CS 188

Scalable Internet Services

John Rothfels

October 15, 2019

Today's Agenda

Motivation

High Availability

- HA datacenter design
- HA on AWS

Deploying on AWS

For Next Time

Motivation

As we've discussed, more and more of our daily lives depend on software.

If software is indeed eating the world, what happens when software systems fail?

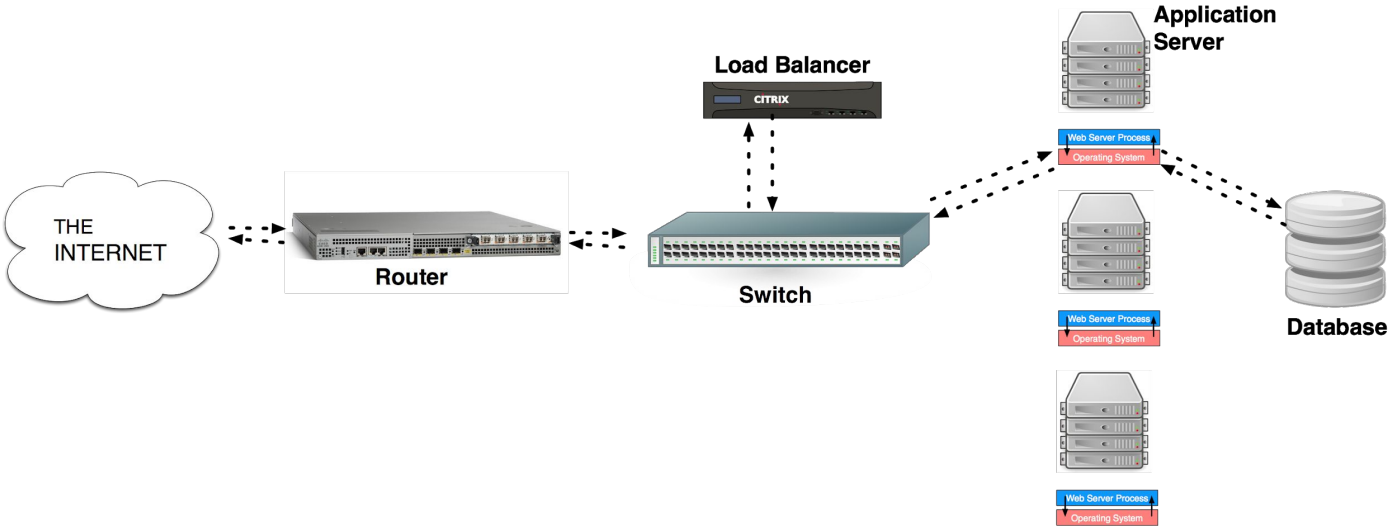
Motivation

Modern web applications power some very important parts of our lives

- Banking, Medical, Telephony (increasingly), etc.
- High availability is increasingly important.
- A common phrase targeted by businesses is “X nines”
  - Three nines = 99.9% uptime =~ 45 minutes a month down
  - Four nines = 99.99% uptime =~ 5 minutes a month down
    - Business applications
  - Five nines = 99.999% uptime =~ five minutes a year down
    - Communications companies

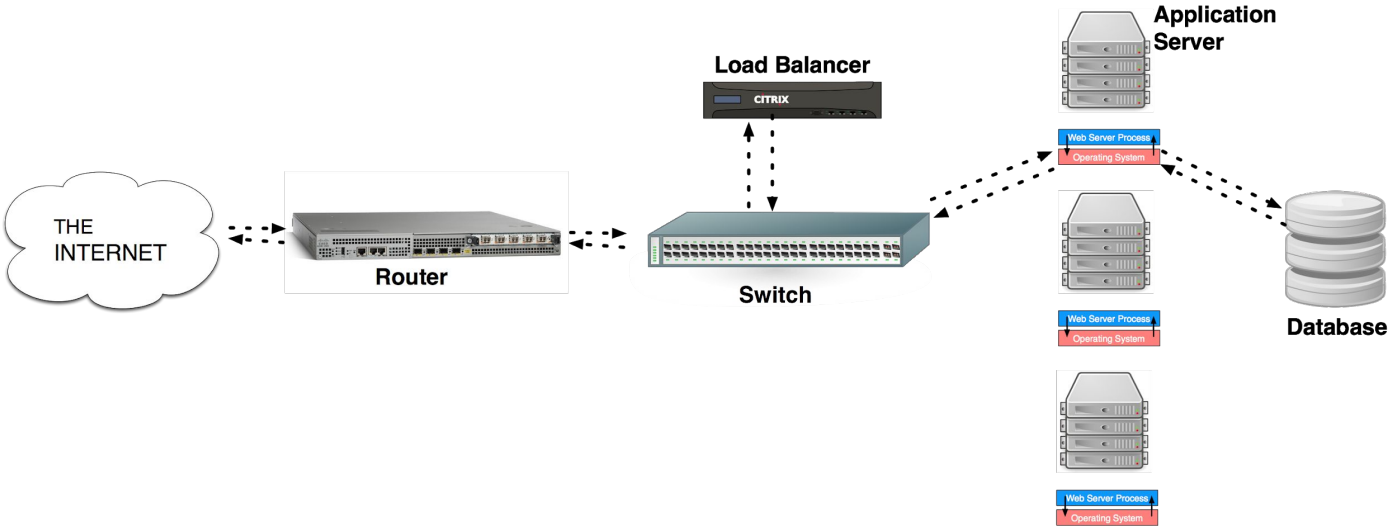
High Availability

What are possible causes of failures?



High Availability

What are possible causes of failures?



- Server process dies?
- Application server fails?
- Load balancer fails?
- Switch fails?
- Internet fails?
- Database fails?
- Entire datacenter fails?

