AWS

SUMMIT

# Building Apps in the Cloud to reduce costs up to 90%

Christian Petters, AWS Solutions Architect

18 May 2017

amazon web services

# AWS EC2 Consumption Models

## On-Demand

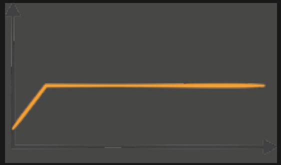
Pay for compute capacity by the hour with no long-term commitments

For spiky workloads, or to define needs

## Reserved

Make a low, one-time payment and receive a significant discount on the hourly charge

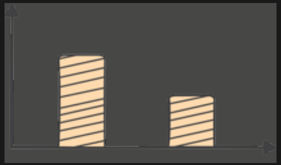
For committed utilization



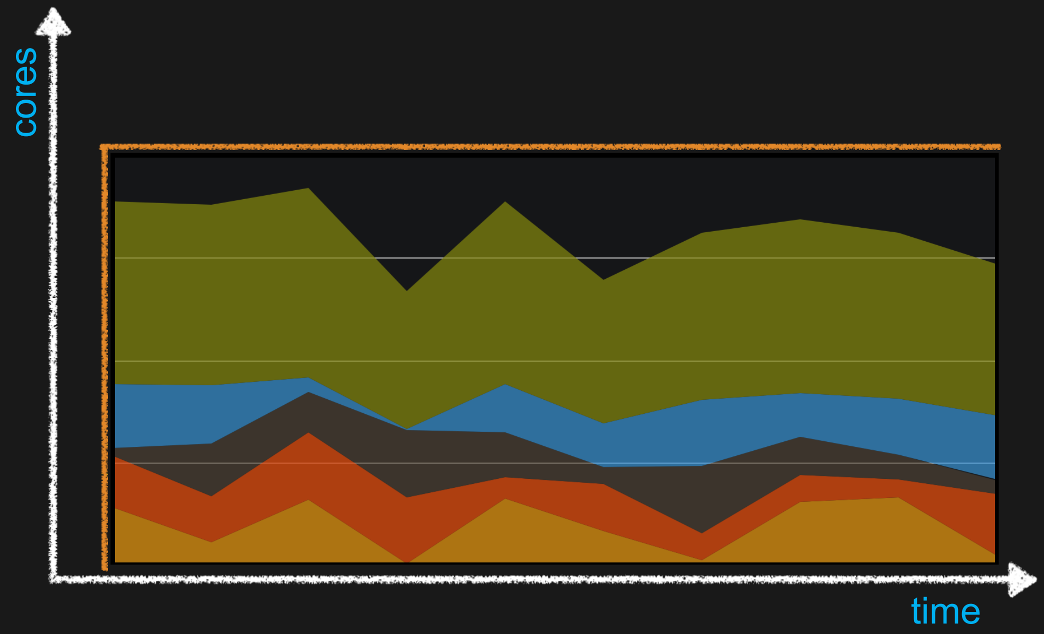
## Spot

Bid for unused capacity, charged at a Spot Price which fluctuates based on supply and demand

For time-insensitive, transient, or stateless workloads



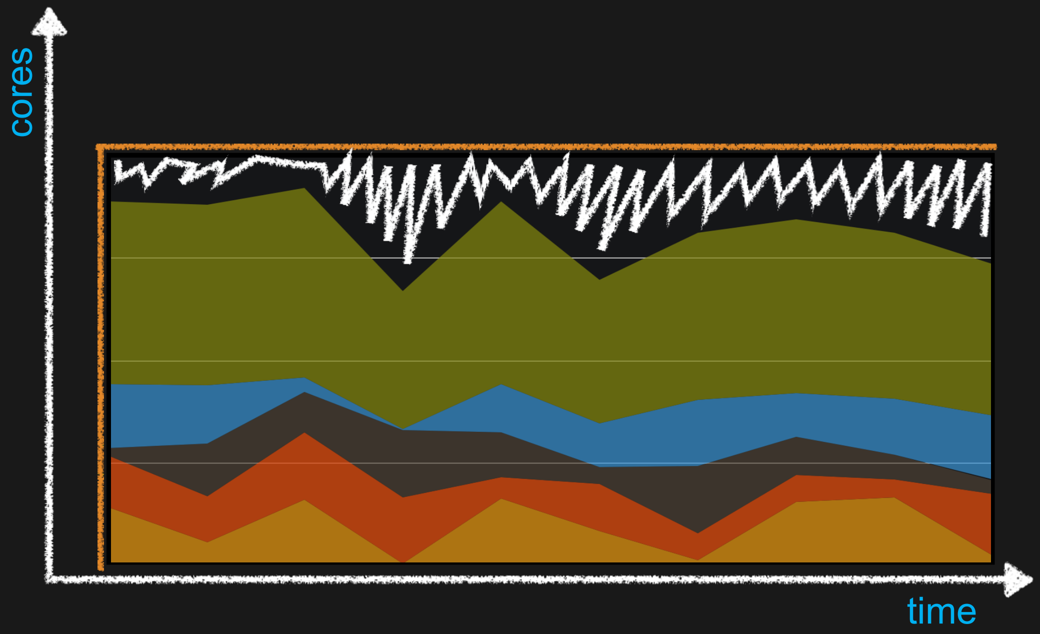
# Miiiiiiiiillions of uncorrelated workloads



## Collective action

When everyone comes together in the cloud to share the resource, and **only pays for what they use**, the efficiency is huge.

# Spot Market



**Spot Market**

Our ultimate space filler.

Spot Instances allow you to name your own price for spare AWS EC2 computing capacity.

Great for workloads that aren’t time sensitive (hint: it’s really cheap).

# The very simple rules of Spot

Markets where the price of compute changes based on supply and demand

You’ll never pay more than your bid. When the market exceeds your bid you get 2 minutes to wrap up your work

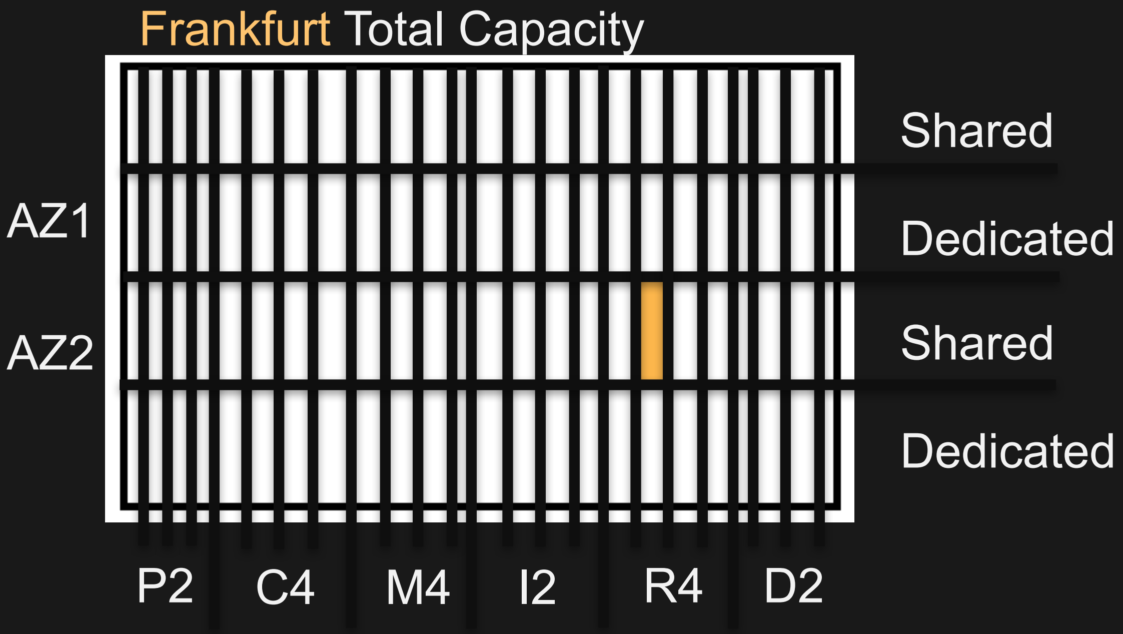
# Get the best value for EC2 capacity

- Since Spot instances typically cost 50-90% less than On-Demand, you can increase your compute capacity by 2-10x within the same budget

- Or you could save 50-90% on your existing workload

- Either way, you should try it!

