

aws  
okc

# Lab: Build a Redundant WordPress Stack in AWS Part 2

Nathan Aker

Director of IT @

[ SYMPHONY TALENT ]

Lab:

## Build a Redundant WordPress Stack in AWS

Intent:

- Get hands on with AWS services with a realistic use case
- Gain confidence to jump in and play with AWS services
- Stay on track to get through the lab ( Exposure ≠ Training )



= Warning – Cost Associated!!!

# Acknowledgement



Lab Taken from:

A Cloud Guru's "AWS Certified Solutions Architect Associate" course

<https://acloud.guru/learn/aws-certified-solutions-architect-associate>



# Prerequisites



- AWS Free Tier Account: [aws.amazon.com/free](https://aws.amazon.com/free)

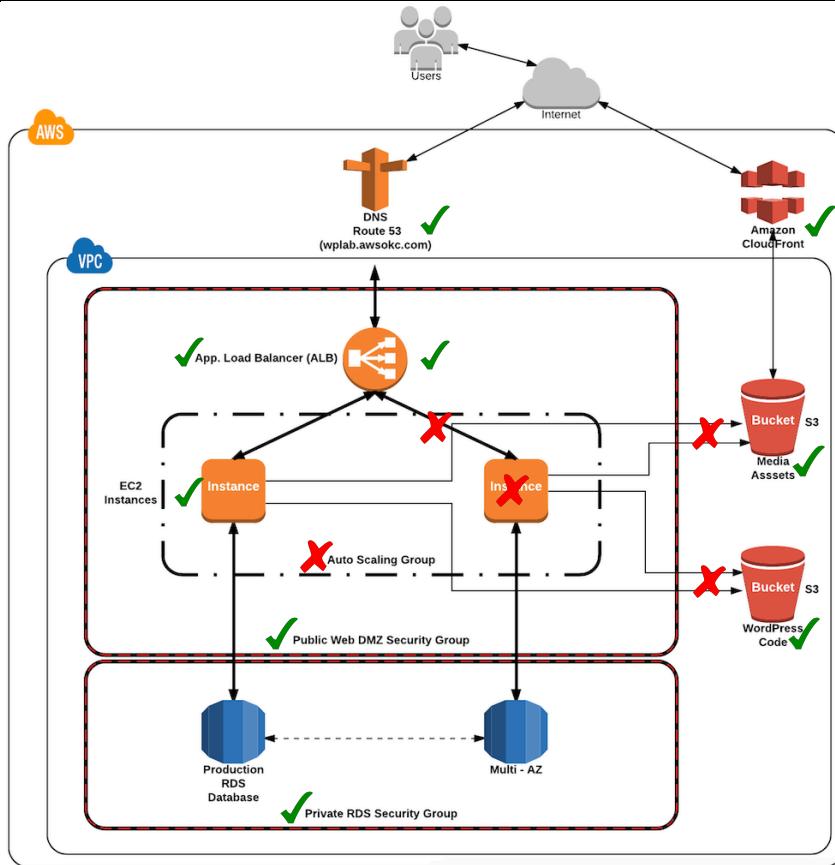


The screenshot shows the AWS Free Tier landing page. At the top, there's a navigation bar with links for Menu, Products, Solutions, Pricing, Resources, More, English, My Account, and Sign In to the Console. Below the navigation is a large banner with a purple-to-orange gradient background. It features the text "AWS Free Tier" and a subtext: "The AWS Free Tier enables you to gain free, hands-on experience with the AWS platform, products, and services." A yellow "Create a Free Account" button is centered in the banner. Below the banner, there are three buttons: "Free Tier Details", "Get Started", and "Free Tier Software". The main content area is titled "AWS Free Tier Details" and includes filters for "FEATURED", "12 MONTHS FREE", "ALWAYS FREE", "PRODUCT CATEGORIES", and "ALL". It highlights several free products: "Amazon EC2" (750 Hours per month), "Amazon QuickSight" (1 GB of SPICE capacity), and "12 months free and always free products". A note at the bottom states: "AWS Free Tier includes offers that expire 12 months following sign-up and others that...".