High Availability

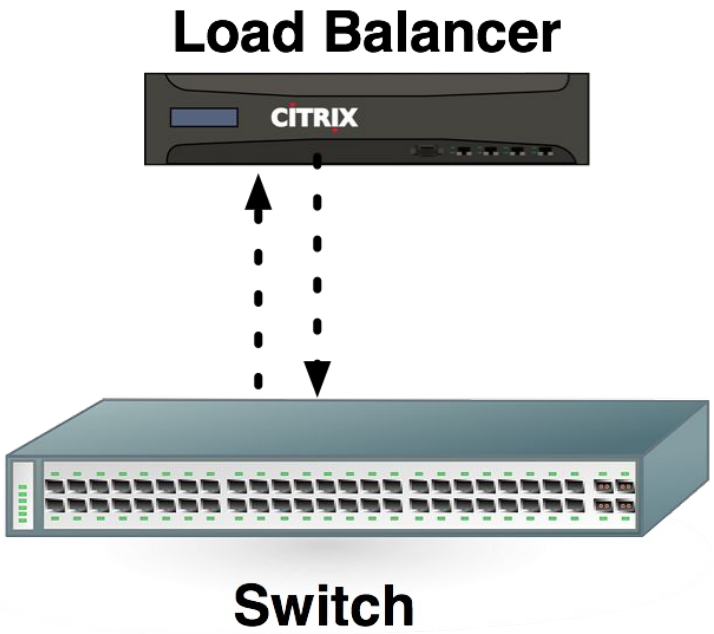
Application Server Fails?

High Availability

Application Server Fails?

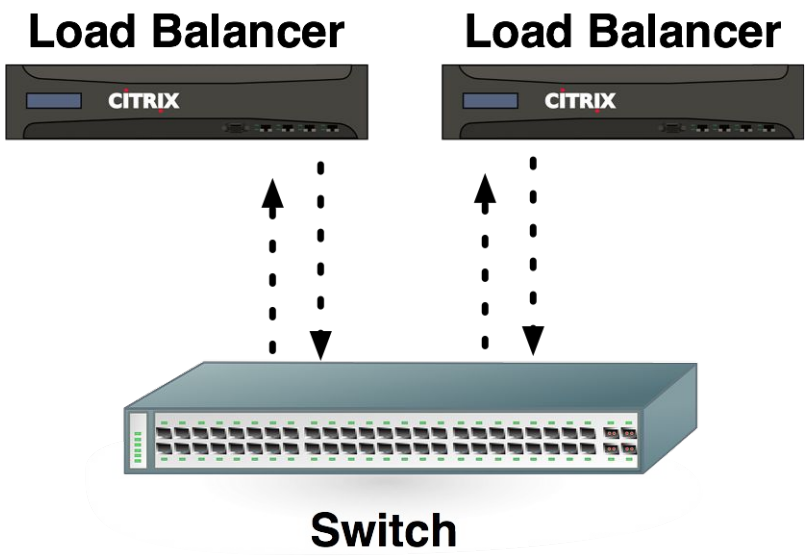
- We've already seen a lot here
- Having process-level isolation reduces disruptions to a single process failure
- A load-balanced configuration means any single app server can go down and we can direct load elsewhere

High Availability



Load balancer fails?

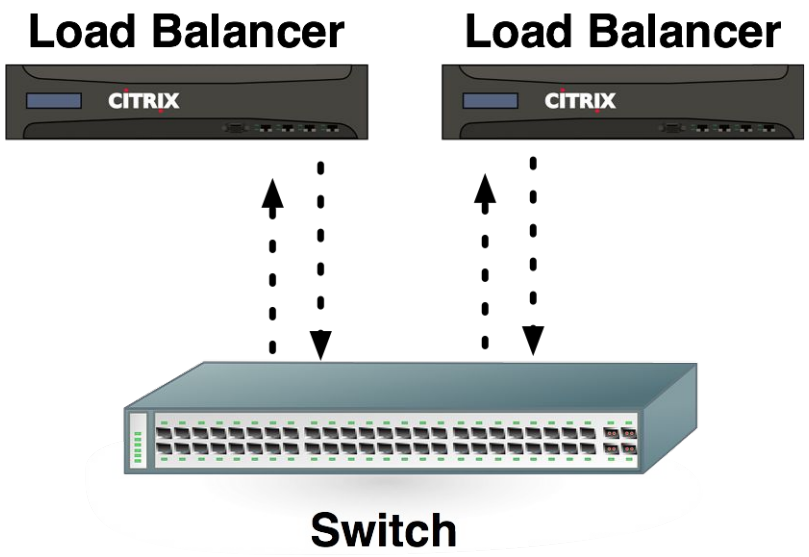
High Availability



Load balancer fails?

- Lets buy two: primary & failover
- **How do we detect when failure has occurred?**

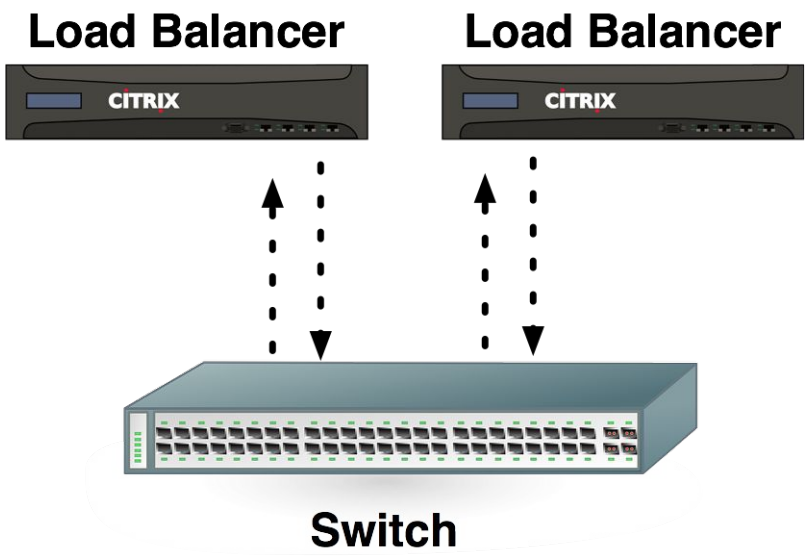
High Availability



Load balancer fails?

- Load balancers use heartbeats to determine health
- During failover, what happens to IP addresses?