# Understanding EC2 capacity



# Capacity and Spot Markets recap

| C4 | 2a | 2b | 2c | On Demand |
|---|---|---|---|---|
| 8XL | $0.50 | $0.27 | $0.29 | $1.76 |
| 4XL | $0.21 | $0.30 | $0.16 | $0.88 |
| 2XL | $0.08 | $0.07 | $0.08 | $0.44 |
| XL | $0.04 | $0.05 | $0.04 | $.22 |
| L | $0.01 | $0.01 | $0.04 | $0.11 |

Each instance family

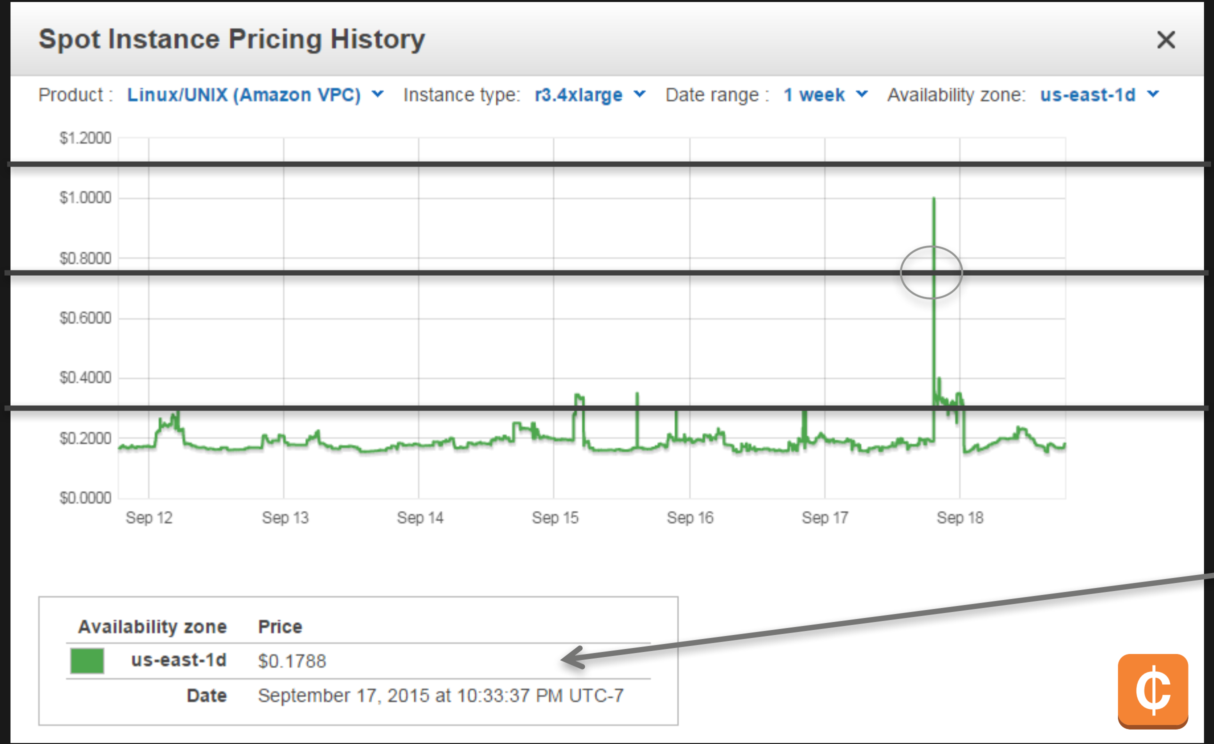
Each instance size

Each Availability Zone

In every region

Is a separate ***Spot Market***

# Bid Price Vs Market Price



Spot Instance Pricing History

Product : Linux/UNIX (Amazon VPC)   Instance type: r3.4xlarge   Date range : 1 week   Availability zone: us-east-1d

| Availability zone | Price |
|---|---|
| 🟩 us-east-1d | $0.1788 |
| Date | September 17, 2015 at 10:33:37 PM UTC-7 |

*75% Bid*

*50% Bid*

*25% Bid*

**You pay the market price**

# EC2 Spot best practices - Flexibility

Stateless

Fault tolerance

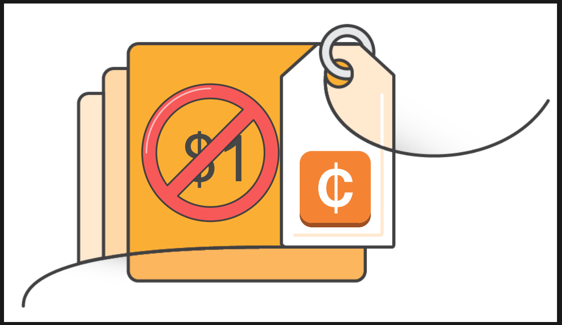
Multi-AZ

Loosely coupled

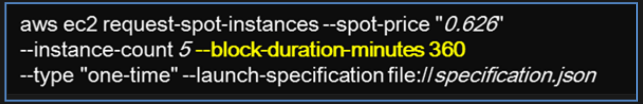
Instance Flexibility

# My instances cannot be interrupted!

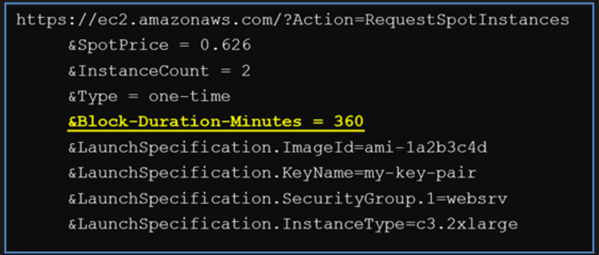
Run continuously
for up to 6 hours



Save up to 50% off
On-Demand pricing



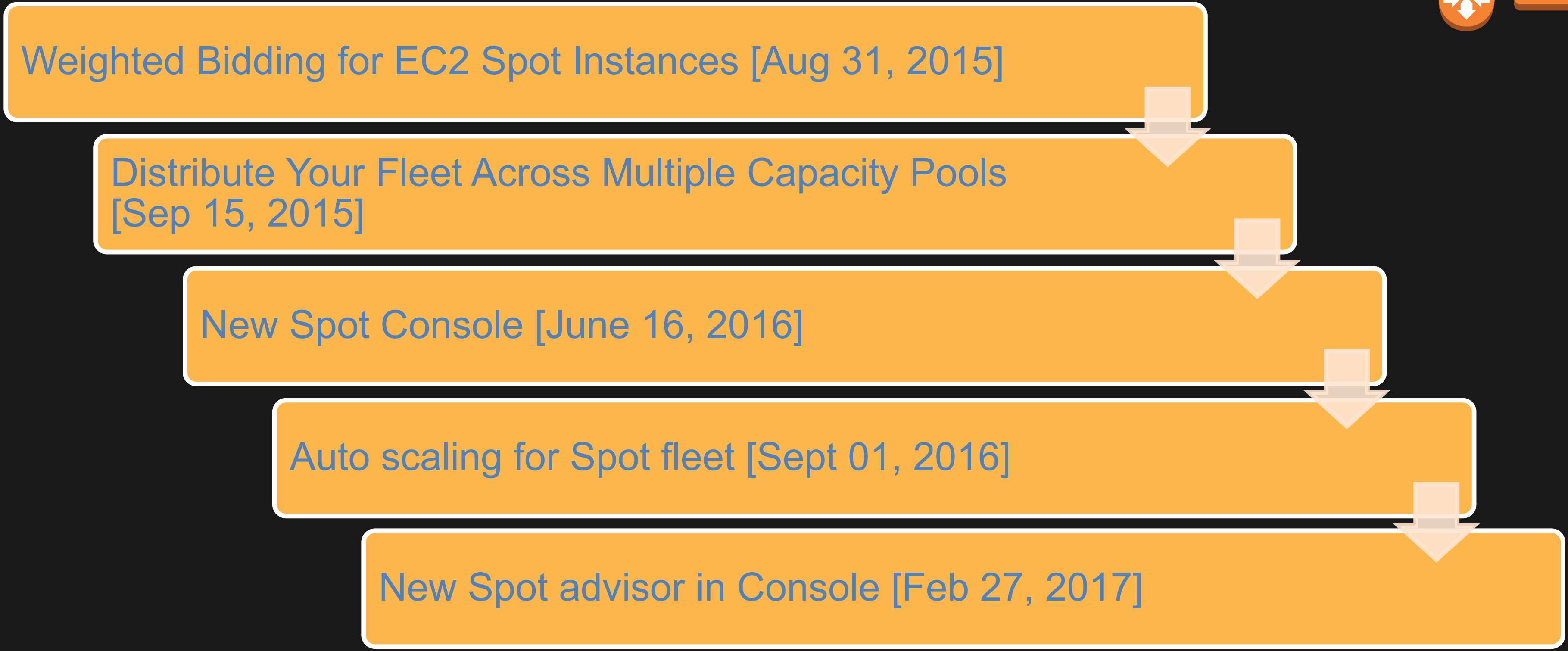
```
aws ec2 request-spot-instances --spot-price "0.626"
--instance-count 5 --block-duration-minutes 360
--type "one-time" --launch-specification file://specification.json
```



```
https://ec2.amazonaws.com/?Action=RequestSpotInstances
     &SpotPrice = 0.626
     &InstanceCount = 2
     &Type = one-time
     &Block-Duration-Minutes = 360
     &LaunchSpecification.ImageId=ami-1a2b3c4d
     &LaunchSpecification.KeyName=my-key-pair
     &LaunchSpecification.SecurityGroup.1=websrv
     &LaunchSpecification.InstanceType=c3.2xlarge
```

