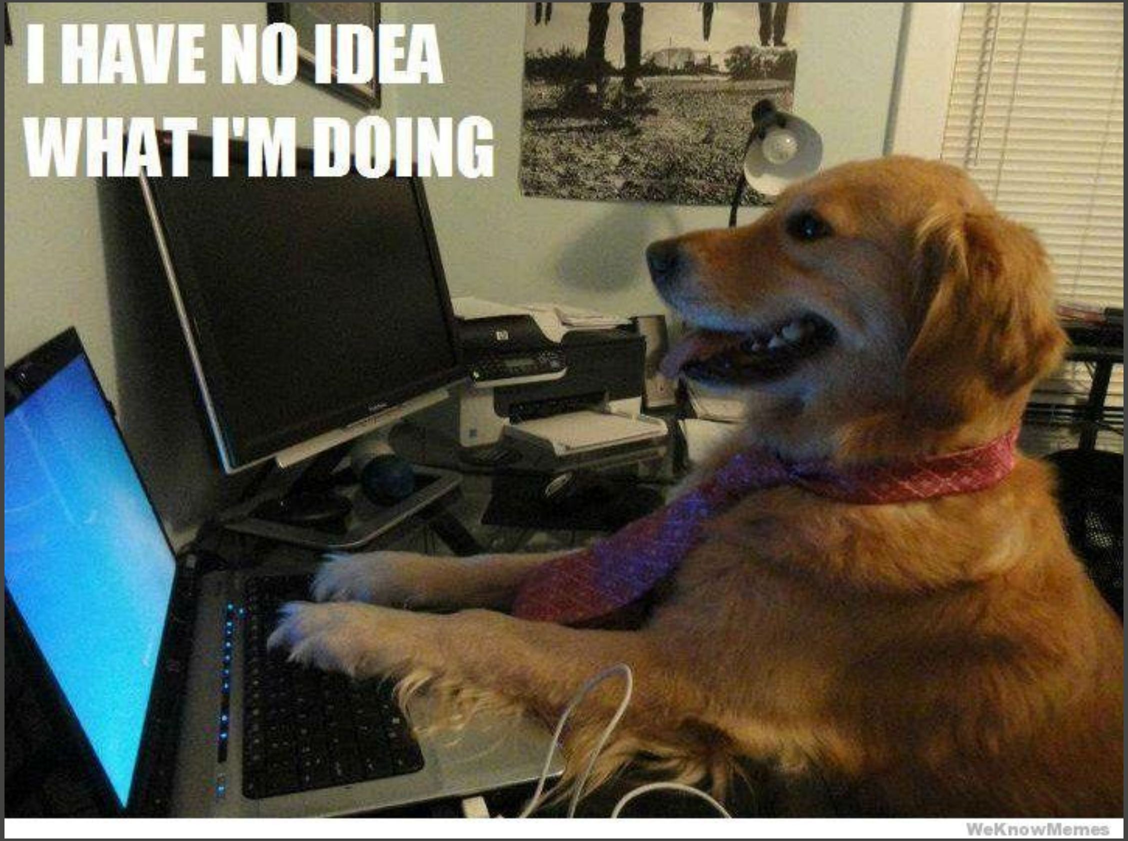
Scaling on AWS for the First 10 Million Users

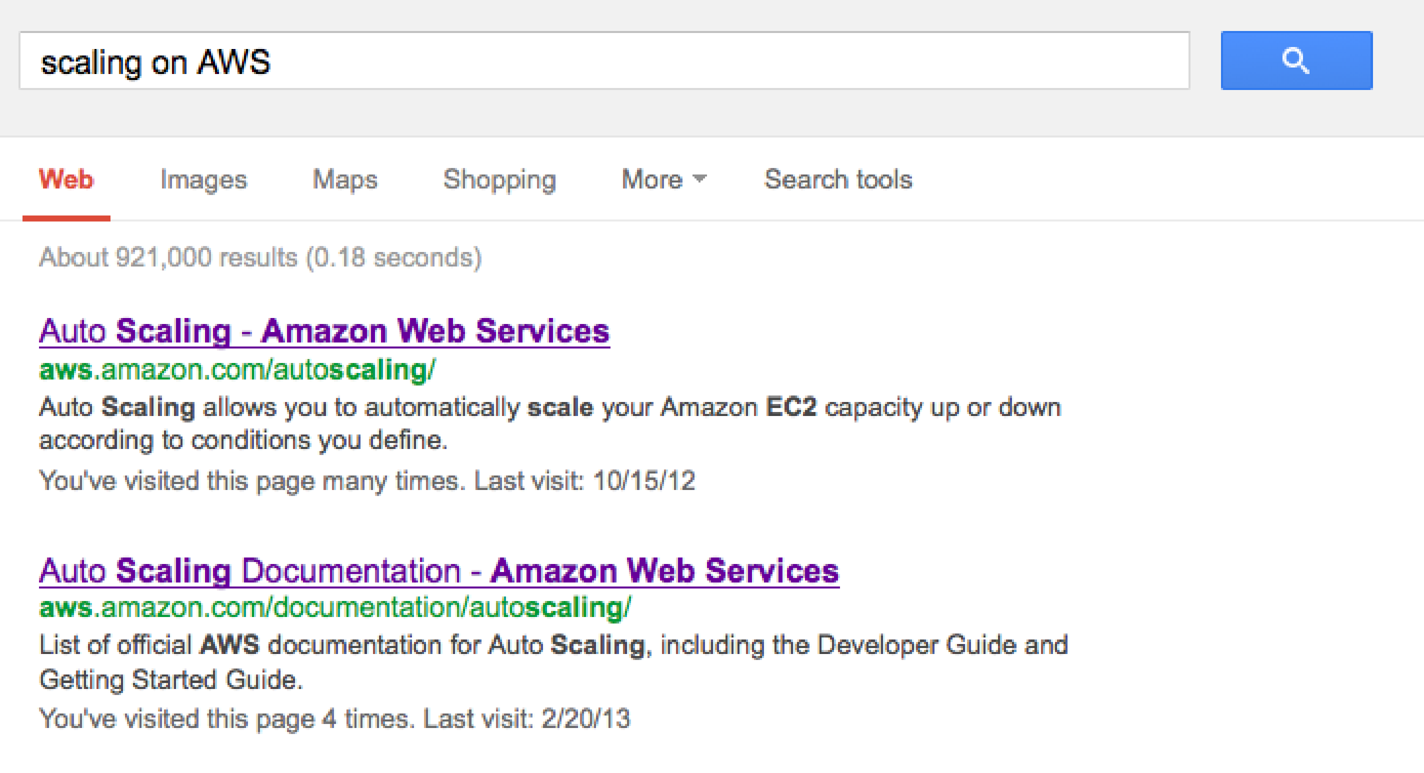
Scaling on AWS for the First 10 Million Users

- ME: – Solutions Architect – Amazon Web Services – jman@amazon.com
- YOU: Here to learn more about scaling infrastructure on AWS
- TODAY: about best practices and things to think about when building for large scale

So how do we scale?



amazon web services



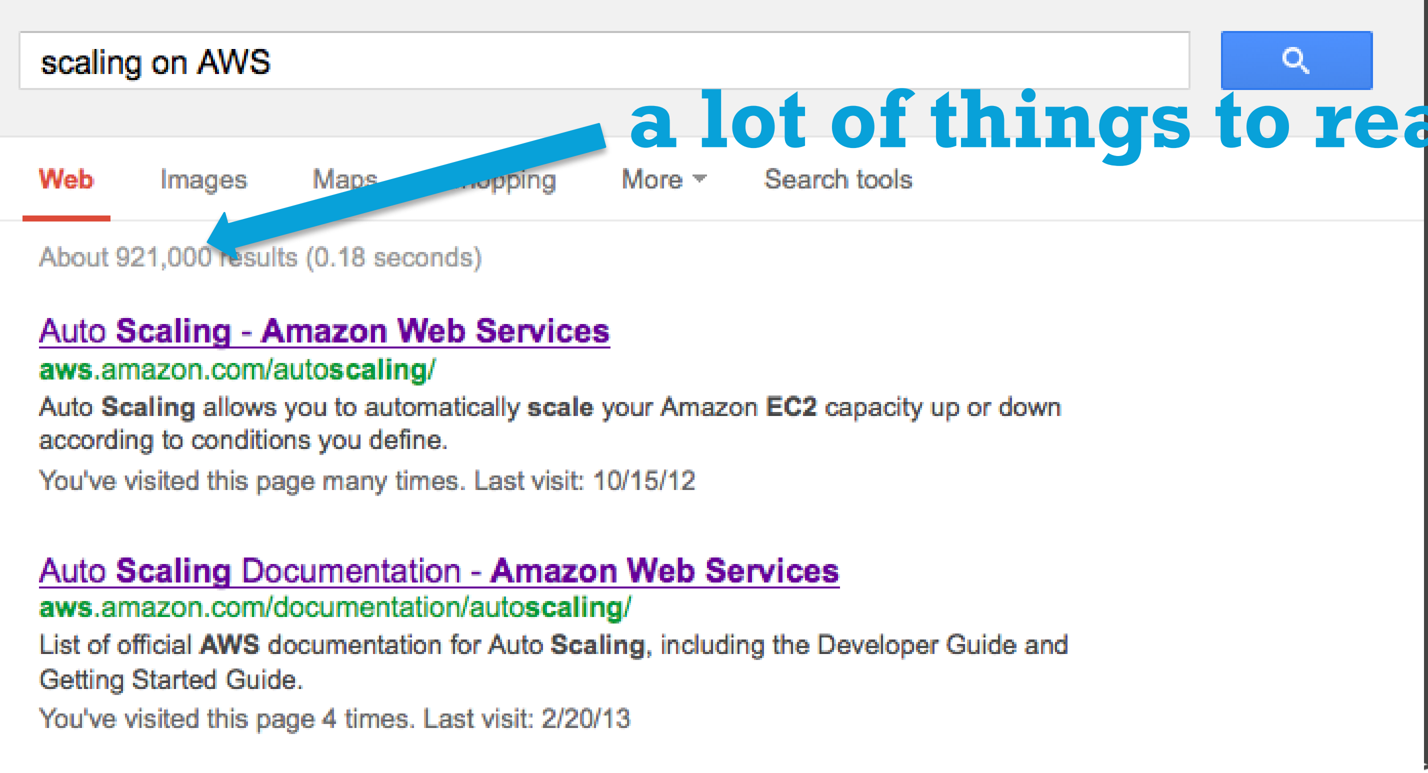
scaling on AWS

About 921,000 results (0.18 seconds)

Auto **Scaling** - **Amazon Web Services**
aws.amazon.com/auto**scaling**/
Auto **Scaling** allows you to automatically **scale** your Amazon **EC2** capacity up or down according to conditions you define.
You've visited this page many times. Last visit: 10/15/12