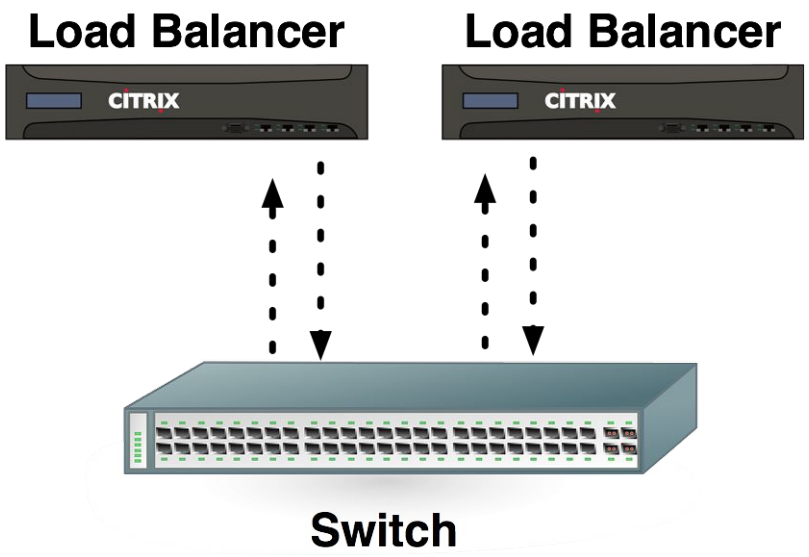
High Availability



Load balancer fails?

- IP address
  - When secondary determines it needs to step in, it issues a Gratuitous ARP
  - Other devices on the network that were communicating with the primary cleanly switch over to secondary

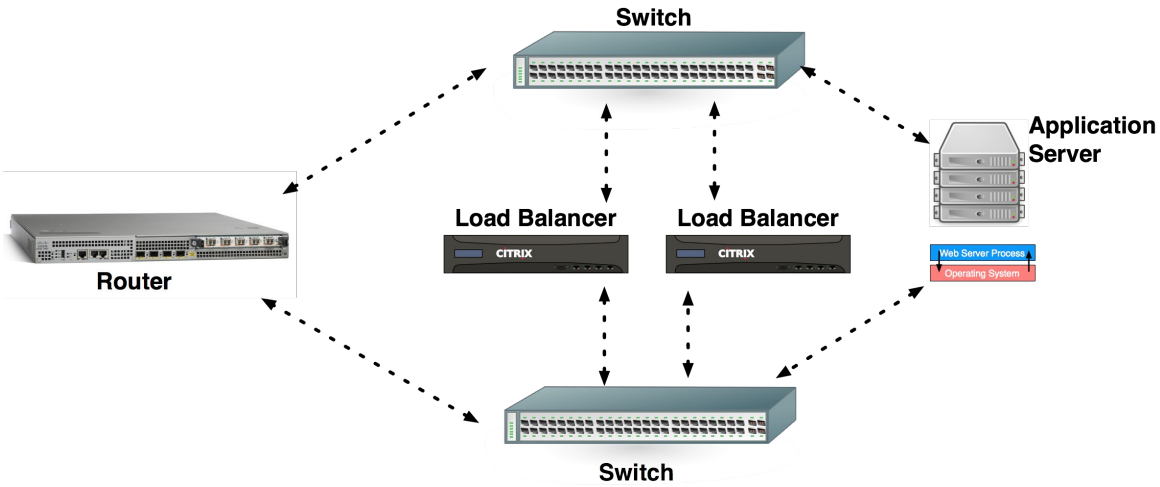
High Availability



Switch fails?

High Availability

Switch fails?

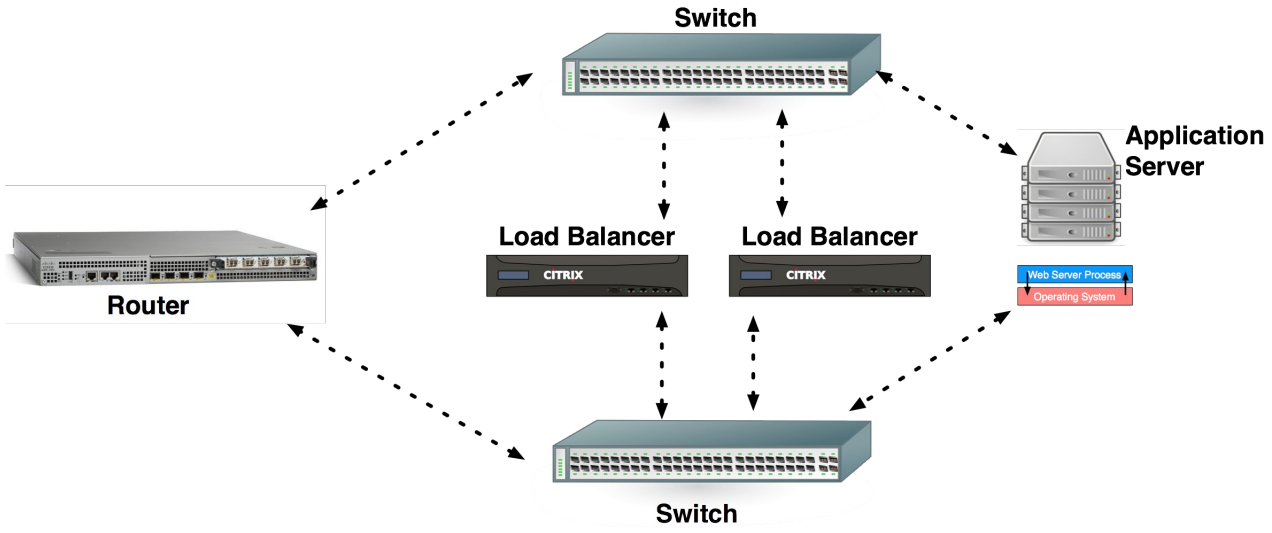


Lets buy two!

- Link aggregation allows multiple interfaces to share a MAC address
- MC-LAG, is Multi-Chassis Link Aggregation
- Link Aggregation Control Protocol handles failure detection
- Failure is simple: no sessions to maintain.

