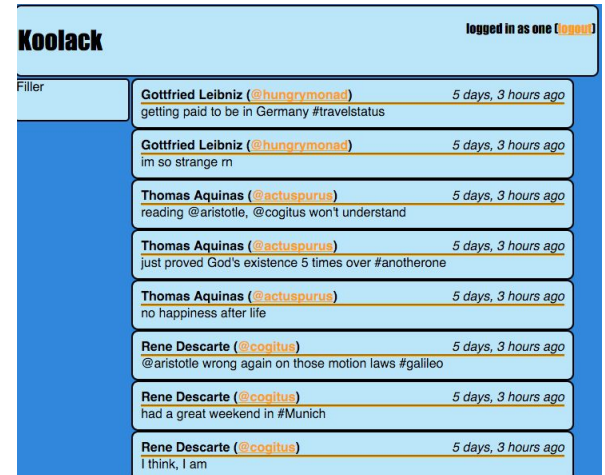
Koolack

Microblogging all the time, even
right now.

Koolack

Microblogging service - Twitter, but
with dynamics and synergy

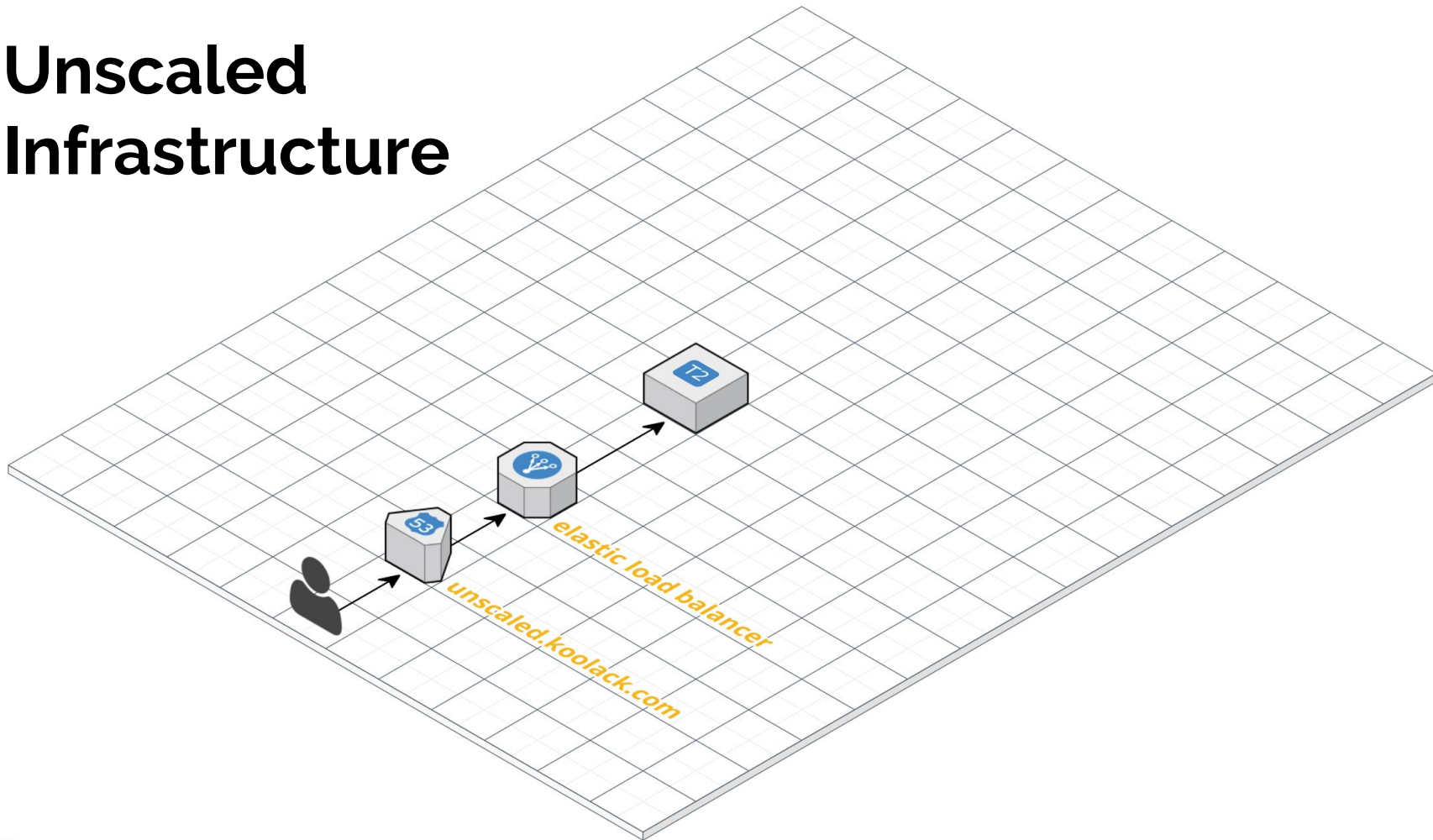
Users post Kools, can
follow/unfollow other users, and can
“ack” others’ Kools