Using a single
additional parameter

# EC2 Spot fleet – significant features added



Weighted Bidding for EC2 Spot Instances [Aug 31, 2015]

Distribute Your Fleet Across Multiple Capacity Pools [Sep 15, 2015]

New Spot Console [June 16, 2016]

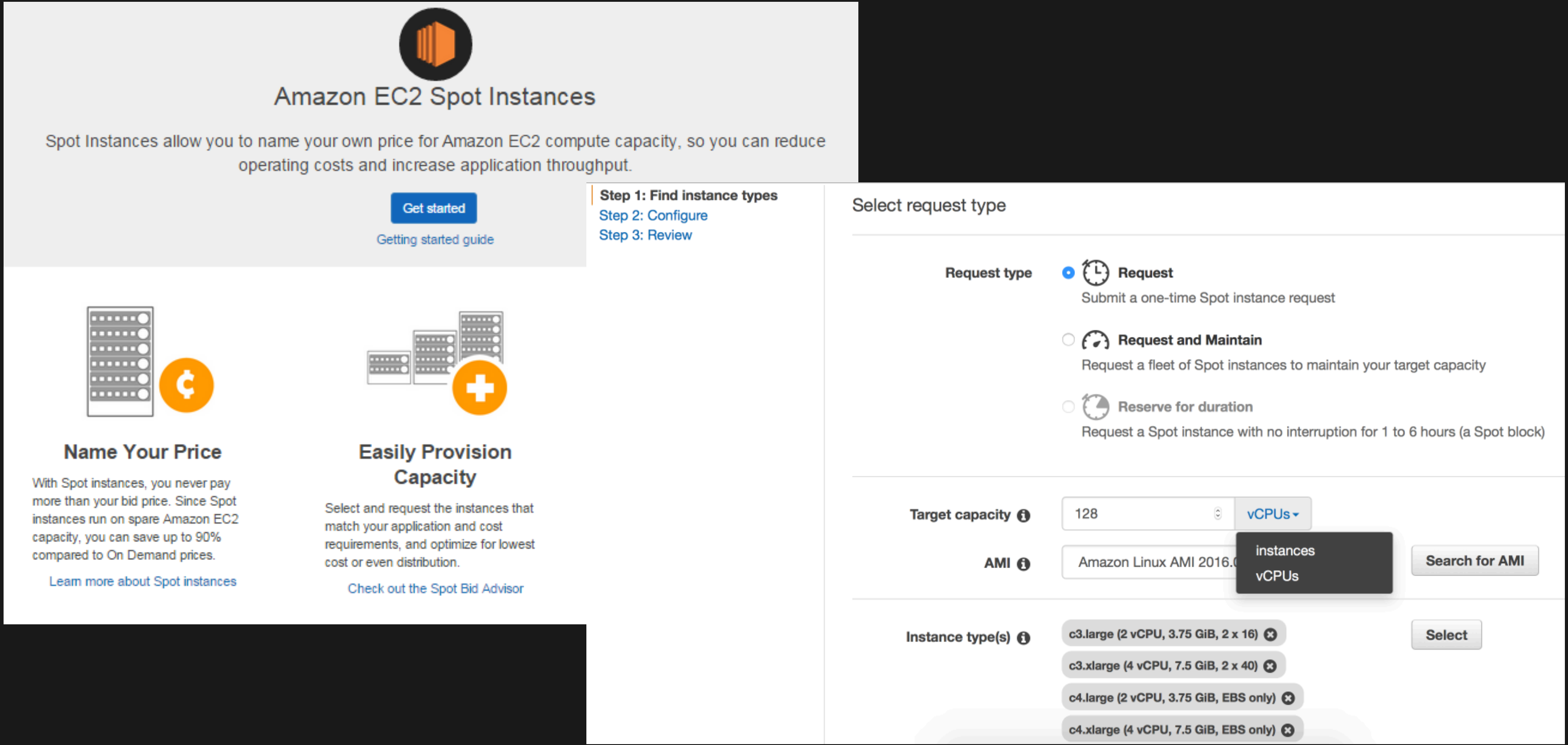
Auto scaling for Spot fleet [Sept 01, 2016]

New Spot advisor in Console [Feb 27, 2017]

# It is easy!

```
aws ec2 request-spot-fleet --spot-fleet-request-config file://config.json { "IamFleetRole":
"arn:aws:iam::781603563322:role/fleet-role", "TargetCapacity": "100", "SpotPrice": "0.03", "ValidFrom": "2015-
09-15T00:56:19Z", "ValidUntil": "2016-09-14T07:00:00Z", "TerminateInstancesWithExpiration": true,
"LaunchSpecifications": [ { "ImageId": "ami-0d4cfd66", "InstanceType": "c3.large", "WeightedCapacity": 2,
"SubnetId": "subnet-d0dc51fb" }, { "ImageId": "ami-0d4cfd66", "InstanceType": "c3.large", "WeightedCapacity":
2, "SubnetId": "subnet-64531413" }, { "ImageId": "ami-0d4cfd66", "InstanceType": "c3.large",
"WeightedCapacity": 2, "SubnetId": "subnet-0b1b8052" }, { "ImageId": "ami-0d4cfd66", "InstanceType":
"c3.xlarge", "WeightedCapacity": 4, "SubnetId": "subnet-d0dc51fb" }, { "ImageId": "ami-0d4cfd66",
"InstanceType": "c3.xlarge", "WeightedCapacity": 4, "SubnetId": "subnet-64531413" }, { "ImageId": "ami-
0d4cfd66", "InstanceType": "c3.xlarge", "WeightedCapacity": 4, "SubnetId": "subnet-0b1b8052" }, { "ImageId":
"ami-0d4cfd66", "InstanceType": "c3.4xlarge", "WeightedCapacity": 16, "SubnetId": "subnet-d0dc51fb" }, {
"ImageId": "ami-0d4cfd66", "InstanceType": "c3.4xlarge", "WeightedCapacity": 16, "SubnetId": "subnet-
64531413" }, { "ImageId": "ami-0d4cfd66", "InstanceType": "c3.4xlarge", "WeightedCapacity": 16, "SubnetId":
"subnet-0b1b8052" }, { "ImageId": "ami-0d4cfd66", "InstanceType": "c3.8xlarge", "WeightedCapacity": 32,
"SubnetId": "subnet-d0dc51fb" }, { "ImageId": "ami-0d4cfd66", "InstanceType": "c3.8xlarge",
"WeightedCapacity": 32, "SubnetId": "subnet-64531413" }, { "ImageId": "ami-0d4cfd66", "InstanceType":
"c3.8xlarge", "WeightedCapacity": 32, "SubnetId": "subnet-0b1b8052" }, { "ImageId": "ami-0d4cfd66",
"InstanceType": "c3.2xlarge", "WeightedCapacity": 8, "SubnetId": "subnet-d0dc51fb" }, { "ImageId": "ami-
0d4cfd66", "InstanceType": "c3.2xlarge", "WeightedCapacity": 8, "SubnetId": "subnet-64531413" }, {
"ImageId": "ami-0d4cfd66", "InstanceType": "c3.2xlarge", "WeightedCapacity": 8, "SubnetId": "subnet-
0b1b8052" } ] }
```

# EC2 Spot Console



## Amazon EC2 Spot Instances

Spot Instances allow you to name your own price for Amazon EC2 compute capacity, so you can reduce operating costs and increase application throughput.