High Availability

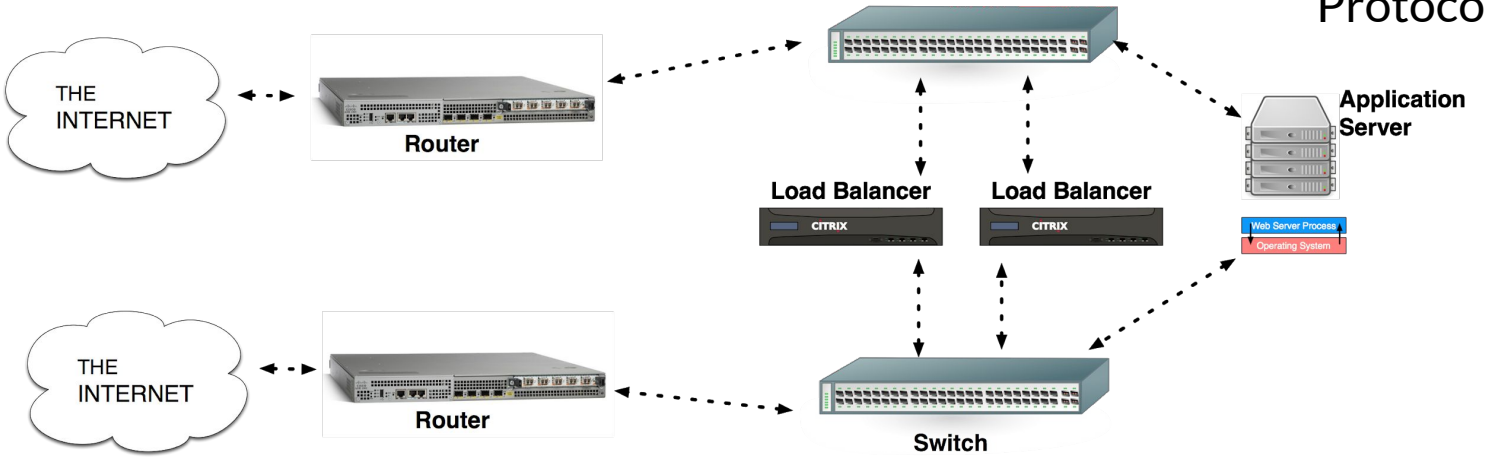
Router fails?



High Availability

Router fails?

- Lets buy two!
- Similar heartbeat and failover system (Hot Standby Routing Protocol)



High Availability

Internet fails?

High Availability

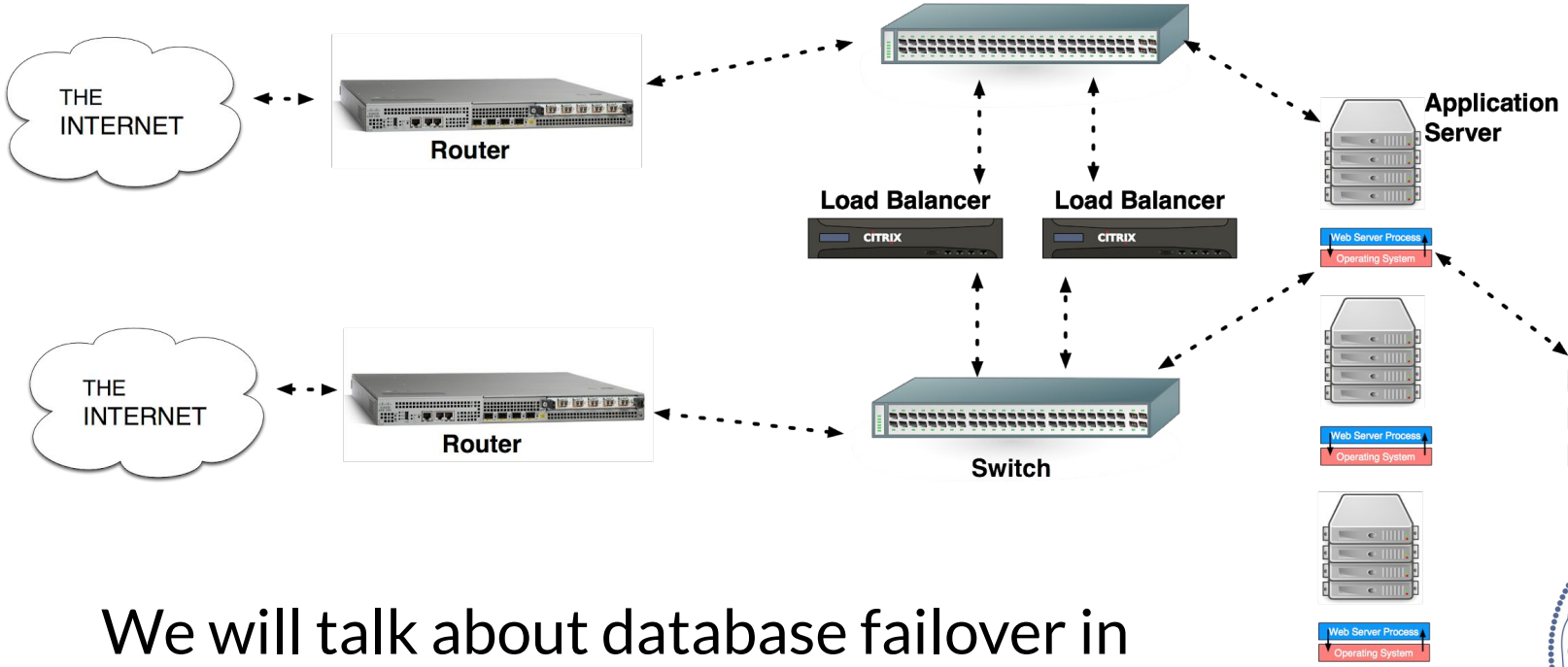
Internet fails?

- Lets have two ISPs.
  - One link to, say, Sprint and another to Internap.

How do we handle routing when we have two ISPs?

- Outgoing traffic is easy, since we control these decisions.
  - Pick the cheapest or most reliable link
  - Pick the “closer” link
- Incoming traffic is hard
  - We can’t directly tell clients how to reach our web app

High Availability



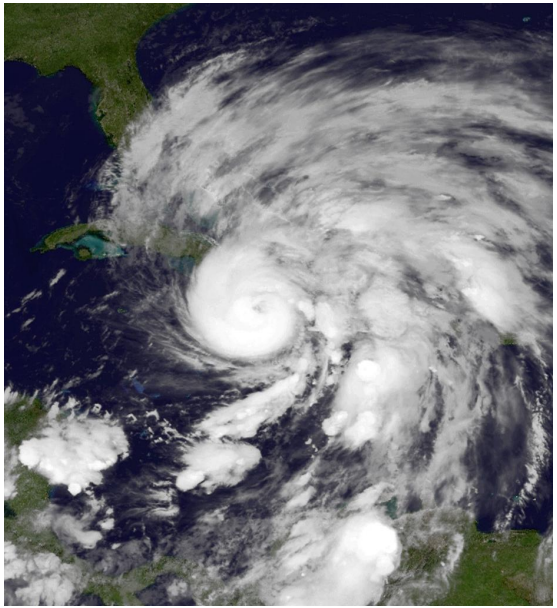
We will talk about database failover in future lectures

High Availability

Ok so we're good right? Nothing can go wrong?