Auto **Scaling** Documentation - **Amazon Web Services**
aws.amazon.com/documentation/auto**scaling**/
List of official **AWS** documentation for Auto **Scaling**, including the Developer Guide and Getting Started Guide.
You've visited this page 4 times. Last visit: 2/20/13

a lot of things to read

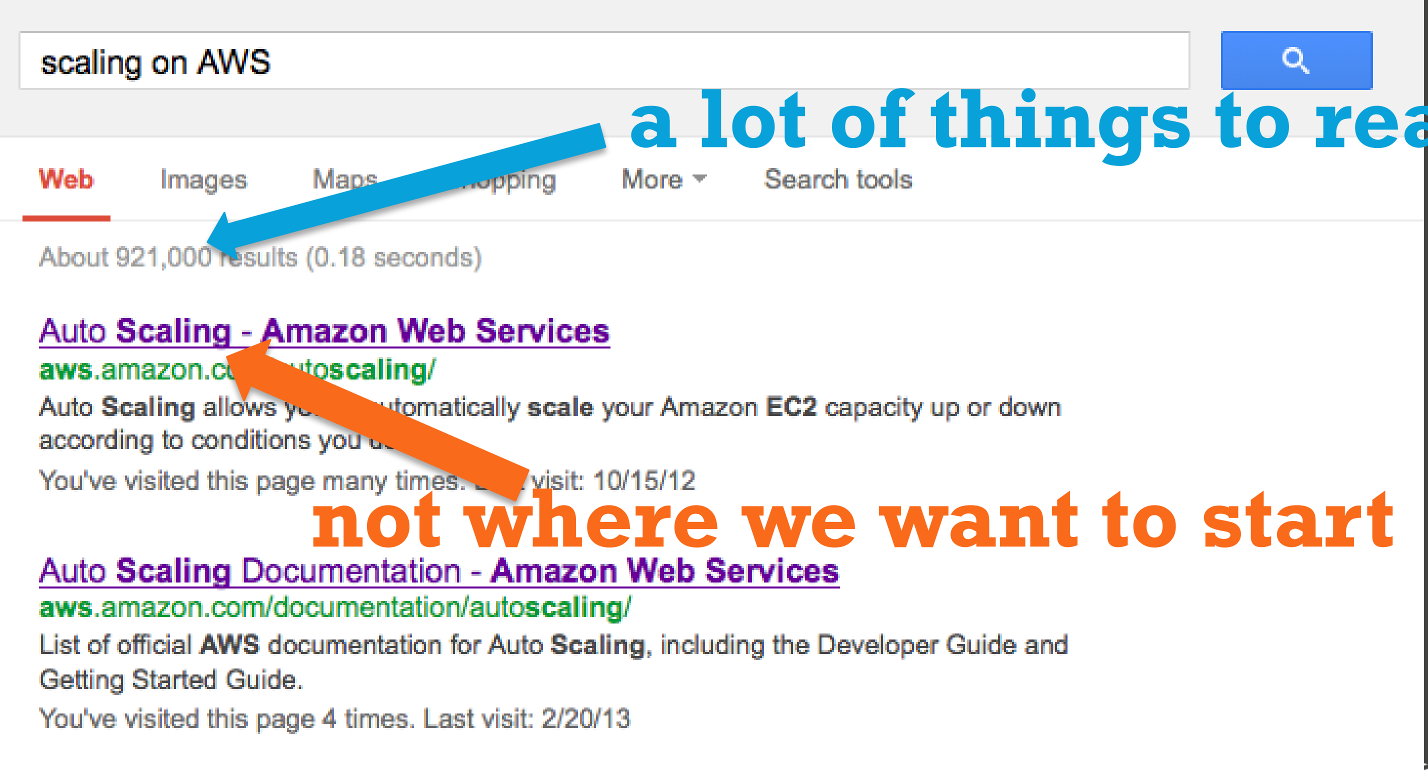


scaling on AWS

About 921,000 results (0.18 seconds)

Auto Scaling - Amazon Web Services
aws.amazon.com/autoscaling/
Auto **Scaling** allows you to automatically **scale** your Amazon **EC2** capacity up or down according to conditions you define.
You've visited this page many times. Last visit: 10/15/12

Auto Scaling Documentation - Amazon Web Services
aws.amazon.com/documentation/autoscaling/
List of official **AWS** documentation for Auto **Scaling**, including the Developer Guide and Getting Started Guide.
You've visited this page 4 times. Last visit: 2/20/13



About 921,000 results (0.18 seconds)

Auto Scaling - Amazon Web Services
aws.amazon.c utoscaling/
Auto Scaling allows y utomatically scale your Amazon EC2 capacity up or down according to conditions you
You've visited this page many times. visit: 10/15/12

Auto Scaling Documentation - Amazon Web Services
aws.amazon.com/documentation/autoscaling/
List of official AWS documentation for Auto Scaling, including the Developer Guide and Getting Started Guide.
You've visited this page 4 times. Last visit: 2/20/13

a lot of things to read

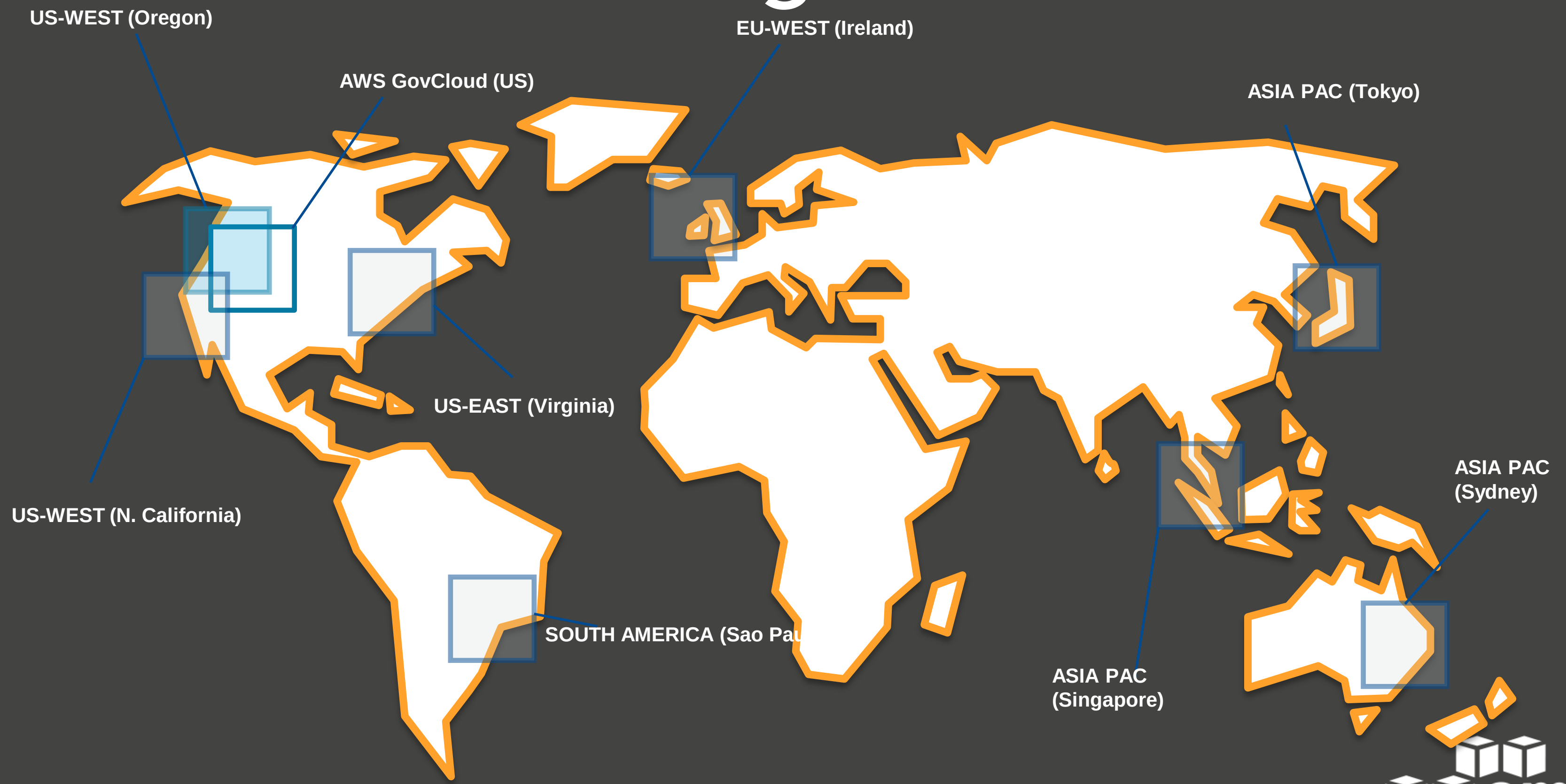
not where we want to start

Auto-Scaling is a tool and a destination. It’s not the single thing that fixes everything.

What do we need first?

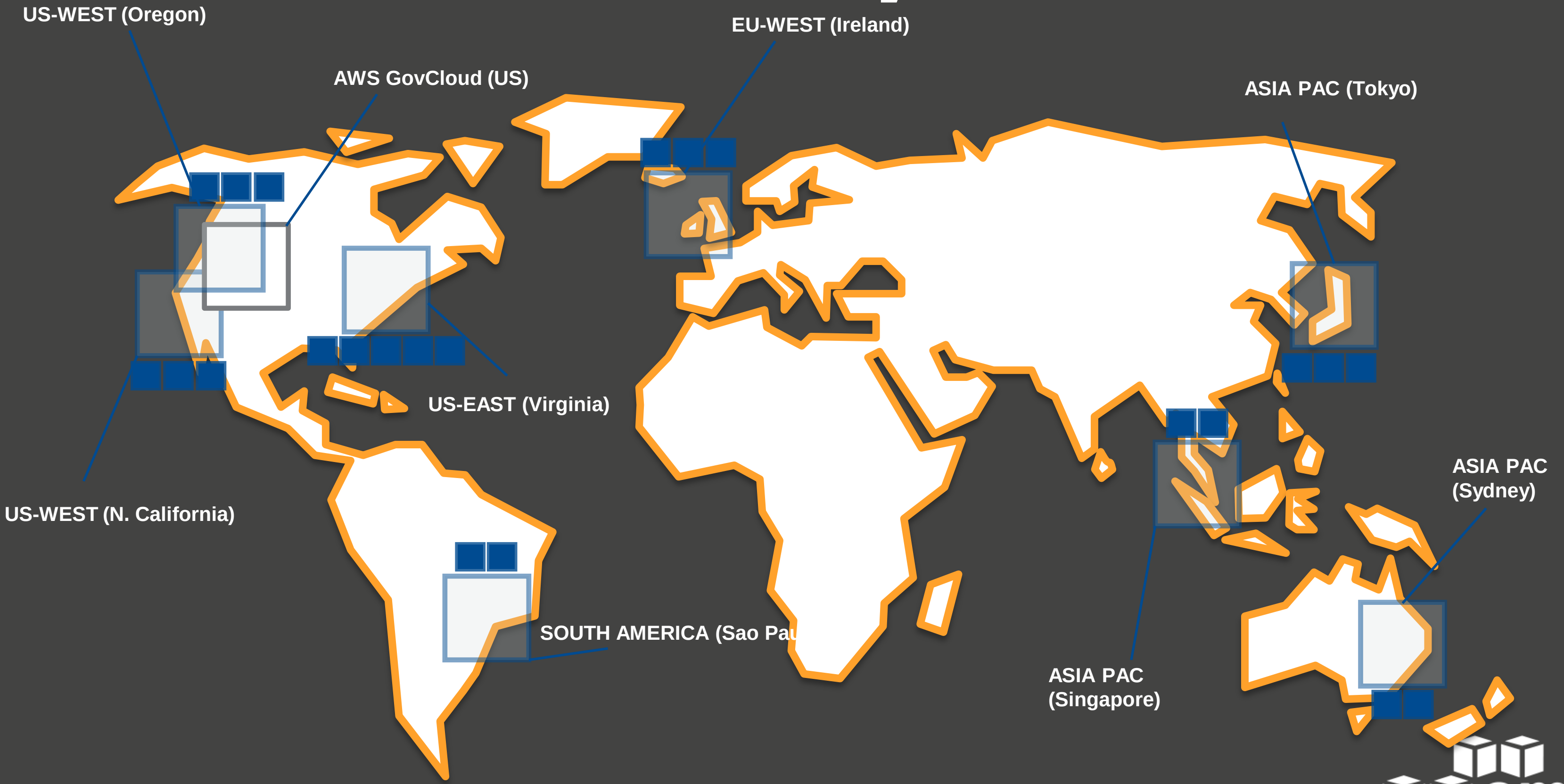
Some basics:

Regions



amazon
web services

Availability Zones



amazon
web services

2003

amazon.com

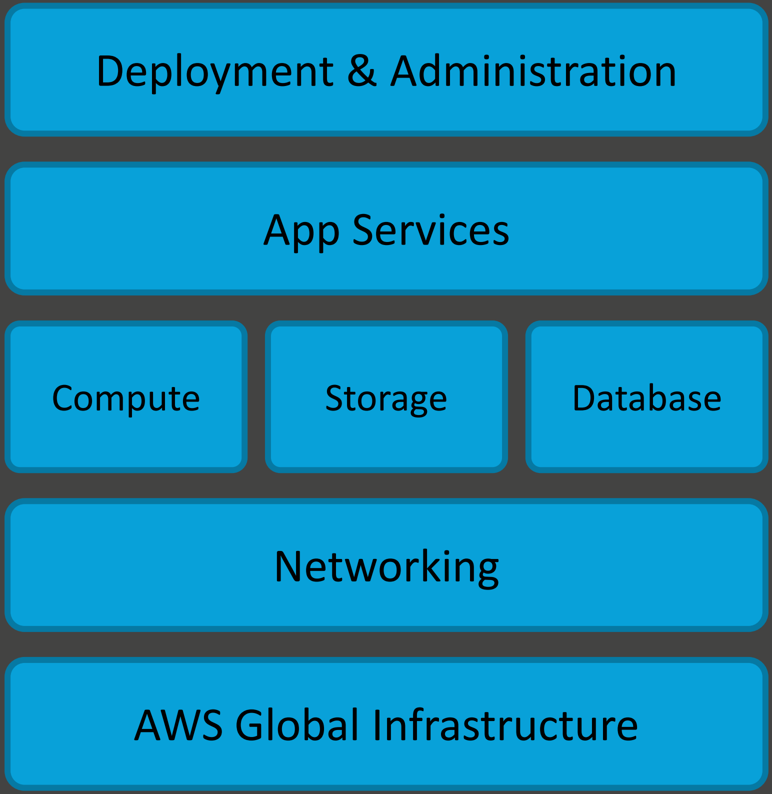
- $5.2B retail business
- 7,800 employees
- A whole lot of servers

2013

amazon web services

Every day, AWS adds enough server capacity to power that whole **$5B** enterprise

amazon web services



Deployment & Administration

App Services

Compute

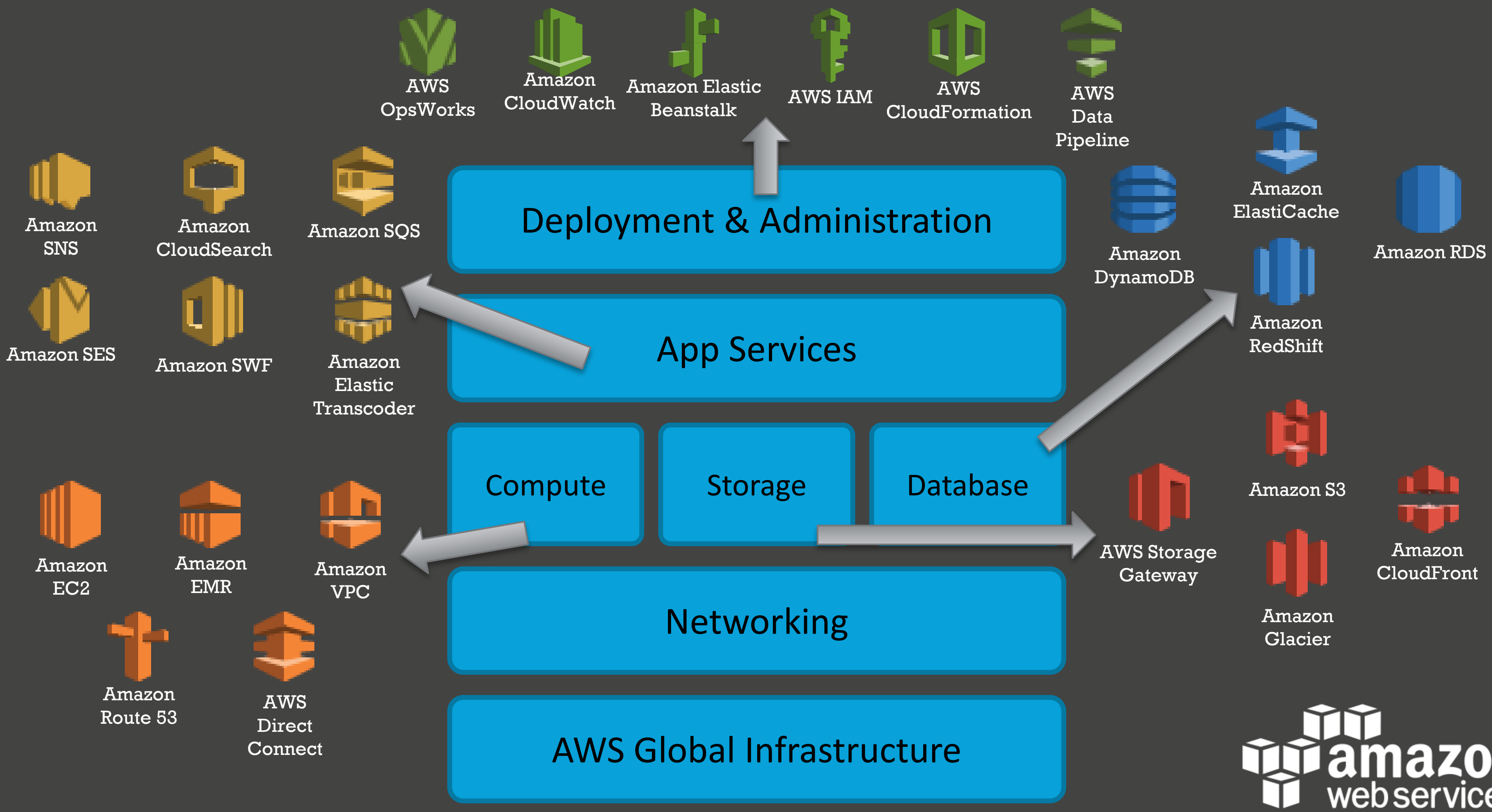
Storage

Database

Networking

AWS Global Infrastructure

amazon
web services



Deployment & Administration

AWS OpsWorks

Amazon CloudWatch

Amazon Elastic Beanstalk

AWS IAM

AWS CloudFormation

AWS Data Pipeline

App Services

Amazon SNS

Amazon CloudSearch

Amazon SQS

Amazon SES

Amazon SWF

Amazon Elastic Transcoder

Compute

Amazon EC2

Amazon EMR

Amazon VPC

Amazon Route 53

AWS Direct Connect

Storage

AWS Storage Gateway

Amazon S3

Amazon CloudFront

Amazon Glacier

Database

Amazon DynamoDB

Amazon ElastiCache

Amazon RDS

Amazon RedShift

Networking

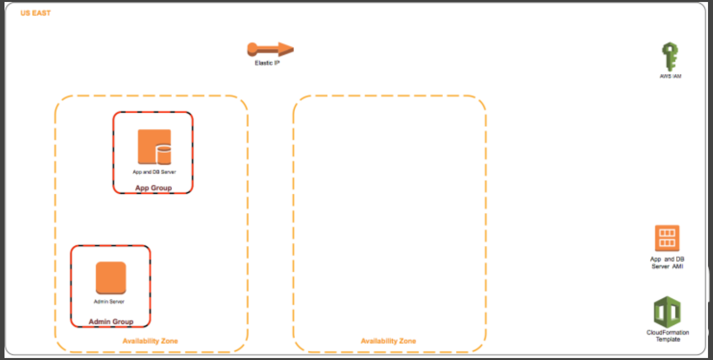
AWS Global Infrastructure

amazon web services

So let's start from day one, user one (you):

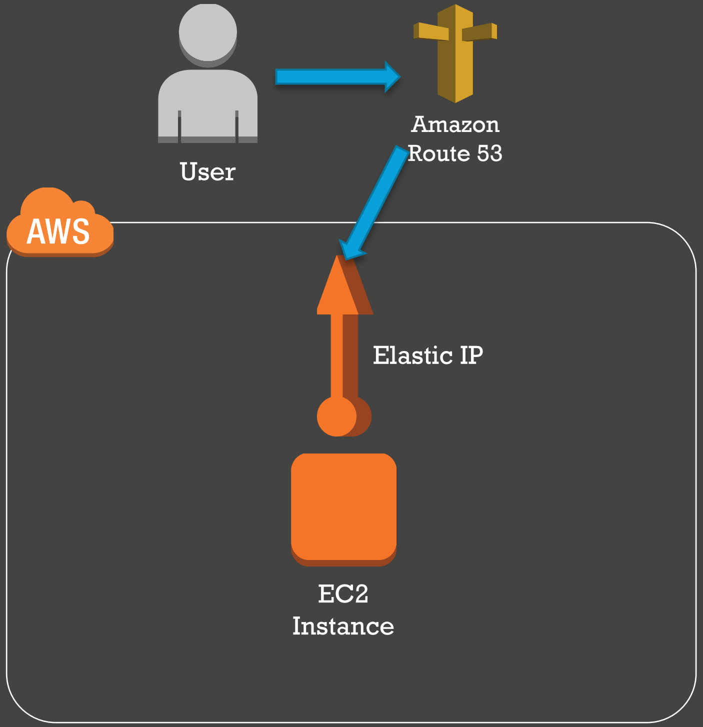
Lab 1

- AWS Setup
  - security groups
  - key pairs
- Application Setup
  - Insoshi
  - MySQL



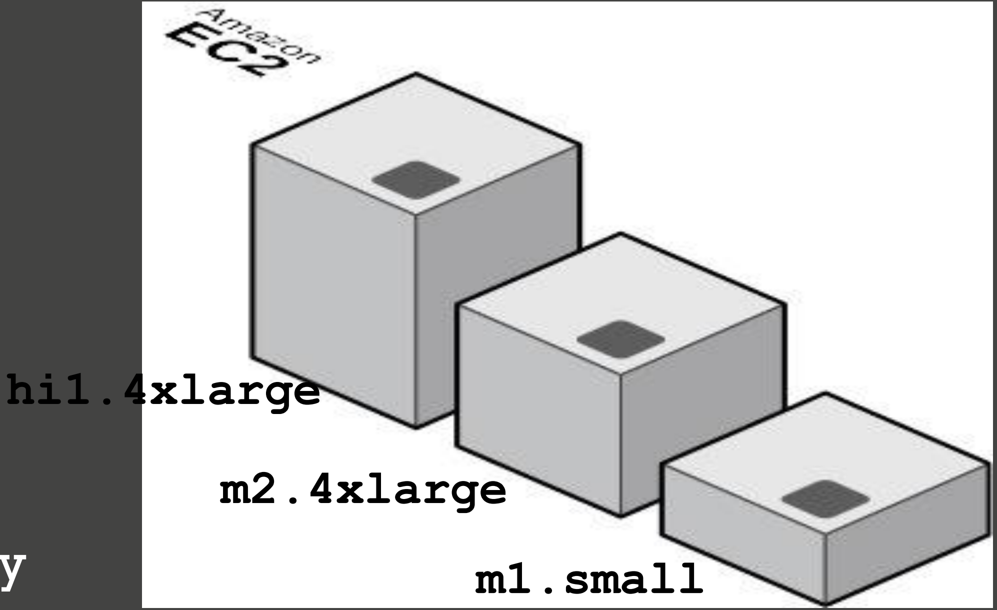
Day One, User One:

- A single EC2 Instance
  - With full stack on this host
    - Web App
    - Database
    - Management
    - Etc.
- A single Elastic IP
- Route53 for DNS



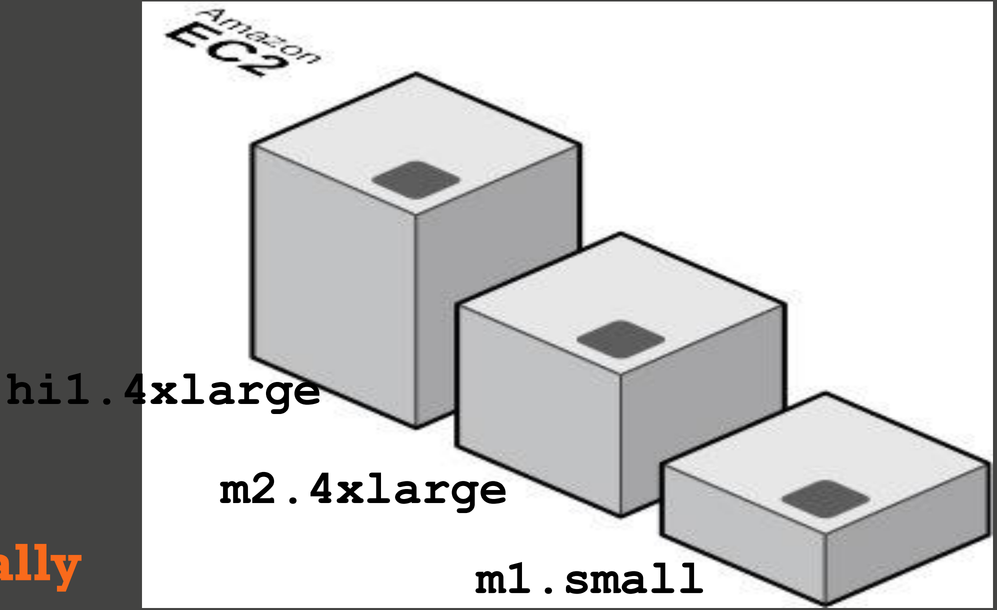
“We’re gonna need a bigger box”

- Simplest approach
- Can now leverage PIOPs
- High I/O instances
- High Memory instances
- High CPU instances
- High storage instances
- Easy to change instance sizes
- Will hit an endpoint eventually



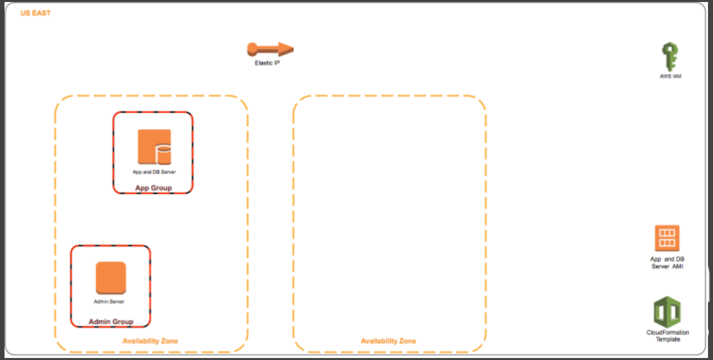
“We’re gonna need a bigger box”

- Simplest approach
- Can now leverage PIOPs
- High I/O instances
- High memory instances
- High CPU instances
- High storage instances
- Easy to change instance sizes
- **Will hit an endpoint eventually**



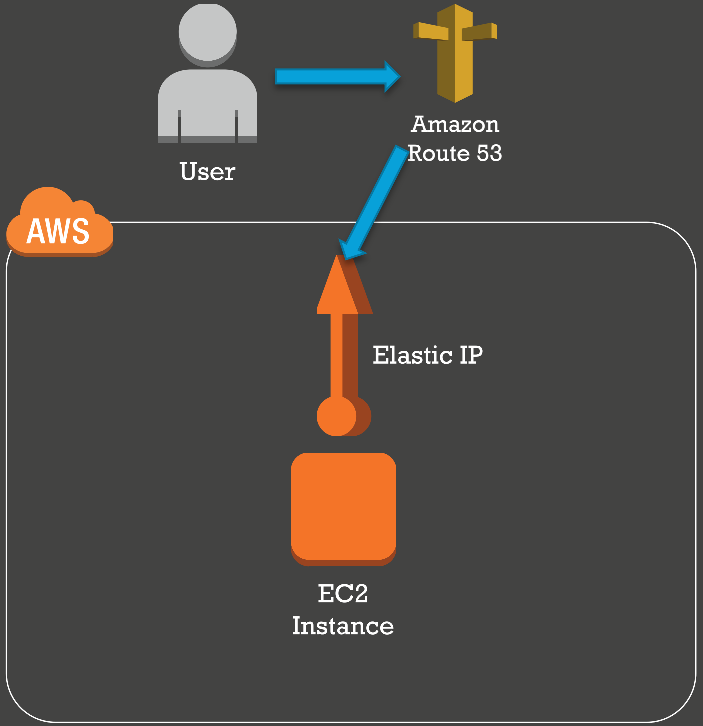
Lab 2

- Instance Resizing
  - Impact to service availability
  - Impact to data persistence



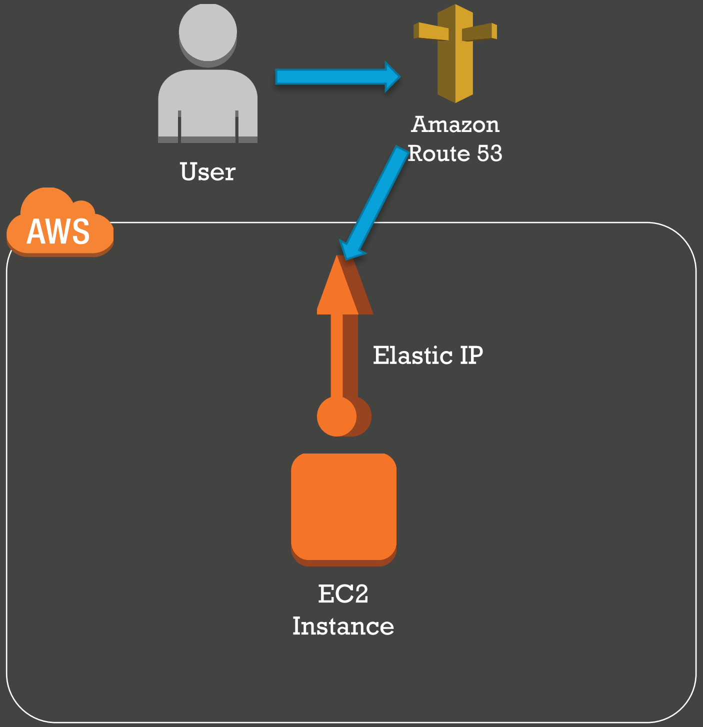
Day One, User One:

- We could potentially get to a few hundred to a few thousand depending on application complexity and traffic
- No failover
- No redundancy
- Too many eggs in one basket



Day One, User One:

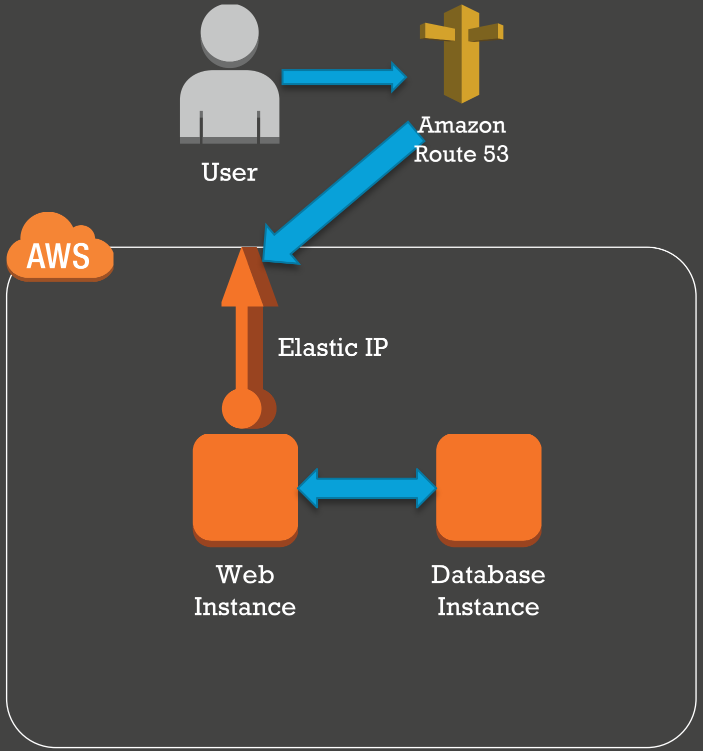
- We could potentially get to a few hundred to a few thousand depending on application complexity and traffic
- **No failover**
- **No redundancy**
- **Too many eggs in one basket**



Day Two, User > 1:

First let's separate out our single host into more than one.

- Web
- Database
  - Make use of a database service?



Database Options

Self-Managed

Database Server on Amazon EC2

Your choice of database running on Amazon EC2

Bring Your Own License (BYOL)

Fully-Managed

Amazon RDS

Microsoft SQL, Oracle or MySQL as a managed service

Flexible licensing BYOL or License Included

Amazon DynamoDB

Managed NoSQL database service using SSD storage

Seamless scalability Zero administration

Amazon Redshift

Massively parallel, petabyte-scale, data warehouse service

Fast, powerful and easy to scale

But how do I choose what DB technology I need? SQL? NoSQL?

Some folks won’t like this. But…

Start with SQL databases

But, but, but, but…

No. You don’t.

Start with SQL databases

Why start with SQL?

- Established and well worn technology
- Lots of existing code, communities, books, background, tools, etc
- You aren't going to break SQL DBs in your first 10 million users. No really, you won't*
- Clear patterns to scalability

*Unless you are doing something SUPER weird with the data or MASSIVE amounts of it, even then SQL will have a place in your stack

AH HA! You said “massive amounts”, I will have massive amounts!

If your usage is such that you will be generating several TB (>5) of data in the first year OR have an incredibly data intensive workload you might need NoSQL

Why else might you need NoSQL?