# Prerequisites

- Download Resources from:
  - <https://github.com/awsokc/wplab>

# Recap



# Now the Fun Stuff



We previously setup a functioning WordPress instance, but one that only had one single web server and lacked any kind of resilience or self-healing capabilities.

Now, we'll add Resilience via Auto-Scaling and CloudFront

# Re-Create ALB (ELB)



Compute

**EC2**

EC2 Container Service

Lightsail

Elastic Beanstalk

Lambda

Batch

**LOAD BALANCING**

**Load Balancers**

Target Groups



Welcome to Elastic Load Balancing  
Select load balancer type

Elastic Load Balancing supports two types of load balancers: Application Load Balancers (new) and Classic Load Balancers. Choose the load balancer type that meets your needs.

Application Load Balancer

Classic Load Balancer

**Application Load Balancer**

Preferred for HTTP/HTTPS

An Application Load Balancer makes routing decisions at the application layer (HTTP/HTTPS), supports path-based routing, and can route requests to one or more ports on each EC2 instance or container instance in your VPC.

**Classic Load Balancer**

A Classic Load Balancer makes routing decisions at either the transport layer (TCP/SSL) or the application layer (HTTP/HTTPS), and supports either EC2-Classic or a VPC.

# Re-Create ALB (ELB)



Services    Resource Groups    N. Virginia    Support

1. Configure Load Balancer    2. Configure Security Settings    3. Configure Security Groups    4. Configure Routing    5. Register Targets    6. Review

### Step 1: Configure Load Balancer

#### Basic Configuration

To configure your load balancer, provide a name, select a scheme, specify one or more listeners, and select a network. The default configuration is an Internet-facing load balancer in the selected network with a listener that receives HTTP traffic on port 80.

Name: My-ALB  
Scheme: Internet-facing  
IP address type: ipv4

#### Listeners

A listener is a process that checks for connection requests, using the protocol and port that you configured.

Load Balancer Protocol	Load Balancer Port
HTTP	80

Add listener

#### Availability Zones

Specify the Availability Zones to enable for your load balancer. The load balancer routes traffic to the targets in these Availability Zones only. You can specify only one subnet per Availability Zone. You must specify subnets from at least two Availability Zones to increase the availability of your load balancer.

VPC: vpc-9aaef8fe (172.31.0.0/16) (default)

Availability Zone	Subnet ID	Subnet IPv4 CIDR	Name
us-east-1a	subnet-43b37b69	172.31.48.0/20	
us-east-1b	subnet-5955ee2f	172.31.0.0/20	
us-east-1c	subnet-2cd60b74	172.31.16.0/20	

Cancel    Next: Configure Security Settings

# Re-Create ALB (ELB)



Services ▾ Resource Groups ▾ N. Virginia ▾ Support ▾

1. Configure Load Balancer 2. Configure Security Settings 3. Configure Security Groups 4. Configure Routing 5. Register Targets 6. Review

### Step 3: Configure Security Groups

A security group is a set of firewall rules that control the traffic to your load balancer. On this page, you can add rules to allow specific traffic to reach your load balancer. First, decide whether to create a new security group or select an existing one.

Assign a security group:  Create a new security group  Select an existing security group