High Availability

Ok so we're good right? Nothing can go wrong?



High Availability

Hurricane Sandy takes data centers offline with flooding, power outages

Hosting customers stranded as generators in NY data centers run out of fuel.

by **Jon Brodkin** - Oct 30 2012, 9:25am PDT

Share Tweet 100

High Availability

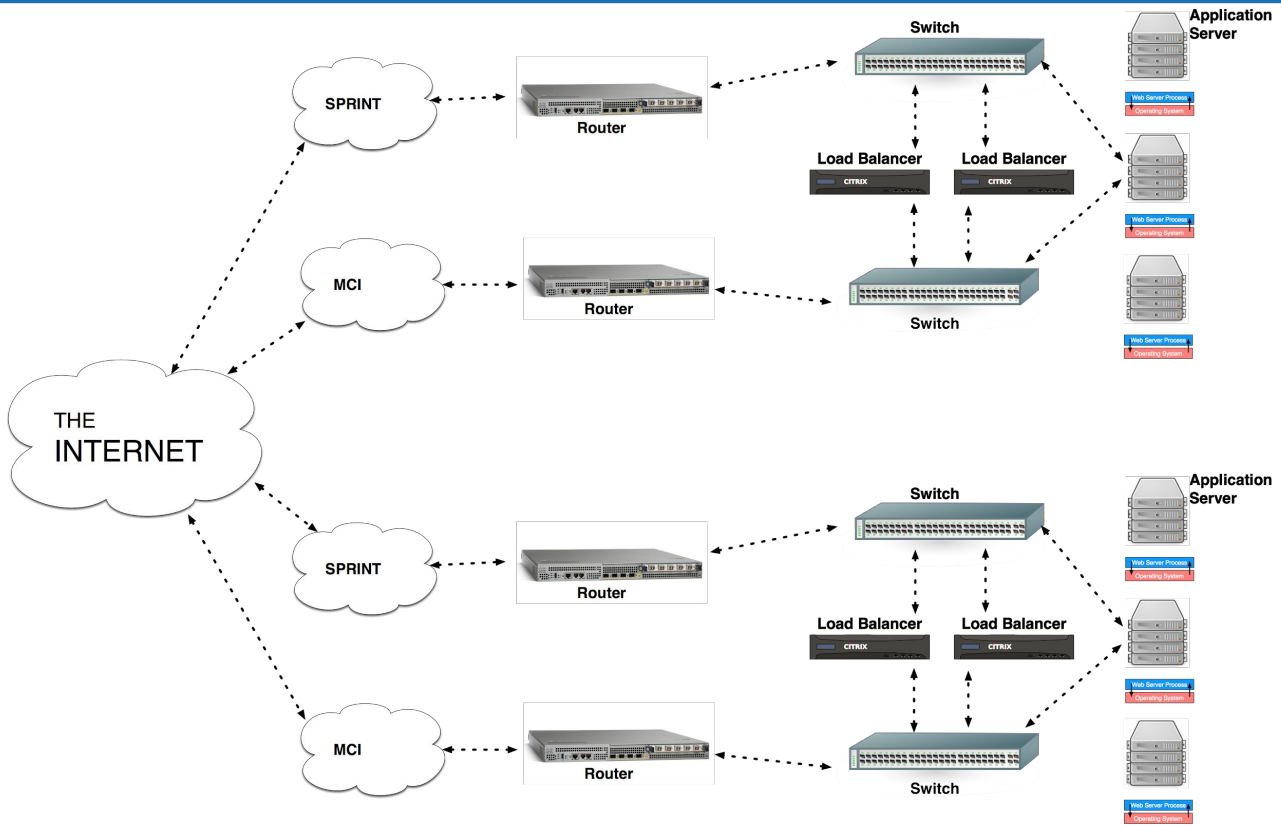
Availability Axiom (Pete Tenereillo):

- The only way to achieve high-availability for browser based clients is the include the use of multiple A-records (DNS).

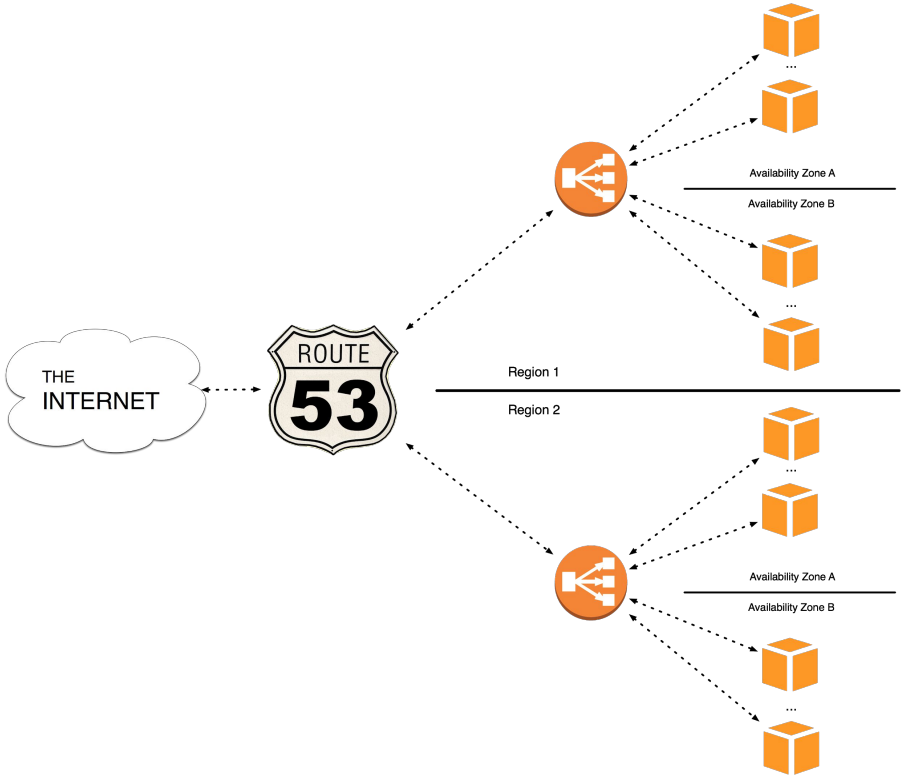
Result:

- For performance, we want to send the browser to one datacenter.
- For availability, we want to send the browser multiple A records.
- We end up having to make a choice between performance and availability.