- Super low latency applications
- Metadata driven datasets
- Highly-unrelational data
- Need schema-less data constructs*
- Massive amounts of data (again, in the TB range)
- Rapid ingest of data (thousands of records/sec)

*Need != “its easier to do dev without schemas”

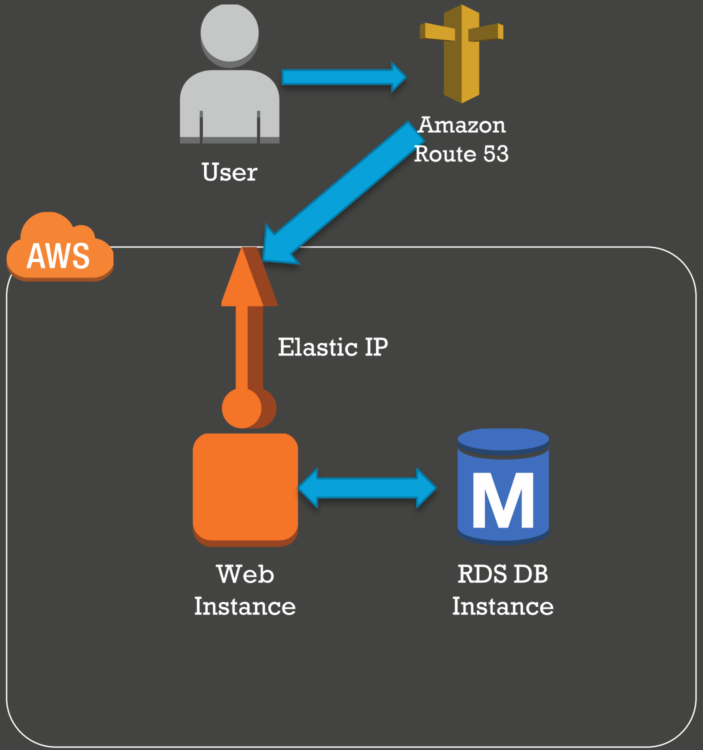
But this is probably less than 90% of you

Unless everyone of you is building semantic/big data websites

User >100:

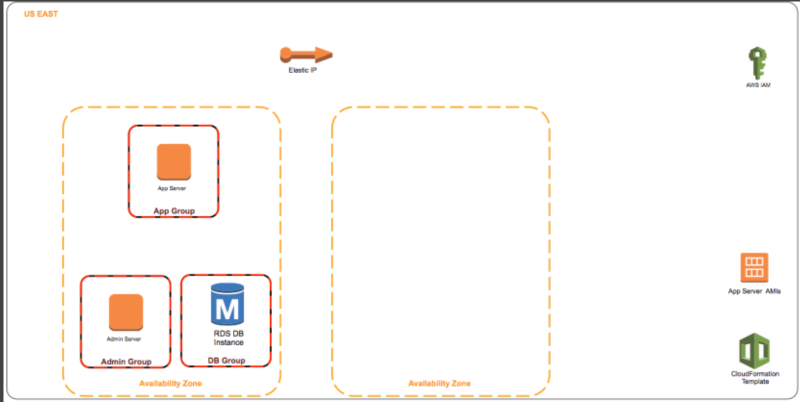
First let's separate out our single host into more than one.

- Web
- Database
  - Use RDS to make your life easier



Lab 3 RDS

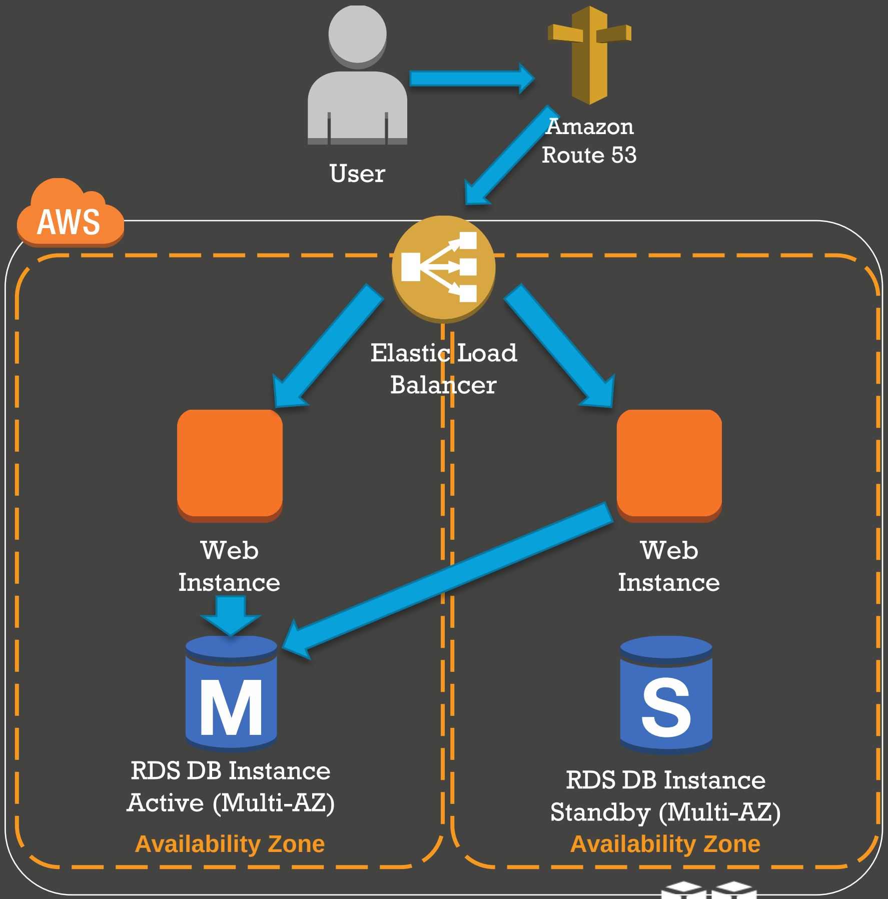
- Review
  - differences for running MySQL on RDS, if any
  - current limitation of you architecture



User > 1000:

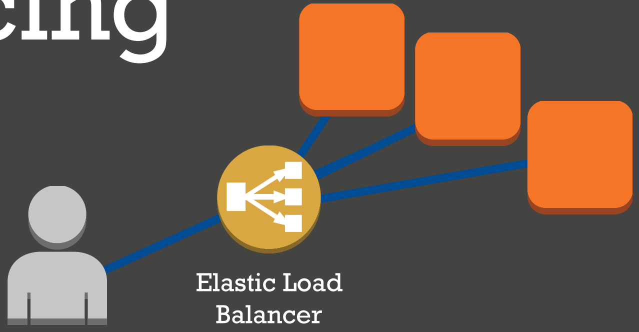
Next let's address our lack of failover and redundancy issues:

- Elastic Load Balancer
- Another Web Instance
  - In another Availability Zone
- Enable RDS Multi-AZ



Elastic Load Balancing

- Create highly scalable applications
- Distribute load across EC2 instances in multiple availability zones

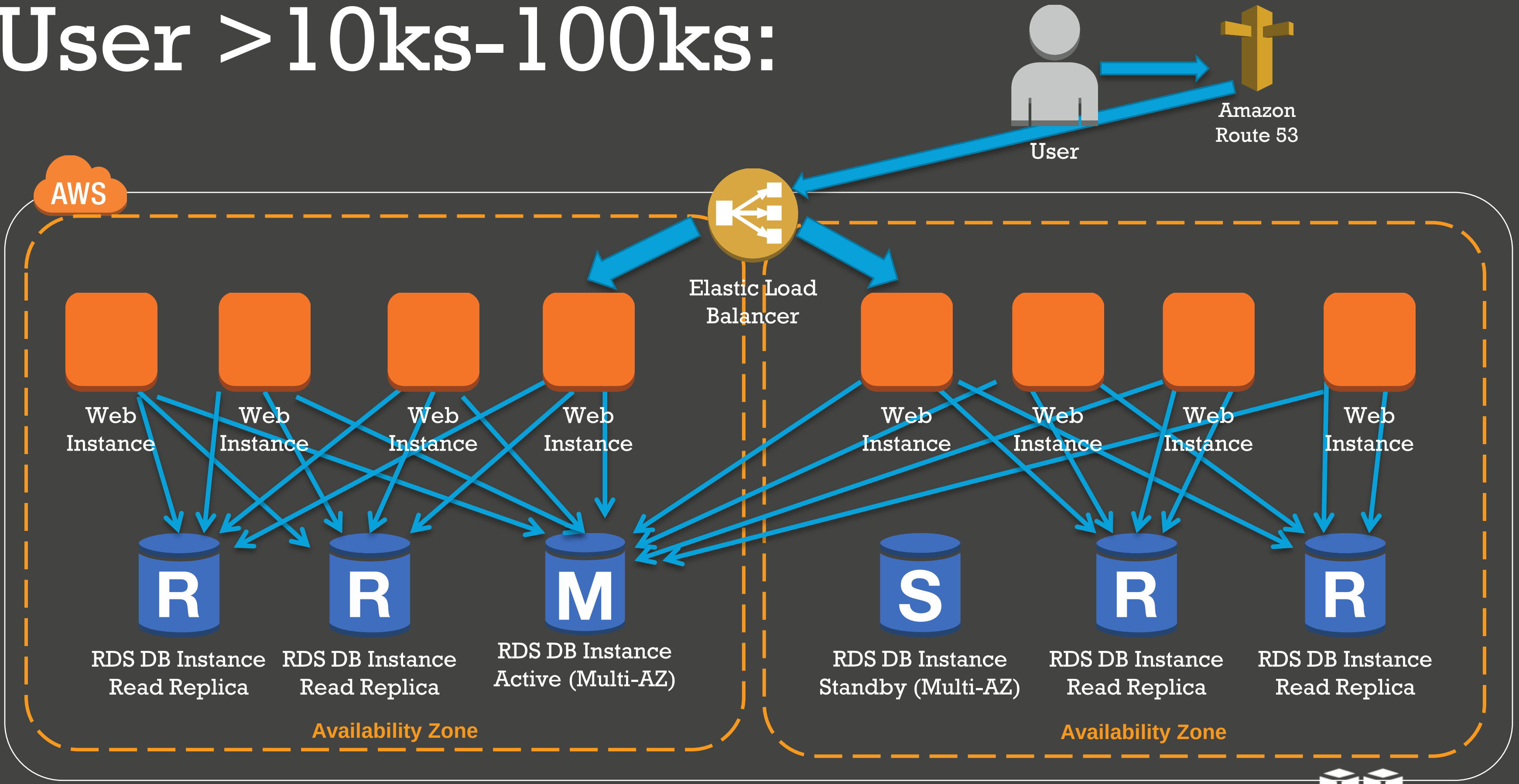


Feature	Details
Available	Load balance across instances in multiple Availability Zones
Health checks	Automatically checks health of instances and takes them in or out of service
Session stickiness	Route requests to the same instance
Secure sockets layer	Supports SSL offload from web and application servers with flexible cipher support
Monitoring	Publishes metrics to CloudWatch

Scaling this horizontally and vertically will get us pretty far

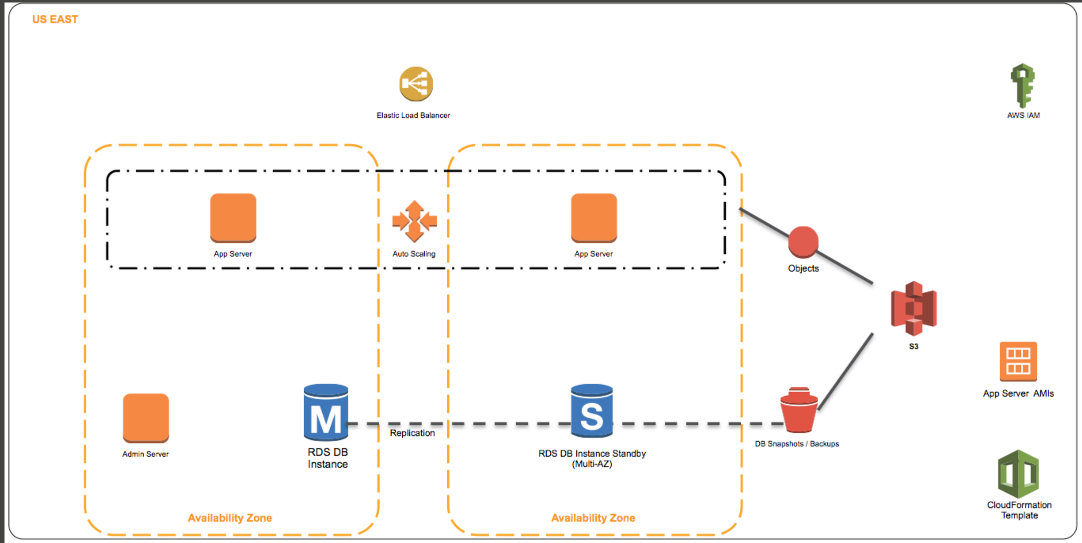
(10s-100s of thousands)