The screenshot shows the Koolack web interface. At the top, the word "Koolack" is displayed in a bold, black font on the left, and "logged in as one [logout]" is on the right. Below this is a search bar labeled "Filter". The main content area displays a list of seven posts, each in a light blue box with a thin border. Each post includes the user's name and handle in orange, the text of the post, and the time it was posted. The posts are as follows:

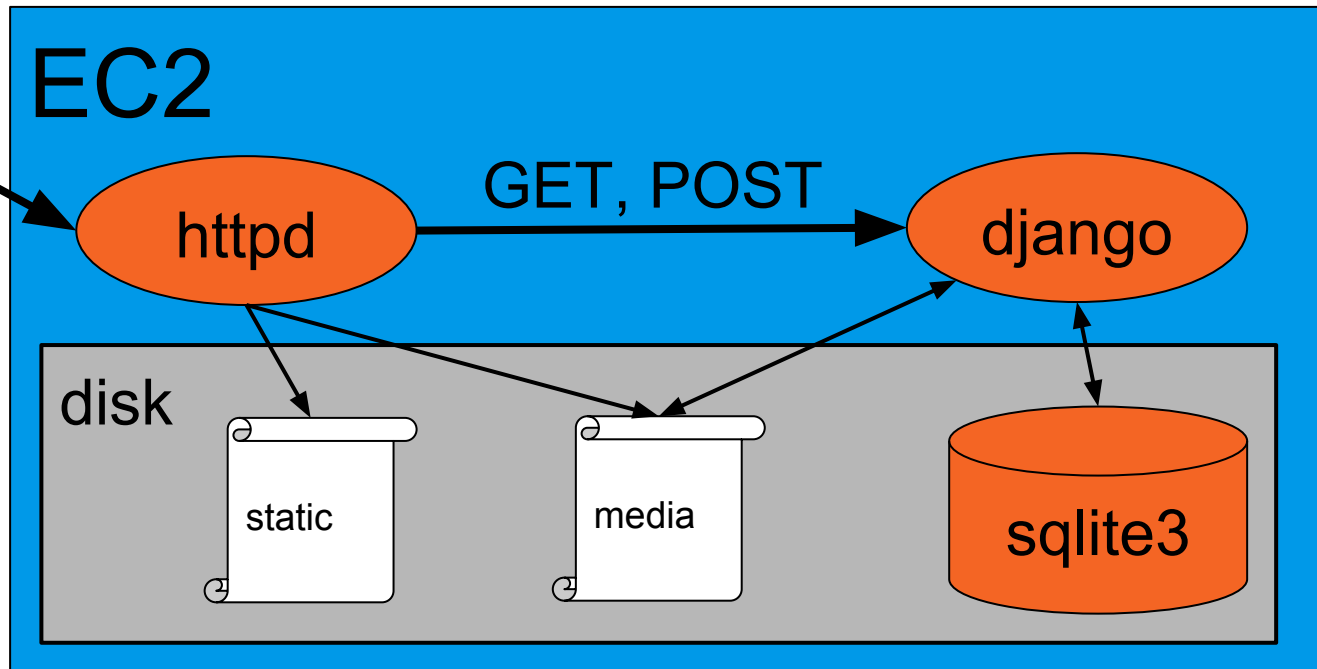
User	Post Text	Time
Gottfried Leibniz (@hungrymonad)	getting paid to be in Germany #travelstatus	5 days, 3 hours ago
Gottfried Leibniz (@hungrymonad)	im so strange rn	5 days, 3 hours ago
Thomas Aquinas (@actuspurus)	reading @aristotle, @cogitus won't understand	5 days, 3 hours ago
Thomas Aquinas (@actuspurus)	just proved God's existence 5 times over #anotherone	5 days, 3 hours ago
Thomas Aquinas (@actuspurus)	no happiness after life	5 days, 3 hours ago
Rene Descarte (@cogitus)	@aristotle wrong again on those motion laws #galileo	5 days, 3 hours ago
Rene Descarte (@cogitus)	had a great weekend in #Munich	5 days, 3 hours ago
Rene Descarte (@cogitus)	I think, I am	5 days, 3 hours ago