High Availability



High Availability on AWS



AWS has regions and availability zones

- **Region**: think a city
- **Availability zone**: think a data center

Failures between availability zones are not correlated*.

For your projects

The AWS tooling we provide you emphasizes testing scaling over High Availability.

Advanced students can still do HA, but it will not be covered in lecture or labs.

AWS Instructions

Now that you've all got a blank Rails app (or more) pushed to Github, it's time to learn how to deploy to AWS.

We will be using Elastic Beanstalk, and here are step-by-step instructions for deploying.

But first, some warnings and rules.

AWS Instructions

These are scarce resources

- This is free time on Amazon's infrastructure, and it's not unlimited.
- Unless you have a specific reason to do otherwise, always use micro instances.
  - Example of a good reason: testing vertical scaling.

AWS Instructions

These are scarce resources

- Our AWS budget has a fixed limit.
- Whenever you are done with an instance, shut it down.
- Never keep important data on the instance, because it can go down at any time.
    - SCP important data back to your laptop.
- I will periodically run a script that terminates all instances that have been up longer than 8 hours.

AWS Instructions

Treat these credentials as secrets.

- Do not check into any publicly accessible repository
- There are automated scripts that actively seek out AWS credentials
  - Why?

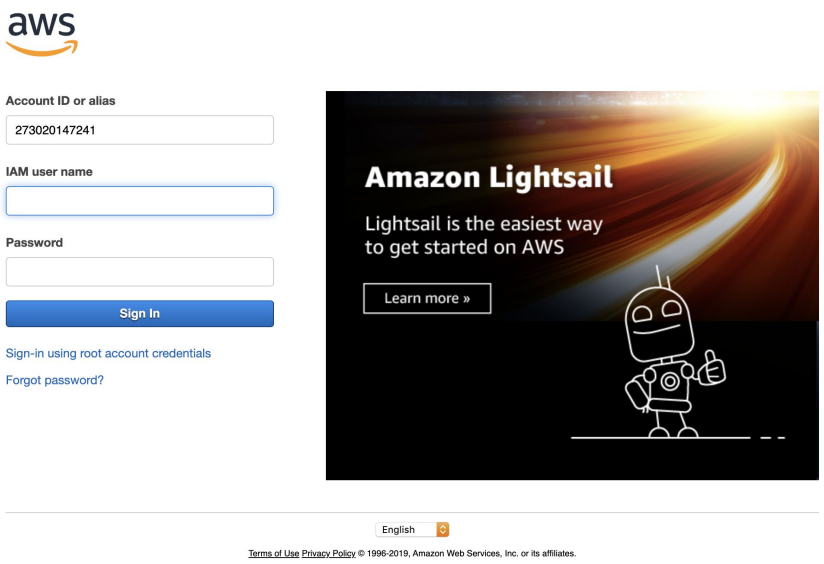
AWS Instructions

This week you will receive an email with AWS credentials. These include:

- A .txt file
  - Username: what you use to log in to the AWS web interface
  - Password: the password you use to log in to the AWS web interface
- A .pem file
  - Contains a private key that you will use to ssh into the instances that you launch.

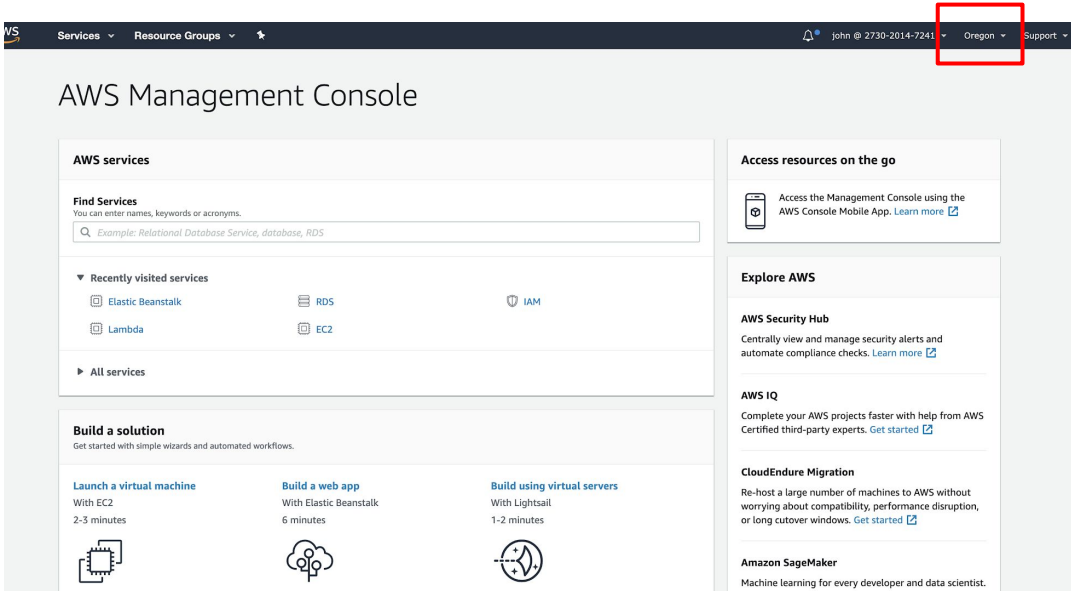
AWS Instructions

Go to https://273020147241.signin.aws.amazon.com/console



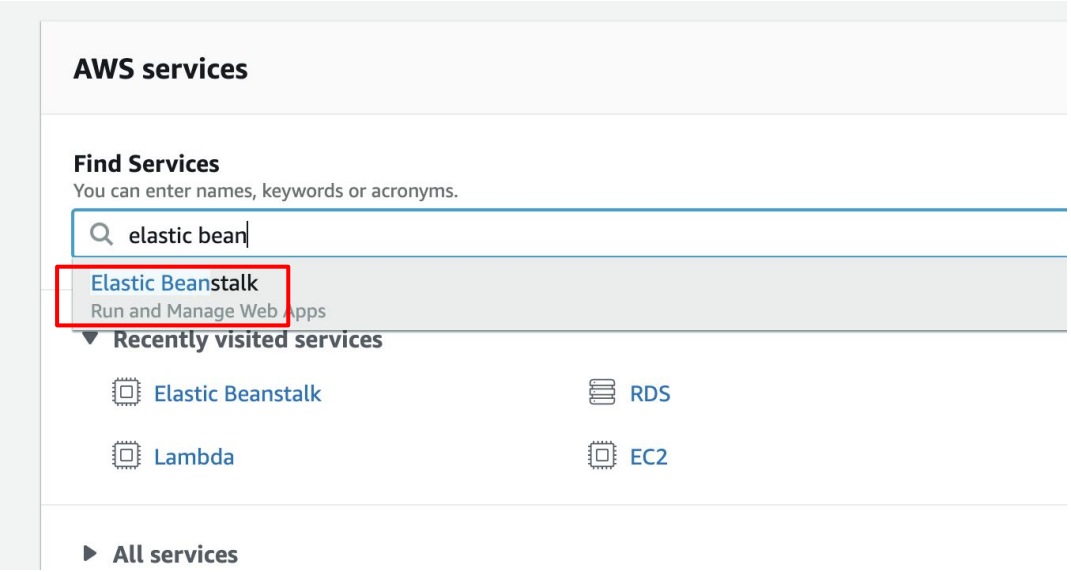
Login with Username and Password provided in txt file in email.