Get started

Getting started guide

### Name Your Price

With Spot instances, you never pay more than your bid price. Since Spot instances run on spare Amazon EC2 capacity, you can save up to 90% compared to On Demand prices.

Learn more about Spot instances

### Easily Provision Capacity

Select and request the instances that match your application and cost requirements, and optimize for lowest cost or even distribution.

Check out the Spot Bid Advisor

**Step 1: Find instance types**

Step 2: Configure

Step 3: Review

Select request type

**Request type**

● **Request**
Submit a one-time Spot instance request

○ **Request and Maintain**
Request a fleet of Spot instances to maintain your target capacity

○ **Reserve for duration**
Request a Spot instance with no interruption for 1 to 6 hours (a Spot block)

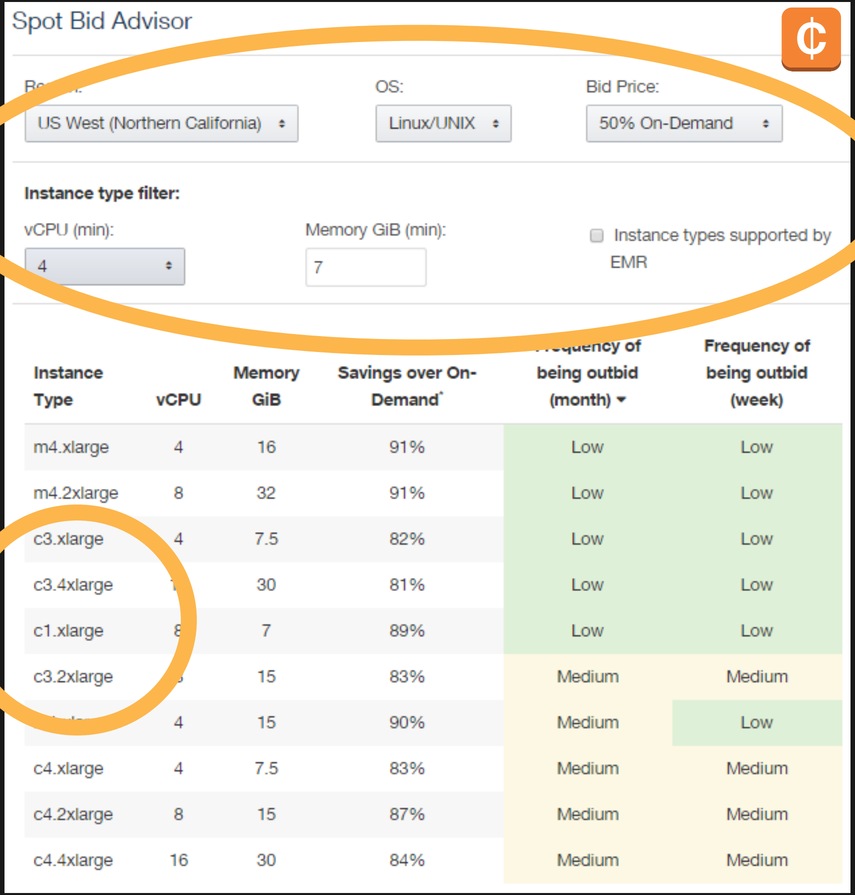
**Target capacity** 128 vCPUs

**AMI** Amazon Linux AMI 2016.(

**Instance type(s)**

- c3.large (2 vCPU, 3.75 GiB, 2 x 16)
- c3.xlarge (4 vCPU, 7.5 GiB, 2 x 40)
- c4.large (2 vCPU, 3.75 GiB, EBS only)
- c4.xlarge (4 vCPU, 7.5 GiB, EBS only)

# Amazon EC2 Spot Bid Advisor



Spot Bid Advisor

Region: US West (Northern California)

OS: Linux/UNIX

Bid Price: 50% On-Demand

**Instance type filter:**

vCPU (min): 4

Memory GiB (min): 7

□ Instance types supported by EMR

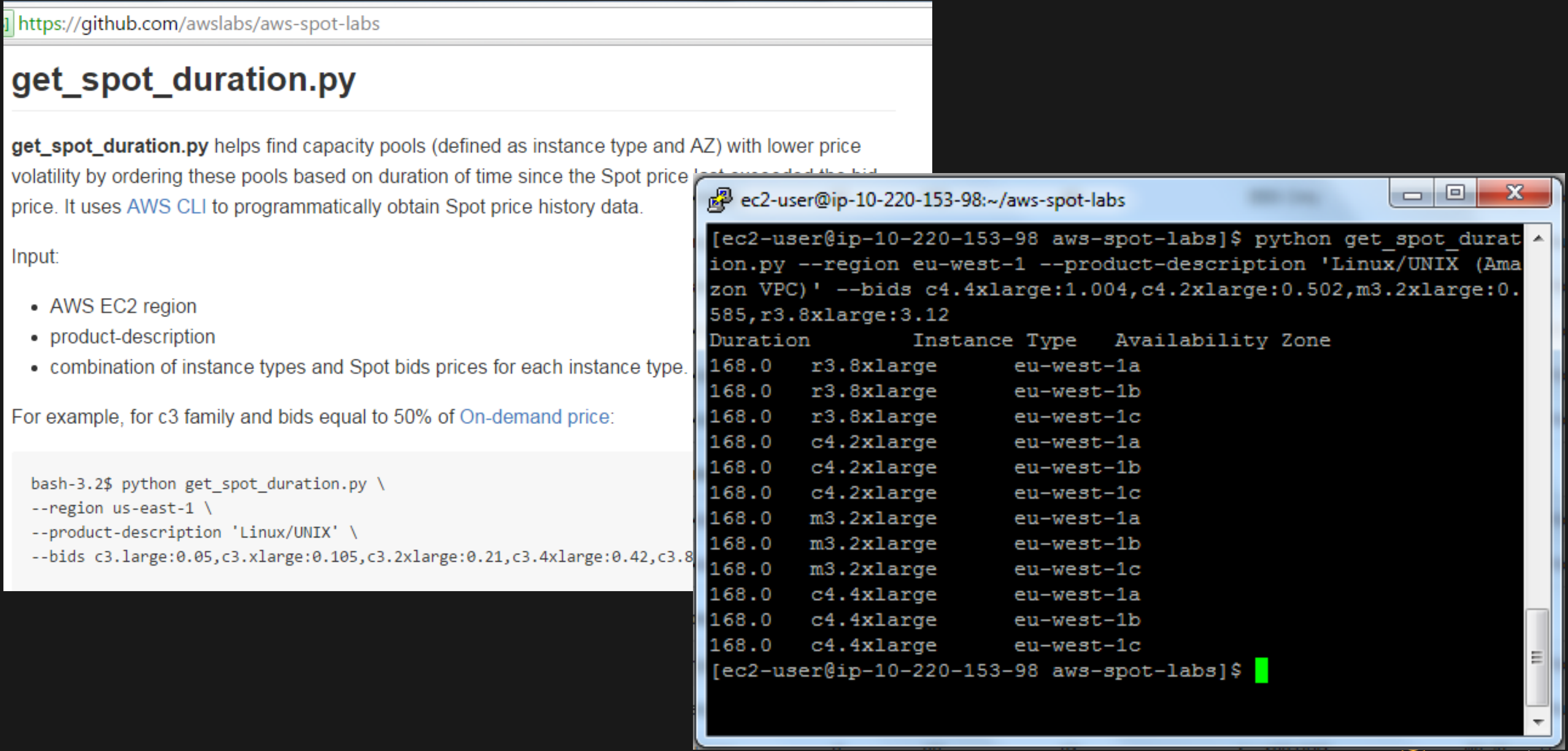
| Instance Type | vCPU | Memory GiB | Savings over On-Demand* | Frequency of being outbid (month) ▾ | Frequency of being outbid (week) |
|---|---|---|---|---|---|
| m4.xlarge | 4 | 16 | 91% | Low | Low |
| m4.2xlarge | 8 | 32 | 91% | Low | Low |
| c3.xlarge | 4 | 7.5 | 82% | Low | Low |
| c3.4xlarge | 16 | 30 | 81% | Low | Low |
| c1.xlarge | 8 | 7 | 89% | Low | Low |
| c3.2xlarge | 8 | 15 | 83% | Medium | Medium |
| [illegible] | 4 | 15 | 90% | Medium | Low |
| c4.xlarge | 4 | 7.5 | 83% | Medium | Medium |
| c4.2xlarge | 8 | 15 | 87% | Medium | Medium |
| c4.4xlarge | 16 | 30 | 84% | Medium | Medium |