Unscaled Infrastructure

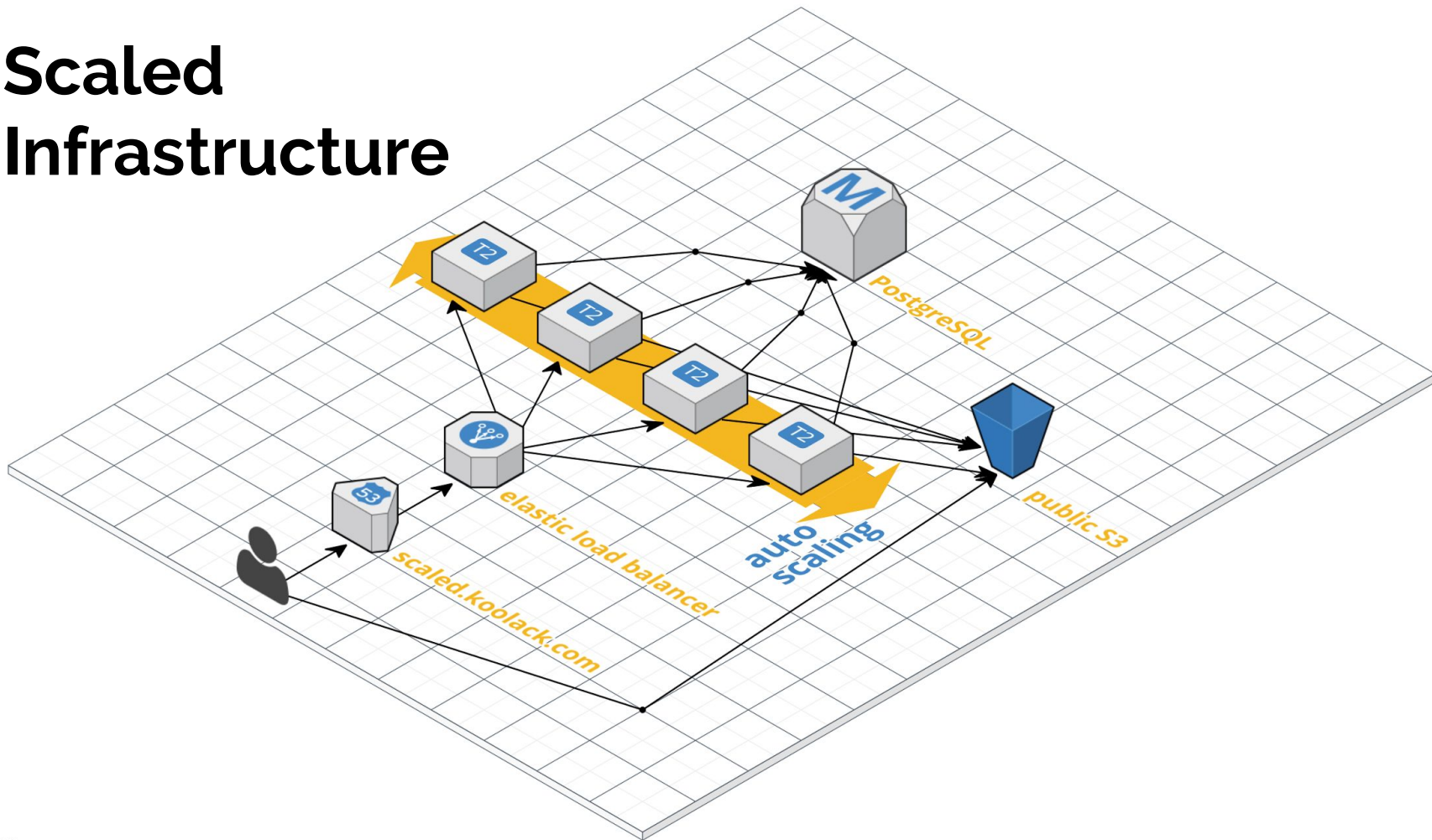


Detail of Unscaled Infrastructure

incoming request



Scaled Infrastructure



Django ORM with NoSQL?

SQL	Django API
<code>SELECT * FROM kools WHERE id = 12343</code>	<code>my_kool = Kool.objects.get(id=12343)</code>
<code>SELECT author FROM kools WHERE id = 12343</code>	<code>my_author = my_kool.author</code>
<code>UPDATE kools SET content = 'updated content' WHERE id = 12343</code>	<code>my_kool.content = 'updated content'</code> <code>my_kool.save()</code>
<code>SELECT * FROM kools, follows WHERE kools.author = follows.followee AND follows.follower = 'username'</code>	<code>Kool.objects.filter(author__followed_by=my_author)</code>

Attack methodology

Use PhantomJS to test high traffic

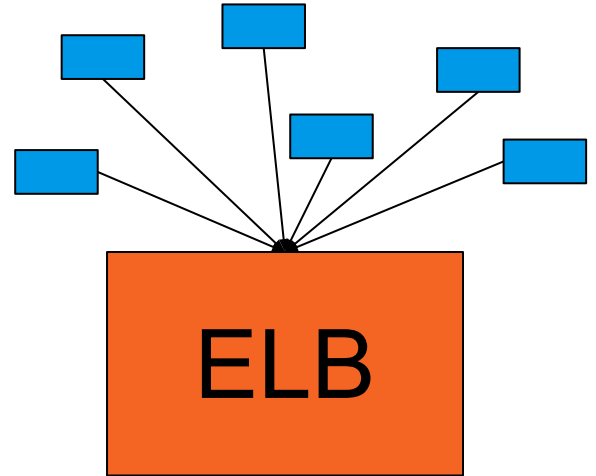
Create scripts that automate
standard user processes

Randomize elements of these scripts
and run them simultaneously and/or
subsequently

Use internal testing features to
measure performance on scaled vs
unscaled

Performance Metrics

- client-side performance or server-side performance?
- Elastic Load Balancing + Amazon CloudWatch = metrics!
- req/s measures overall performance
- increase number of clients until req/s reaches limit
- Elastic Beanstalk to add more EC2



One more thing...

Future plans

Neaten up interface - aesthetic and functional purpose

Finalize scaling

Complete testing