User >10ks-100ks:


User
Amazon Route 53
AWS
Elastic Load Balancer
Web Instance
Web Instance
Web Instance
Web Instance
Web Instance
Web Instance
Web Instance
Web Instance
R
RDS DB Instance Read Replica
R
RDS DB Instance Read Replica
M
RDS DB Instance Active (Multi-AZ)
S
RDS DB Instance Standby (Multi-AZ)
R
RDS DB Instance Read Replica
R
RDS DB Instance Read Replica
Availability Zone
Availability Zone

Lab 4: HA

- RDS behavior during failover
- Self-healing limitations

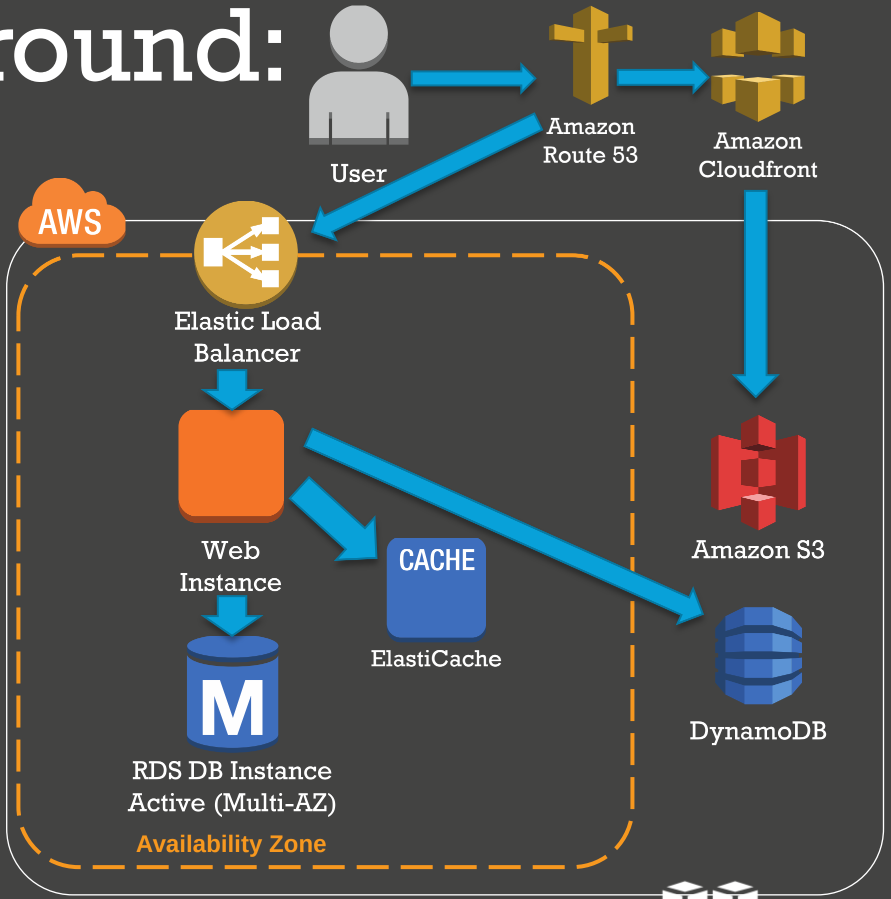


This will take us pretty far honestly, but we care about performance and efficiency, so let’s clean this up a bit

Shift some load around:

Let's lighten the load on our web and database instances:

- Move static content from the Web Instance to S3 and CloudFront
- Move session/state and DB caching to ElastiCache or DynamoDB



amazon web services

Working with S3 - Amazon Simple Storage Service

- Object based storage for the web
- 11 9s of durability
- Good for things like:
  - Static assets (css, js, images, videos)
  - Backups
  - Logs
  - Ingest of files for processing
- “Infinitely scalable”

- Supports fine grained permission control
- Ties in well with CloudFront
- Ties in with EMR
- Acts as a logging endpoint for S3/CloudFront/Billing
- Supports Encryption at transit and at rest
- Reduced Redundancy 1/3 cheaper
- Glacier for super long term storage

Jeff Barr @jeffbarr 18 Apr

Announced at AWS Summit - Amazon S3 now holds 2 trillion objects, processes 1.1 million requests / second: bit.ly/ZBN5k2 #awssummit

DynamoDB

- Provisioned throughput NoSQL database
- Fast, predictable performance
- Fully distributed, fault tolerant architecture
- Considerations for non-uniform data

Feature	Details
Provisioned throughput	Dial up or down provisioned read/write capacity.
Predictable performance	Average single digit millisecond latencies from SSD-backed infrastructure.
Strong consistency	Be sure you are reading the most up to date values.
Fault tolerant	Data replicated across Availability Zones.
Monitoring	Integrated to CloudWatch.
Secure	Integrates with AWS Identity and Access Management (IAM).
Elastic MapReduce	Integrates with Elastic MapReduce for complex analytics on large datasets.

amazon web services

ElastiCache

- Hosted Memcached
  - Speaks same API as traditional open source memcached
- Scale from one to many nodes
- Self healing (replaces dead instance)
- Very fast (single digit ms speeds usually (or less))
- Local to a single AZ
  - So need to run different clusters across different AZs
- Data is only in memory, so not persistent
- Use AWS's Auto Discovery client to simplify clusters growing and shrinking without affecting your application

CACHE

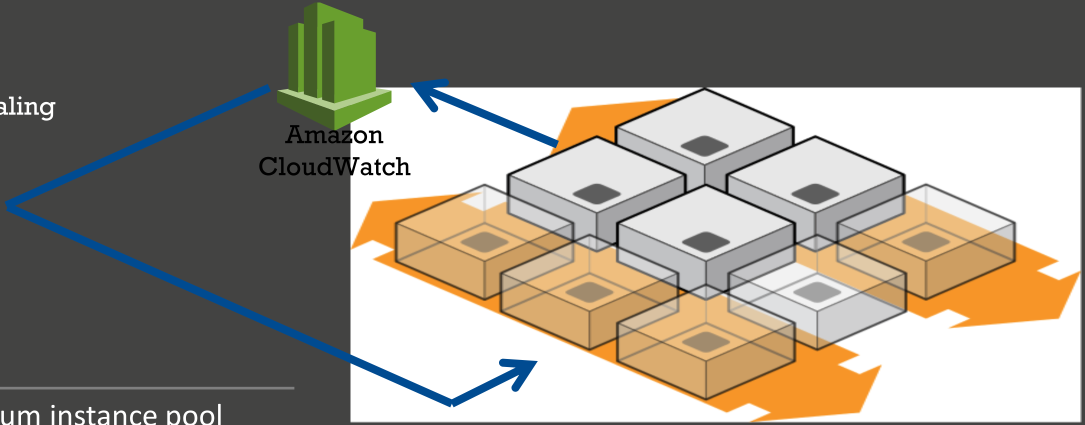
Now that our Web tier is much more lightweight, we can revisit the beginning of our talk…

Auto-Scaling!

Auto-Scaling

Trigger auto-scaling policy

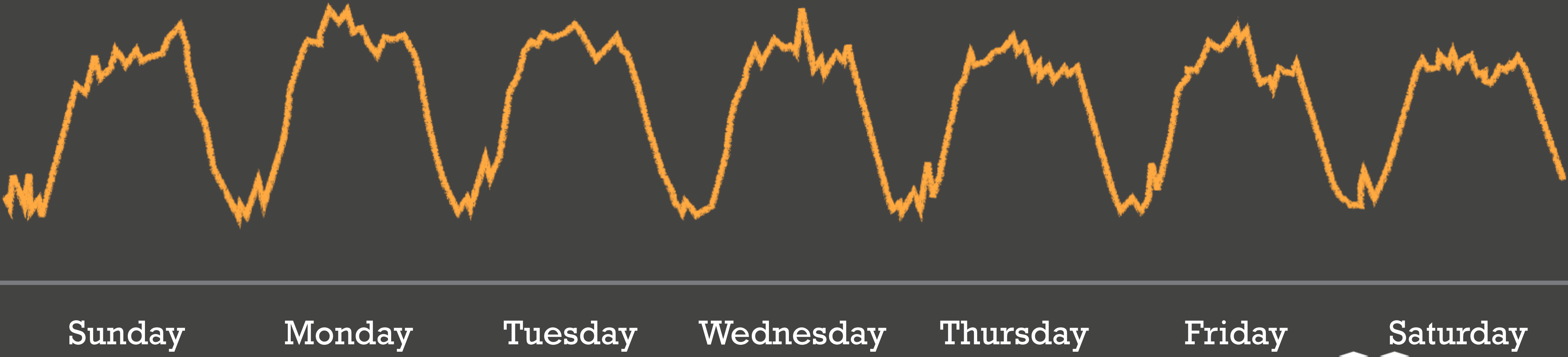
Automatic resizing of compute clusters based on demand



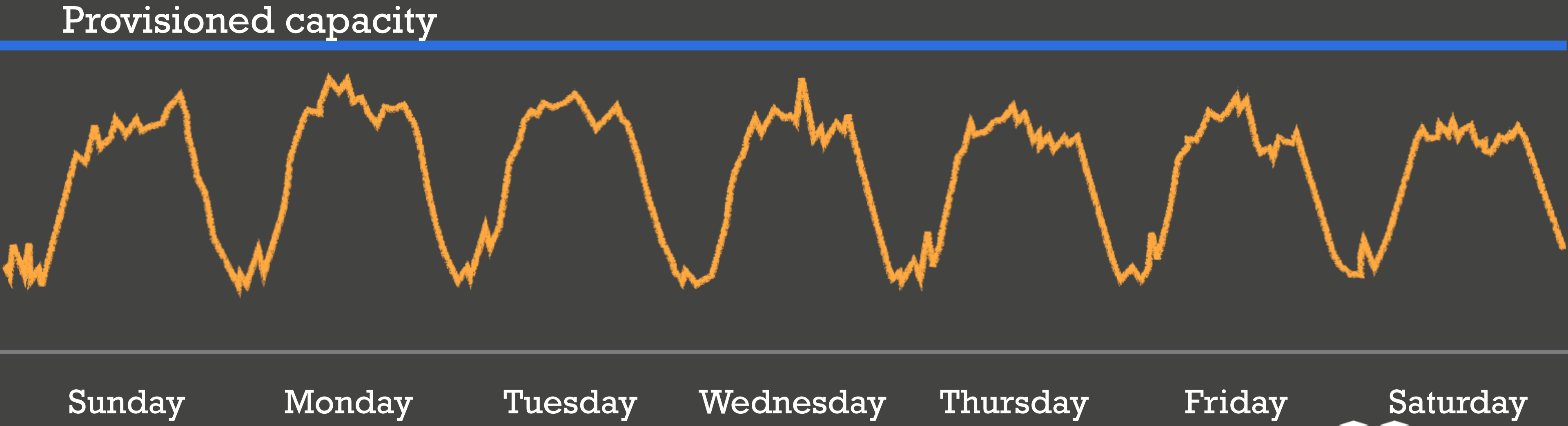
Feature	Details
Control	Define minimum and maximum instance pool sizes and when scaling and cool down occurs.
Integrated to Amazon CloudWatch	Use metrics gathered by CloudWatch to drive scaling.
Instance types	Run Auto Scaling for On-Demand and Spot Instances. Compatible with VPC.

```
as-create-auto-scaling-group MyGroup
     --launch-configuration MyConfig
     --availability-zones us-east-1a
     --min-size 4
     --max-size 200
```