1) We make this easy using the Spot bid advisor

2) With deliberate pool selection and bidding, you will keep your Spot instance as long as you need to

3) And with new features like Spot fleet diversified we do the heavy lifting for you...

# EC2 Spot Labs



https://github.com/awslabs/aws-spot-labs

## get_spot_duration.py

**get_spot_duration.py** helps find capacity pools (defined as instance type and AZ) with lower price volatility by ordering these pools based on duration of time since the Spot price last exceeded the bid price. It uses AWS CLI to programmatically obtain Spot price history data.

Input:

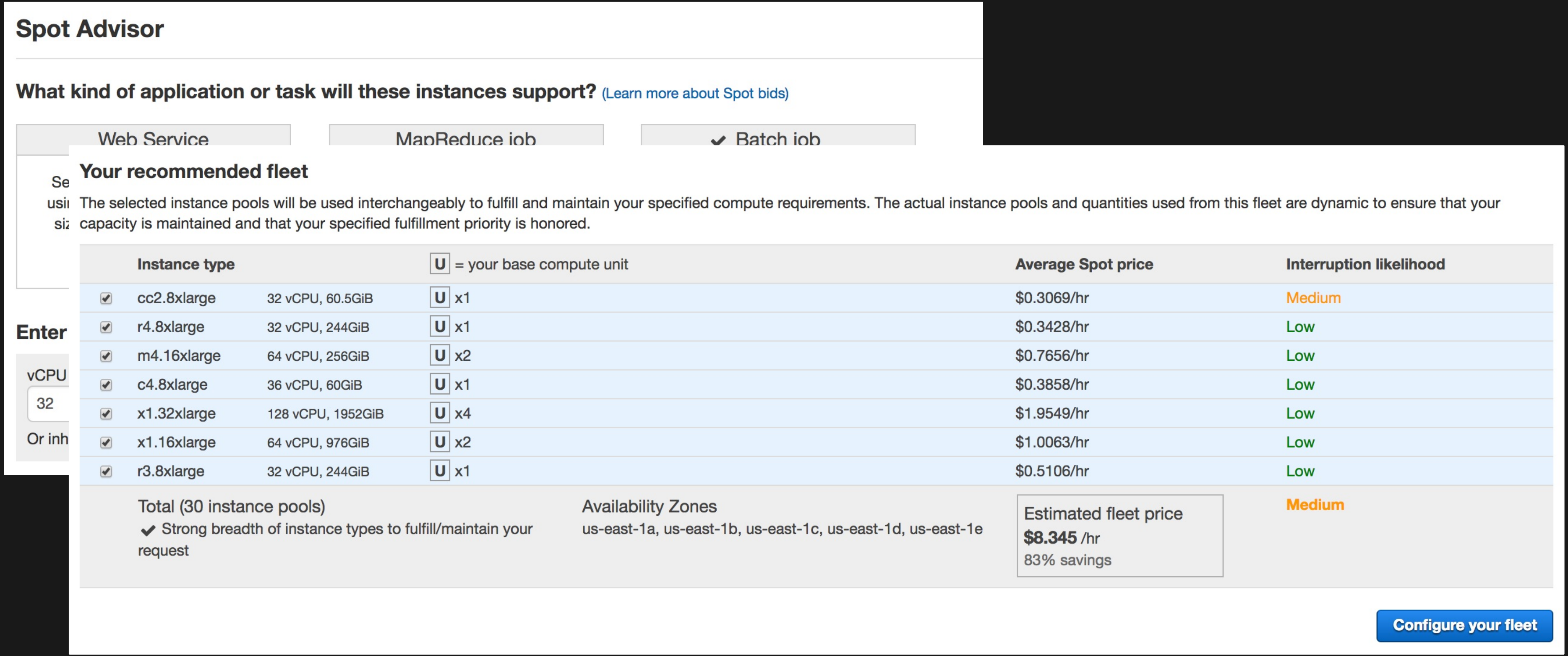
- AWS EC2 region
- product-description
- combination of instance types and Spot bids prices for each instance type.

For example, for c3 family and bids equal to 50% of On-demand price:

```
bash-3.2$ python get_spot_duration.py \
--region us-east-1 \
--product-description 'Linux/UNIX' \
--bids c3.large:0.05,c3.xlarge:0.105,c3.2xlarge:0.21,c3.4xlarge:0.42,c3.8
```

```
[ec2-user@ip-10-220-153-98 aws-spot-labs]$ python get_spot_durat
ion.py --region eu-west-1 --product-description 'Linux/UNIX (Ama
zon VPC)' --bids c4.4xlarge:1.004,c4.2xlarge:0.502,m3.2xlarge:0.
585,r3.8xlarge:3.12
Duration        Instance Type   Availability Zone
168.0   r3.8xlarge      eu-west-1a
168.0   r3.8xlarge      eu-west-1b
168.0   r3.8xlarge      eu-west-1c
168.0   c4.2xlarge      eu-west-1a
168.0   c4.2xlarge      eu-west-1b
168.0   c4.2xlarge      eu-west-1c
168.0   m3.2xlarge      eu-west-1a
168.0   m3.2xlarge      eu-west-1b
168.0   m3.2xlarge      eu-west-1c
168.0   c4.4xlarge      eu-west-1a
168.0   c4.4xlarge      eu-west-1b
168.0   c4.4xlarge      eu-west-1c
[ec2-user@ip-10-220-153-98 aws-spot-labs]$
```

# New Spot Advisor in Console



## Spot Advisor

**What kind of application or task will these instances support?** (Learn more about Spot bids)

Se
usi
siz

**Enter**

vCPU
32
Or inh

### Your recommended fleet

The selected instance pools will be used interchangeably to fulfill and maintain your specified compute requirements. The actual instance pools and quantities used from this fleet are dynamic to ensure that your capacity is maintained and that your specified fulfillment priority is honored.

| | Instance type | | U = your base compute unit | Average Spot price | Interruption likelihood |
|---|---|---|---|---|---|
| ☑ | cc2.8xlarge | 32 vCPU, 60.5GiB | U x1 | $0.3069/hr | Medium |
| ☑ | r4.8xlarge | 32 vCPU, 244GiB | U x1 | $0.3428/hr | Low |
| ☑ | m4.16xlarge | 64 vCPU, 256GiB | U x2 | $0.7656/hr | Low |
| ☑ | c4.8xlarge | 36 vCPU, 60GiB | U x1 | $0.3858/hr | Low |
| ☑ | x1.32xlarge | 128 vCPU, 1952GiB | U x4 | $1.9549/hr | Low |
| ☑ | x1.16xlarge | 64 vCPU, 976GiB | U x2 | $1.0063/hr | Low |
| ☑ | r3.8xlarge | 32 vCPU, 244GiB | U x1 | $0.5106/hr | Low |

Total (30 instance pools)
✓ Strong breadth of instance types to fulfill/maintain your request