AWS Instructions



Make sure your region is set to “US-West-2 Oregon”

AWS Instructions



We will be deploying with Elastic Beanstalk, so select that that

AWS Instructions

```
ssh -i ~/.ssh/TEAMNAME.pem
TEAMNAME@ec2-52-35-41-146.us-west-2.compute.amazonaws.com

git clone https://github.com/scalableinternetservices/TEAMNAME.git

cat TEAMNAME_key.txt                    # for eb init

cd ./TEAMNAME

eb init                                 # details on next slide
```

AWS Instructions

`eb init`:

- Use the us-west-2 region (default).
- Provide your aws-access-id. (if prompted, output from `cat` operation)
- Provide your aws-secret-key. (if prompted, output from `cat` operation)
- Create a new application (default) for your team if no such application already exists.
- Use your team name as your application's name.
- Indicate that you are using ruby.
- For now, choose Ruby 2.5 (Puma) as your platform (default).
- Do not continue with CodeCommit (default).
- Indicate that you do want to set up SSH for your instances (default).
- Choose the keypair that matches your team's name

AWS Instructions

Elastic Beanstalk creates “environments” for running your application

- An environment can be one or more EC2 instances, a connected database, etc.
- Your application (team) can run multiple environments but should only need one at time

AWS Instructions

```
eb create -db.engine postgres -db.i db.t3.micro -db.user u -db.pass
password --envvars SECRET_KEY_BASE= any-secret-you-like --single
whatever-you-want-to-call-it
```

AWS Instructions

aws Services Resource Groups

Elastic Beanstalk | demo-john2 | demo-karl | demo-rishab

plications › demo-john2

nvironments

pplication versions

aved configurations

demo6

Environment tier: Web Server
Platform: Puma with Ruby 2.5 running on 64bit Amazon Linux/2.10.2
Running versions: app-1ef4-191007_023602
Last modified: 2019-10-06 19:48:07 UTC-0700
URL: demo6.difmv4gfpe.us-west-2.elasticbeanstalk.com
Health status: Ok

demo6-2

Environment tier: Web Server
Platform: Puma with Ruby 2.5 running on 64bit Amazon Linux/2.10.2
Running versions:
Last modified: 2019-10-14 22:00:26 UTC-0700
URL:
Health status: Pending

AWS Instructions

Creating demo6-2

This will take a few minutes..

10:01pm	Environment health has transitioned to Pending. Initialization in progress (running for 22 seconds). There are no instances.
10:00pm	Created EIP: 100.21.191.213
10:00pm	Creating RDS database named: aabl5io1riangr. This may take a few minutes.
10:00pm	Created security group named: awseb-e-xe6c5cmn5w-stack-AWSEBSecurityGroup-UYHX0YD1KYP9
10:00pm	Using elasticbeanstalk-us-west-2-273020147241 as Amazon S3 storage bucket for environment data.
10:00pm	createEnvironment is starting.

AWS Instructions

- Dashboard
- Configuration
- Logs
- Health
- Monitoring
- Alarms
- Managed Updates
- Events
- Tags

Events

Severity TRACE | 2019-08-26 22:14:00 UTC-0700

Time	Type	Details
2019-10-14 22:14:01 UTC-0700	INFO	Environment health has transitioned from Pending to Ok. Initialization completed 38 seconds ago and took 13 minutes.
2019-10-14 22:13:15 UTC-0700	INFO	Successfully launched environment: demo6-2
2019-10-14 22:13:15 UTC-0700	INFO	Application available at demo6-2.difmv4gfpe.us-west-2.elasticbeanstalk.com.
2019-10-14 22:11:01 UTC-0700	INFO	Added instance [i-0696063b90ab9aef7] to your environment.
2019-10-14 22:10:29 UTC-0700	INFO	Waiting for EC2 instances to launch. This may take a few minutes.
2019-10-14 22:09:25 UTC-0700	INFO	Created RDS database named: aabl5io1riangr
2019-10-14 22:01:02 UTC-0700	INFO	Environment health has transitioned to Pending. Initialization in progress (running for 22 seconds). There are no instances.
2019-10-14 22:00:57 UTC-0700	INFO	Created EIP: 100.21.191.213
2019-10-14 22:00:57 UTC-0700	INFO	Creating RDS database named: aabl5io1riangr. This may take a few minutes.
2019-10-14 22:00:42 UTC-0700	INFO	Created security group named: awseb-e-xe6c5cmn5w-stack-AWSEBSecurityGroup-UYHX0YD1KYP9
2019-10-14 22:00:22 UTC-0700	INFO	Using elasticbeanstalk-us-west-2-273020147241 as Amazon S3 storage bucket for environment data.

AWS Instructions

- Dashboard
- Configuration
- Logs
- Health
- Monitoring
- Alarms
- Managed Updates
- Events
- Tags

Configuration overview

Cancel | Review changes | Appl

Grid Vie

Search for an option name or value

Category	Options
Software	Environment properties: BUNDLE_WITHOUT, RACK_ENV, RAILS_SKIP_ASSET_COMPILATION, RAILS_SKIP_MIGRATIONS, SECRET_KEY_BASE Rotate logs: disabled Log streaming: disabled X-Ray daemon: disabled
Instances	AMI ID: ami-0db2491371ecd1d75 Instance type: t2.micro Monitoring interval: 5 minute IOPS: container default Size: container default Root volume type: container default EC2 security groups: awseb-e-xe6c5cmn5w-stack-AWSEBSecurityGroup-UYHX0YD1KYP9
Capacity	Scaling cooldown: 360 seconds Environment type: single instance Time-based Scaling:
Load balancer	*This configuration does not contain a load balancer.*
	Command timeout: 600 Deployment policy: All at once

AWS Instructions

How do I SSH into my instance?