Typical weekly traffic to Amazon.com

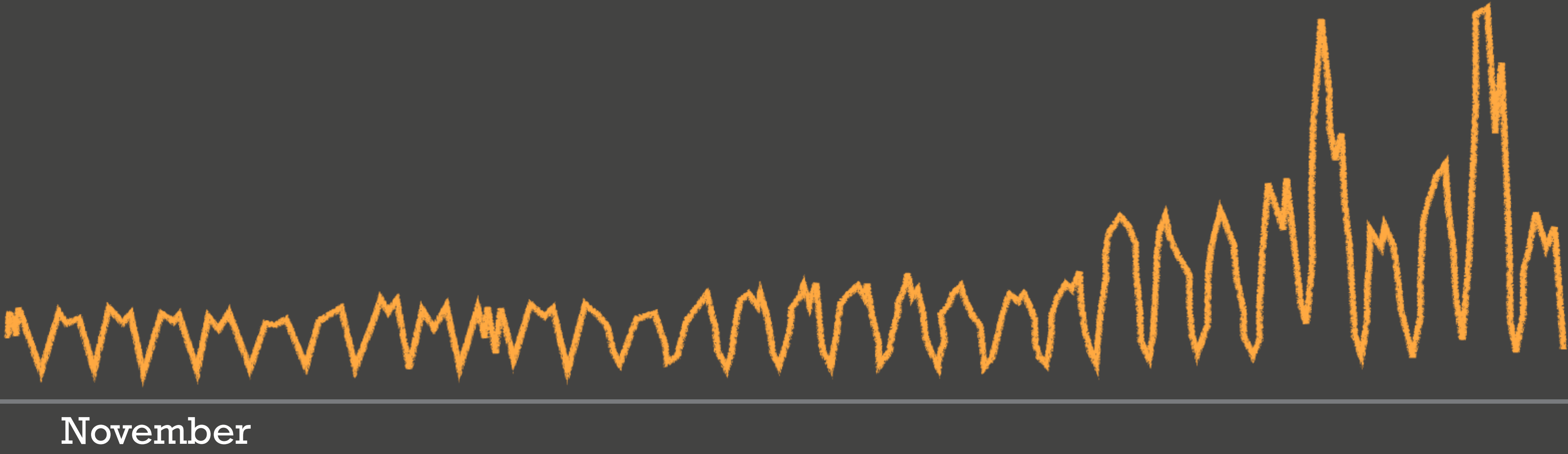


Typical weekly traffic to Amazon.com



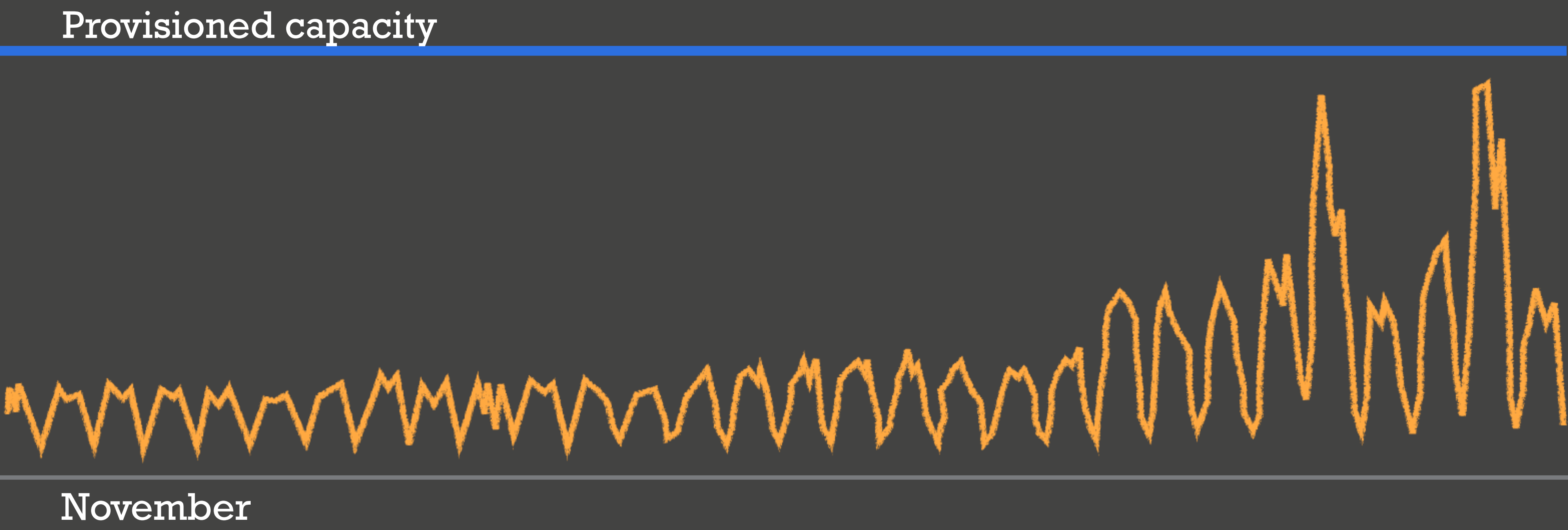
amazon web services

November traffic to Amazon.com

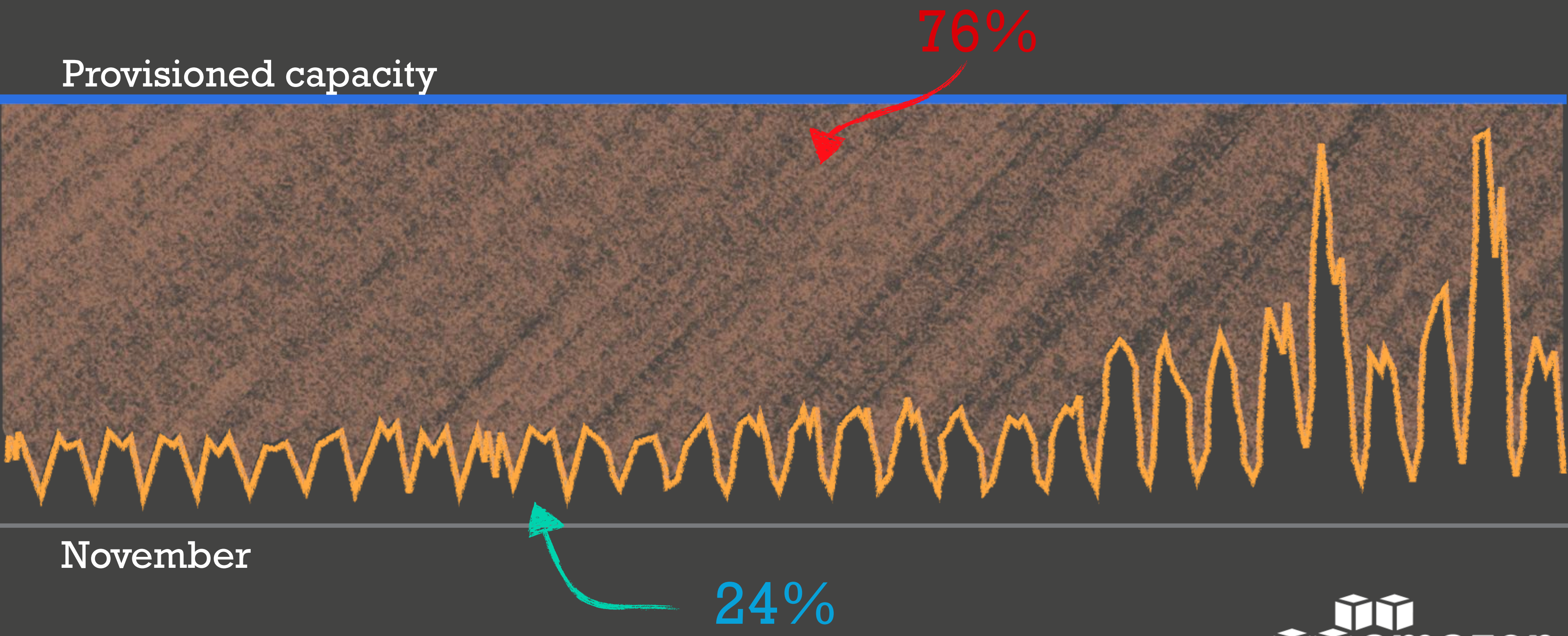


amazon web services

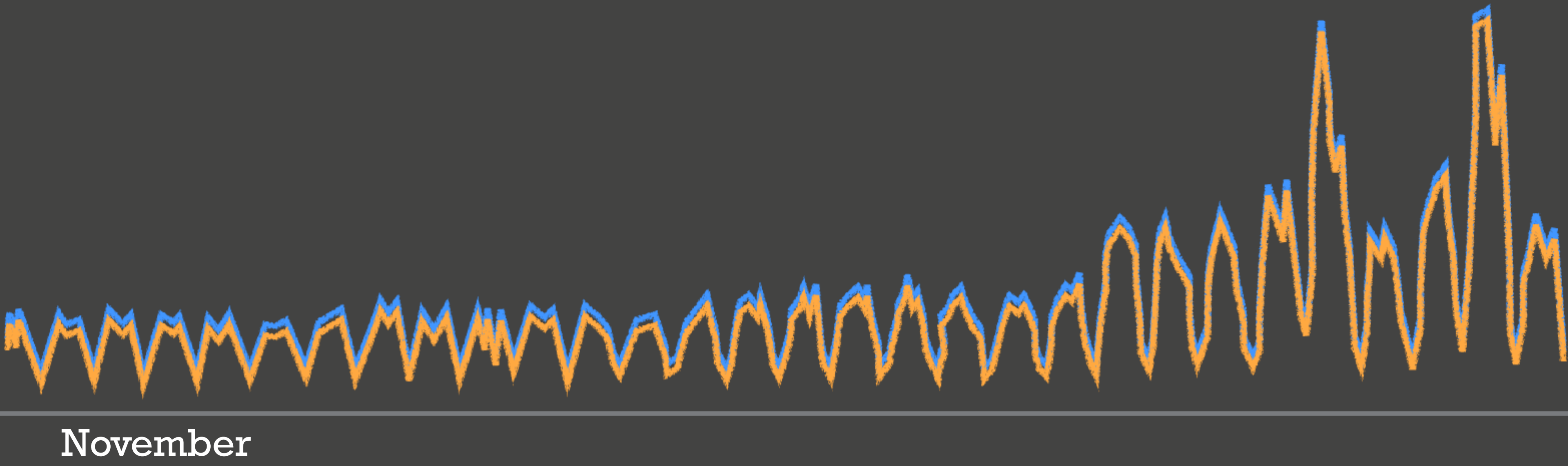
November traffic to Amazon.com



November traffic to Amazon.com



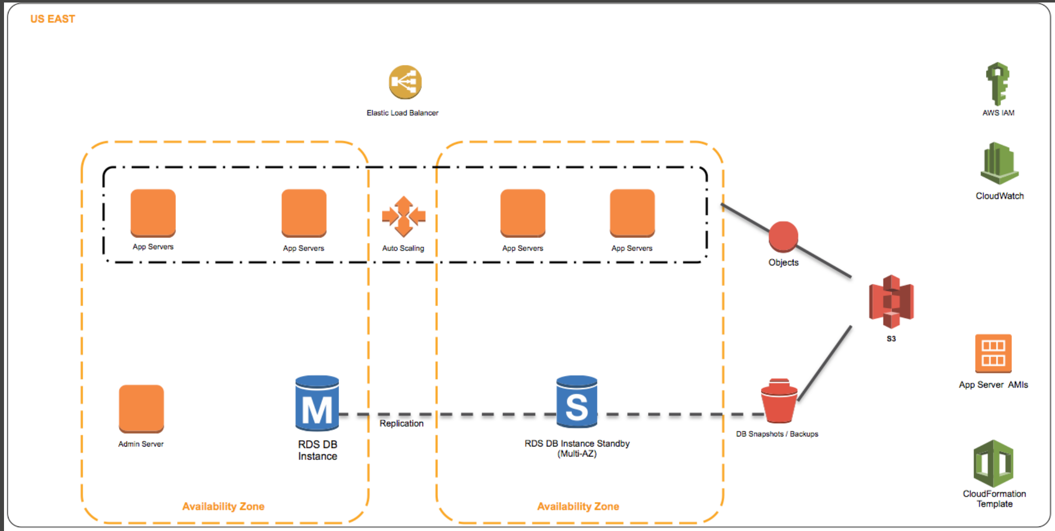
November traffic to Amazon.com



Auto-Scaling lets you do this!