Availability Zones
us-east-1a, us-east-1b, us-east-1c, us-east-1d, us-east-1e

Estimated fleet price
**$8.345** /hr
83% savings

**Medium**

# Sample script – two minutes left!

```
 $ if curl -s http://169.254.169.254/latest/meta-
data/spot/termination-time | \

 grep -q .*T.*Z; then instance_id=$(curl -s
http://169.254.169.254/latest/meta-data/instance-id); \
 aws elb deregister-instances-from-load-balancer \
  --load-balancer-name my-load-balancer \
  --instances $instance_id;
/env/bin/flushsessiontoDBonterminationscript.sh; fi
```

1) Check for 2 minute warning

2) If YES, detach instance from ELB

3) OTHERWISE, do nothing

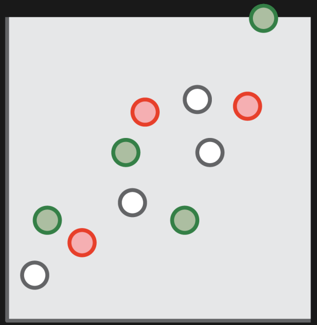
4) Sleep for 5 seconds

# Let’s see Spot fleet with Auto Scaling in action...

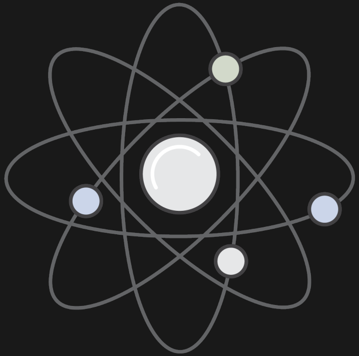
- Queue and Batch based processing
- Stateless Applications (e.g. web tiers)
- Amazon EC2 Container Service powered by Spot fleet

# Batch Processing with Amazon EC2 Spot

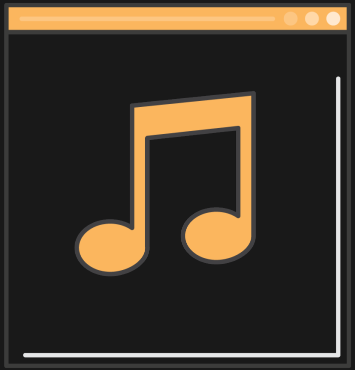
Batch oriented applications can leverage on-demand processing using EC2 Spot to save up to 90% cost:



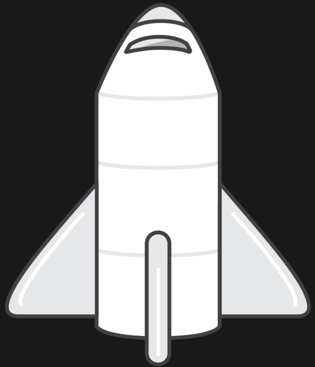
Monte Carlo simulation



Molecular modeling

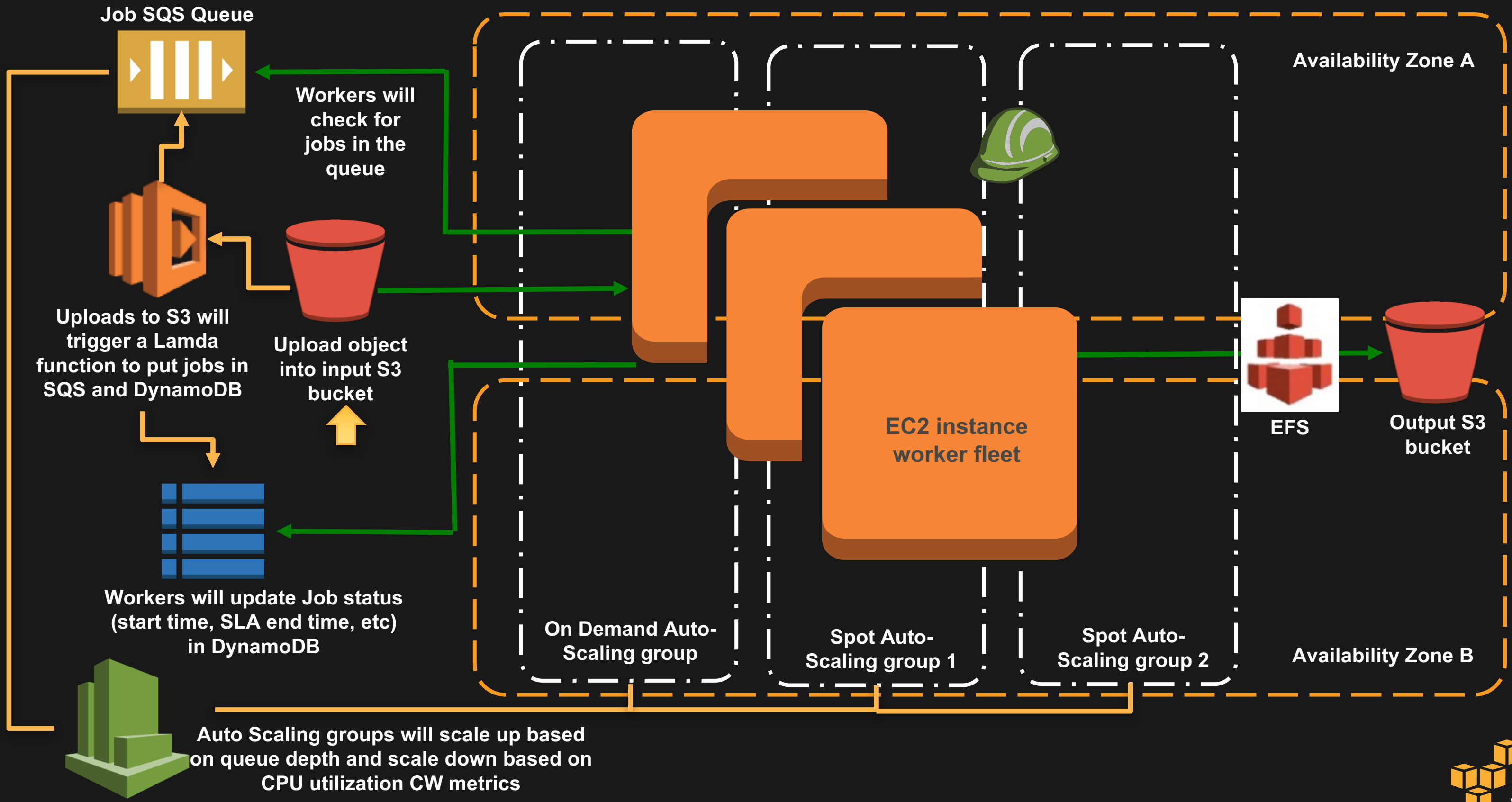


Media processing

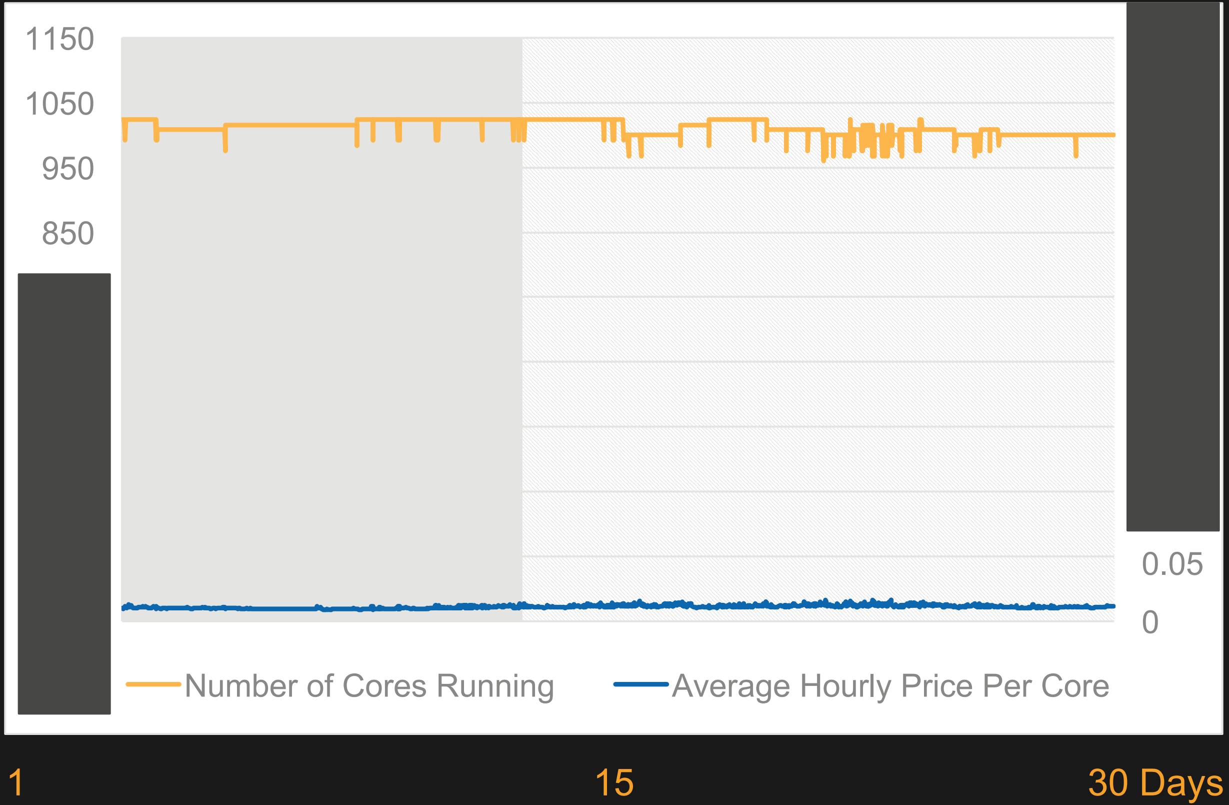


High energy simulations

# Queue based processing


Job SQS Queue
Workers will check for jobs in the queue
Uploads to S3 will trigger a Lamda function to put jobs in SQS and DynamoDB
Upload object into input S3 bucket
Workers will update Job status (start time, SLA end time, etc) in DynamoDB
Availability Zone A
EC2 instance worker fleet
EFS
Output S3 bucket
On Demand Auto-Scaling group
Spot Auto-Scaling group 1
Spot Auto-Scaling group 2
Availability Zone B
Auto Scaling groups will scale up based on queue depth and scale down based on CPU utilization CW metrics