- ssh -i *[your pemfile here]* ec2-user@ec2-something.us-west-2.compute.amazonaws.com

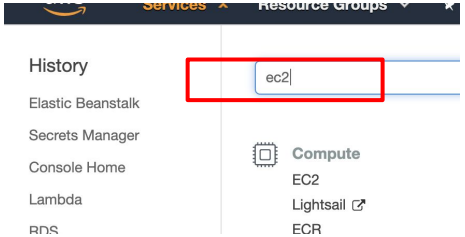
How do I copy files to/from my instance?

- scp -i *[your pemfile here]* ec2-user@ec2-something.us-west-2.compute.amazonaws.com:fromfile tofile

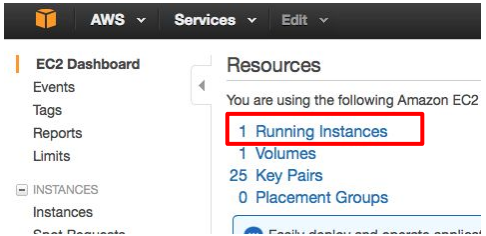
AWS Instructions

My Elastic Beanstalk stack failed to deploy. How do I debug this?

1. Go to the AWS dashboard and select EC2



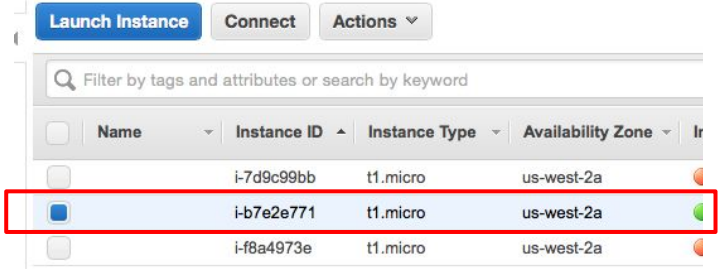
2. Click on “X Running Instances”



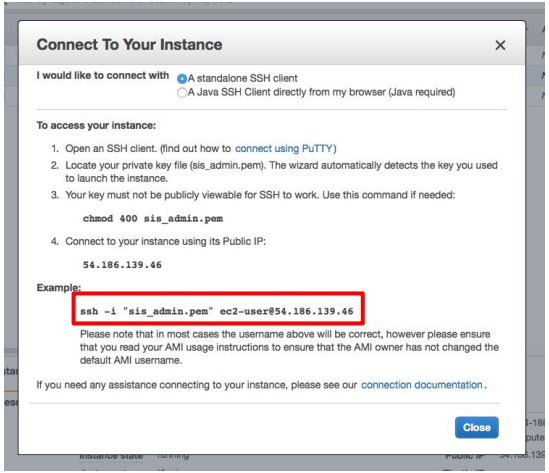
AWS Instructions

My Elastic Beanstalk environment failed to deploy. How do I debug this?

3. Select the instance corresponding to your team



4. Click the connect button and copy and paste the ssh command.



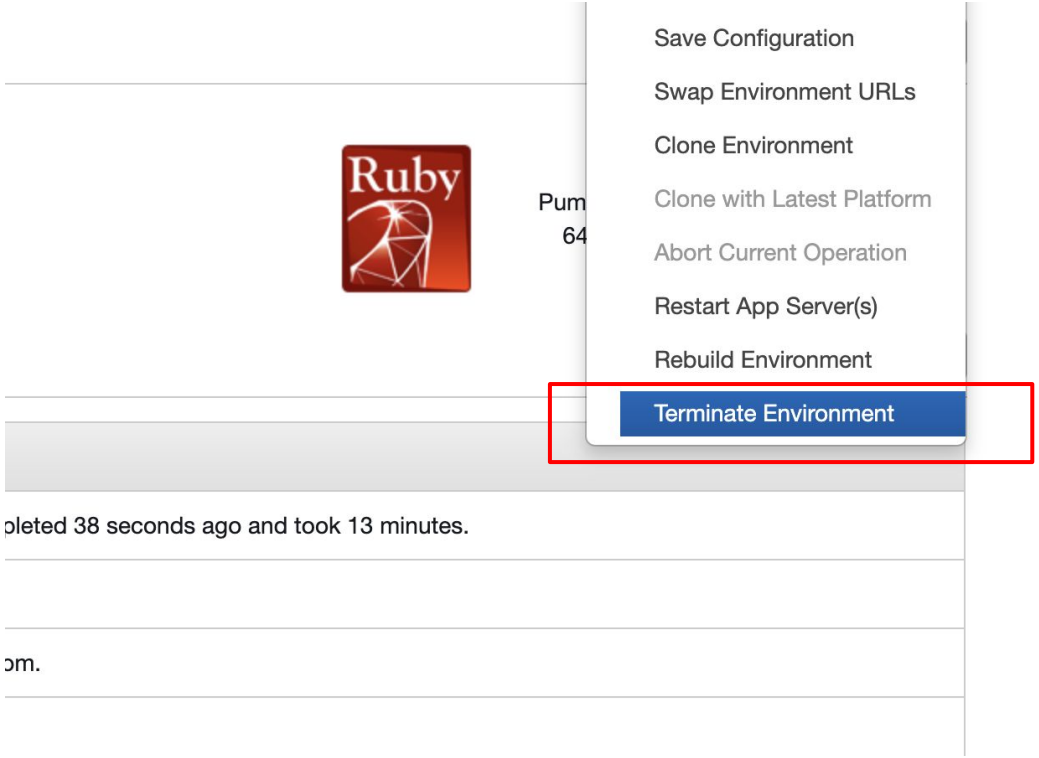
AWS Instructions

My cloudformation stack failed to deploy. How do I debug this?

Once you have SSHed into your instance, you can find details of problems at:

```
/var/log/eb-cfn-init.log
```

AWS Instructions



When you are done with your stack, remember to delete it!

If you have any questions, please post to Piazza!

Motivation

After today you should understand

High Availability

- HA datacenter design
- HA on AWS

Deploying on AWS w/ Elastic Beanstalk

For Next Time...

You should be working on your first sprint's worth of stories.

Please update your git repos with a README file. Please include:

- Project name
- Project description
- For each student your name and a photo