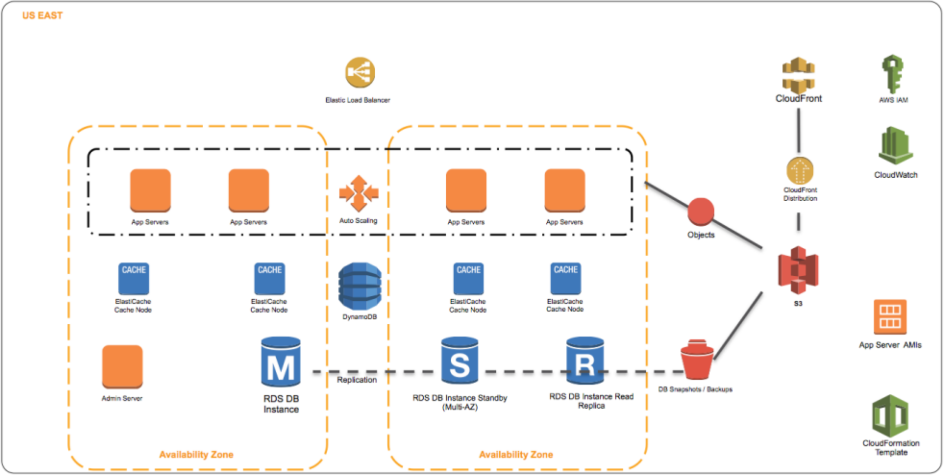
Lab 5 Review: Auto Scaling

- Approach to Auto Scaling Policies
- Approaches to Load Simulation

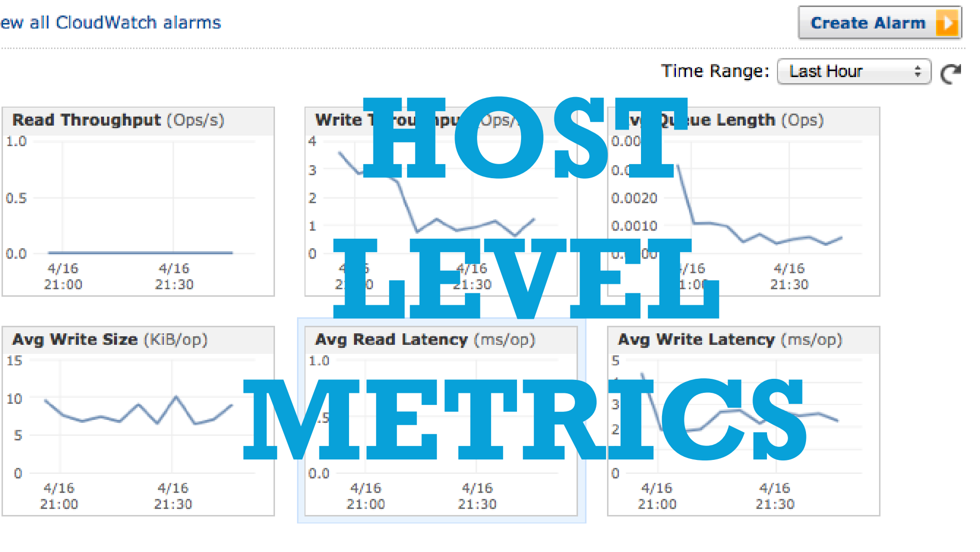


Option Lab 7: Caching

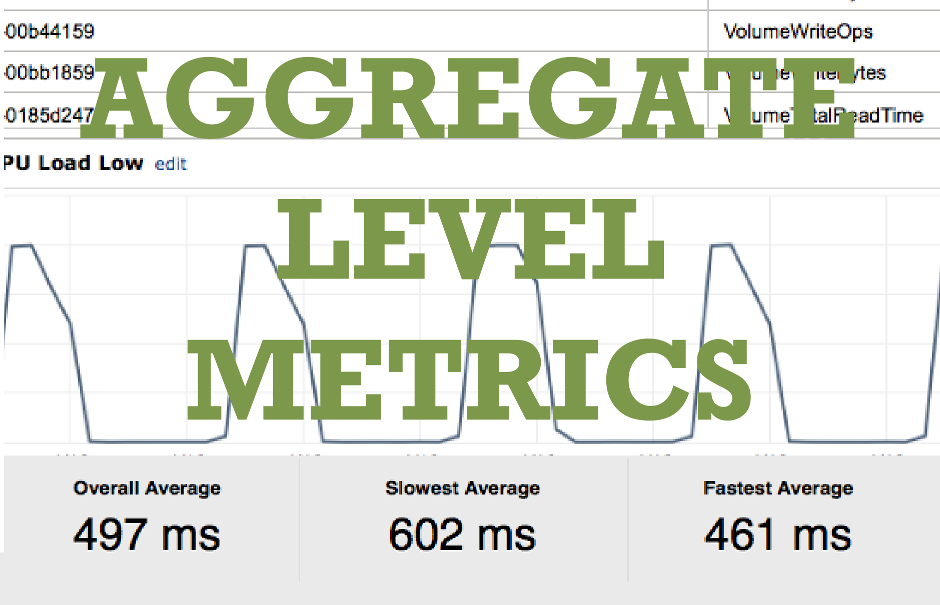
- static content CloudFront consideration
- MemCached and Elastic Cache
- DynamoDB implementation



Not having proper monitoring/metrics is like flying a plane with an eye mask on in a thunderstorm. Oh and your wing is on fire.

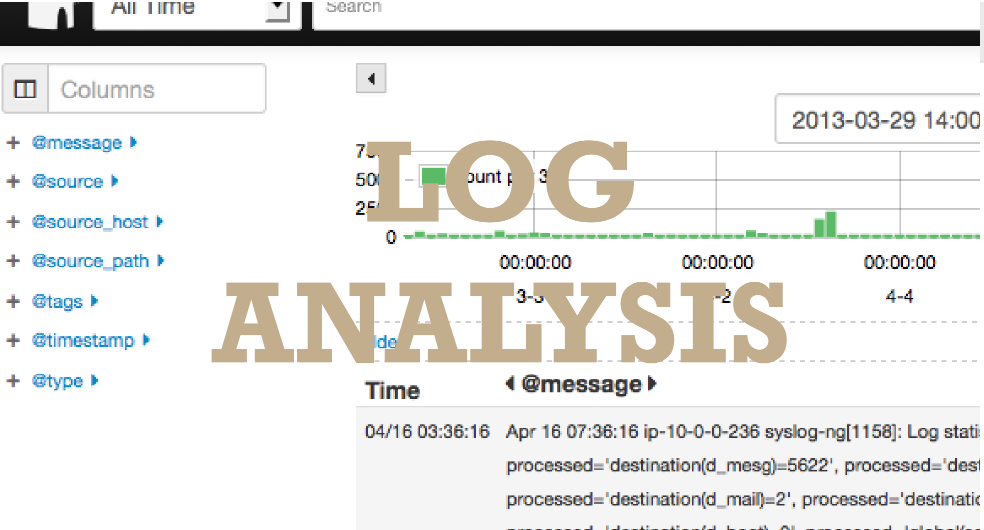


HOST LEVEL METRICS



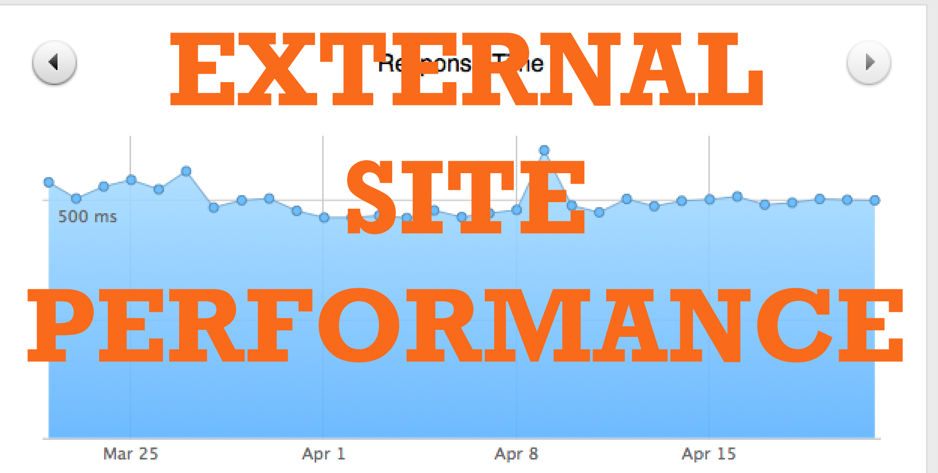
AGGREGATE LEVEL METRICS

Overall Average	Slowest Average	Fastest Average
497 ms	602 ms	461 ms



LOG ANALYSIS

```
Time            @message
04/16 03:36:16  Apr 16 07:36:16 ip-10-0-0-236 syslog-ng[1158]: Log stati
                processed='destination(d_mesg)=5622', processed='dest
                processed='destination(d_mail)=2', processed='destinatic
```



EXTERNAL SITE PERFORMANCE

AWS Marketplace & Partners Can Help

- Customer can find, research, buy software
- Simple pricing, aligns with EC2 usage model
- Launch in minutes
- Marketplace billing integrated into your AWS account
- 700+ products across 20+ categories

boundary

sumologic

loggly

Nagios

librato.

DATADOG

metrics

AppFirst

Learn more at: aws.amazon.com/marketplace

amazon web services

Option Lab 8: Monitoring

- Custom CloudWatch Metrics
- Third party Monitoring

Next steps?

READ! –

- aws.amazon.com/documentation
- aws.amazon.com/architecture
- aws.amazon.com/start-ups

Next steps?

START USING AWS –

aws.amazon.com/free/

Next steps?

ASK FOR HELP!

- forums.aws.amazon.com
- aws.amazon.com/support
- Your local account manager

THANKS F0R LISTENING!

- jman@amazon.com

amazon
web services