# Results – Batch processing



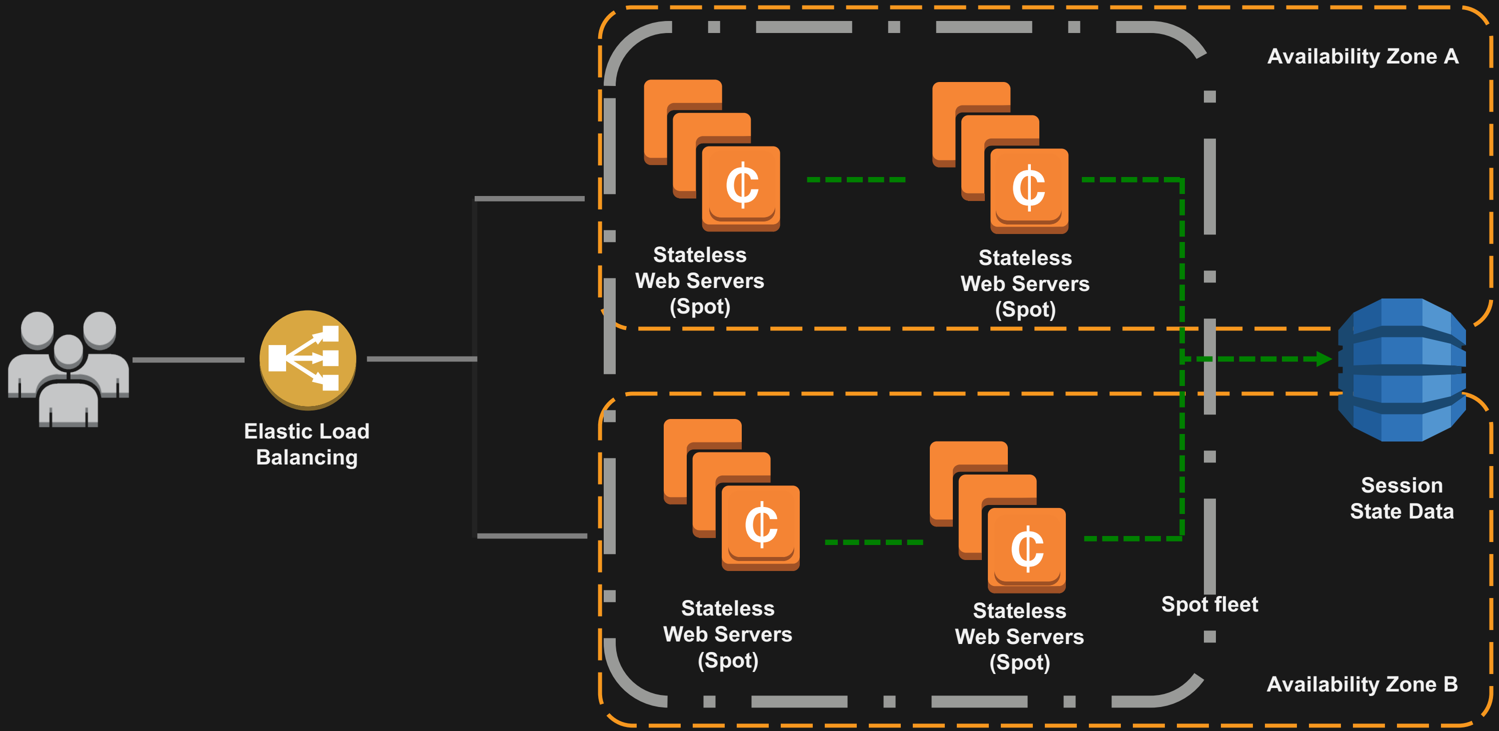
Requested 1000 vCores over 30 days

Minimum 960 vCores
Mode 1024 vCores
Average 1012 vCores

Average Price of $0.012 per vCore

Savings of over 80%

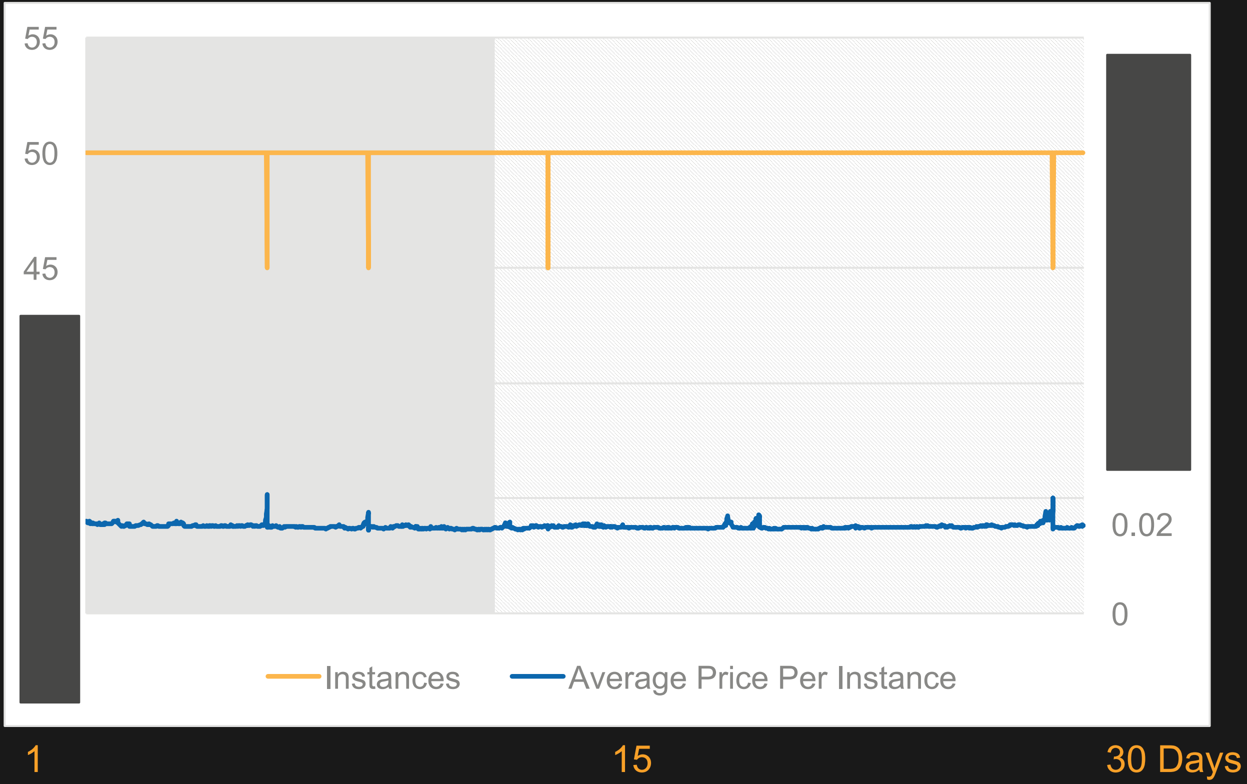
# Stateless Web Application



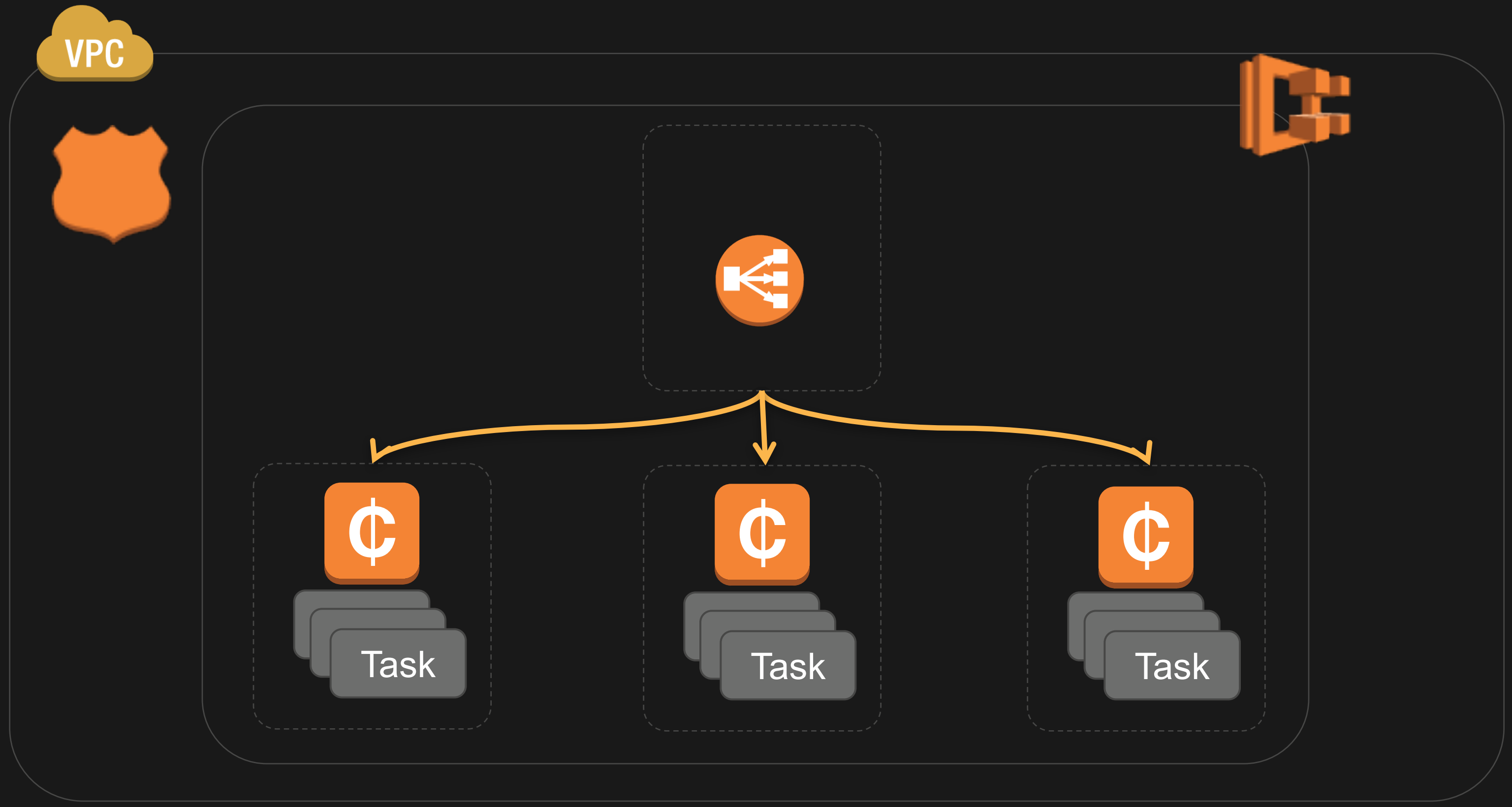
Availability Zone A
Stateless Web Servers (Spot)
Stateless Web Servers (Spot)
Elastic Load Balancing
Session State Data
Stateless Web Servers (Spot)
Stateless Web Servers (Spot)
Spot fleet
Availability Zone B

# Results - Web Application

- 50 instances requested, over 30 days.
- Never dropped below 45 instances
- 85% discount if you wanted 50 and could withstand dropping to 45
- If you only wanted 45 the discount is still 83%

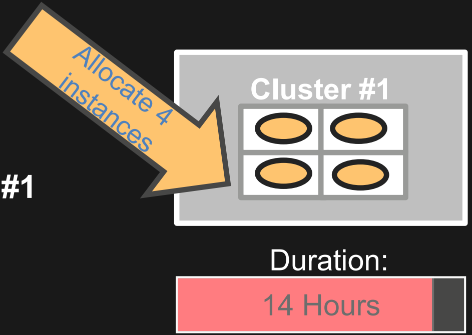


# Amazon EC2 Container Service powered by Spot fleet



# Amazon Elastic Map Reduce powered by Spot fleet

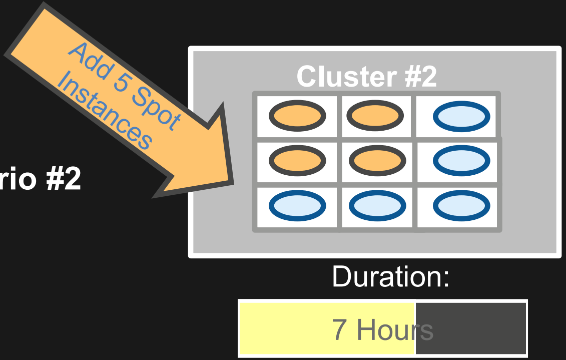
**Scenario #1**



**#1: Cost without Spot**
4 instances *14 hrs * $0.45
Total = $25.20

**Mix Spot and On Demand EC2 Instances to maximize value and minimize processing time**

**Scenario #2**



**#2: Cost with Spot**
4 instances *7 hrs * $0.45 = $12.60 +
5 instances * 7 hrs * $0.225 = $7.875
Total = $20.475

**Time Savings:** 50%

**Cost Savings:** ~19%

AWS SUMMIT

# Thank you!

amazon web services

# Appendix

# Reference links

EC2 Spot Documentation:

http://aws.amazon.com/ec2/spot/
http://aws.amazon.com/ec2/spot/bid-advisor/
http://aws.amazon.com/ec2/spot/getting-started/
http://aws.amazon.com/ec2/spot/faqs/
http://aws.amazon.com/ec2/spot/testimonials/

User Guide

http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/using-spot-instances.html
http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/spot-fleet.html

Helpful AWS Blog Posts

https://aws.amazon.com/blogs/aws/focusing-on-spot-instances-lets-talk-about-best-practices/
https://aws.amazon.com/blogs/aws/building-price-aware-applications-using-ec2-spot-instances/
https://aws.amazon.com/blogs/compute/cost-effective-batch-processing-with-amazon-ec2-spot/
https://aws.amazon.com/blogs/compute/dynamic-scaling-with-ec2-spot-fleet/

# Reference links continued

EC2 Spot Labs:
https://github.com/awslabs/ec2-spot-labs

Amazon EC2 Spot fleet Jenkins plugin:
https://wiki.jenkins-ci.org/display/JENKINS/Amazon+EC2+Fleet+Plugin

Deploy a Deep Learning Framework on Amazon ECS powered by EC2 Spot fleet
https://github.com/awslabs/ecs-deep-learning-workshop