Filter: VPC security groups

Security	Name	Description
sg-b63799cc	awseb-e-imusstpmeg-stack-AWSEBSecurityGroup-1Y1UU7UMXPB7D	SecurityGroup for ElasticBeanstalk environment
sg-845326f8	CodeDeploySampleStack-2dw6722807qfwltsm7vi-SecurityGroup-1C62RDO7806MK	Enable HTTP access via port 80 and SSH access
sg-c21c0ba	default	default VPC security group
sg-dc70b9a1	ElasticMapReduce-master	Master group for Elastic MapReduce created on 2016-09-21T13:17:17.7
sg-df70b9a2	ElasticMapReduce-slave	Slave group for Elastic MapReduce created on 2016-09-21T13:17:17.7
sg-28f98452	launch-wizard-1	launch-wizard-1 created 2016-09-21T13:17:17.7
sg-44d3c43e	launch-wizard-2	launch-wizard-2 created 2016-10-18T19:56:08.6
sg-33c41a4f	launch-wizard-3	launch-wizard-3 created 2017-01-19T18:34:32.9
sg-cd563bb1	launch-wizard-4	launch-wizard-4 created 2017-02-06T23:08:26.9
sg-34bd8245	launch-wizard-5	launch-wizard-5 created 2017-07-26T15:07:24.4
sg-f1c2fd80	launch-wizard-6	launch-wizard-6 created 2017-07-26T15:15:03.3
sg-a9b768d9	launch-wizard-7	launch-wizard-7 created 2017-08-14T12:11:53.3
sg-16349a6c	rds-awseb-e-imusstpmeg-stack-awsebrdsdbsecuritygroup-1udrb80khgap-lzjr	Security group for RDS DB Security Group awseb-e-imusstpmeg-stack-awsebrdsdbsecuritygroup-1udrb80khgap-lzjr
sg-a4dfbdde	rds-launch-wizard	Created from the RDS Management Console
sg-284d2354	rds-launch-wizard-1	Created from the RDS Management Console
sg-22abe552	RDS-SG	RDS-SG
sg-1c2ee964	smartpostrs	slemma
sg-ce14fb94	symmetricids-aurora	SymmetricIDS Aurora
sg-e3a4ea93	Web-DMZ	Web-DMZ

Cancel Previous Next: Configure Routing

# Re-Create ALB (ELB)



Screenshot of the AWS CloudFormation console showing the 'Configure Routing' step of a new Load Balancer creation.

The navigation bar at the top includes 'Services', 'Resource Groups', 'N. Virginia', and 'Support'.

The steps are numbered 1. Configure Load Balancer, 2. Configure Security Settings, 3. Configure Security Groups, 4. Configure Routing (the current step), 5. Register Targets, and 6. Review.

**Step 4: Configure Routing**

Your load balancer routes requests to the targets in this target group using the protocol and port that you specify, and performs health checks on the targets using these health check settings. Note that each target group can be associated with only one load balancer.

**Target group**

Target group: New target group  
Name: MyWebServers  
Protocol: HTTP  
Port: 80

**Health checks**

Protocol: HTTP  
Path: /healthy.html

[Advanced health check settings](#)

Buttons at the bottom: Cancel, Previous, Next: Register Targets

# Re-Create ALB (ELB)



Services ▾ Resource Groups ▾ N. Virginia ▾ Support ▾

1. Configure Load Balancer 2. Configure Security Settings 3. Configure Security Groups 4. Configure Routing 5. Register Targets 6. Review

### Step 5: Register Targets

Register targets with your target group. If you register an instance running in an enabled Availability Zone, the load balancer starts routing requests to the instance as soon as the registration process completes and the instance passes the initial health checks.

#### Registered instances

To deregister instances, select one or more registered instances and then click Remove.

Remove

Instance	Name	Port	State	Security groups	Zone
No instances available.					

#### Instances

To register additional instances, select one or more running instances, specify a port, and then click Add. The default port is the port specified for the target group. If the instance is already registered on the specified port, you must specify a different port.

Add to registered on port 80

Search Instances X

Instance	Name	State	Security groups	Zone	Subnet ID	Subnet CIDR
No instances available.						

Cancel Previous Next: Review

# Re-Create ALB (ELB)



Screenshot of the AWS CloudFormation console showing the 'Review' step of creating a new Application Load Balancer (ALB).

The review step displays the configuration details for the ALB, which has been set up with the following parameters:

- Load balancer:**
  - Name: My-ALB
  - Scheme: internet-facing
  - Listeners: Port:80 - Protocol:HTTP
  - IP address type: ipv4
  - VPC: vpc-9aaef8fe
  - Subnets: subnet-43b37b69, subnet-5955ee2f, subnet-2cd60b74, subnet-11ef7b74, subnet-7362054e, subnet-6ab70966
  - Tags
- Security settings:**
  - Certificate name
  - Security policy name
- Security groups:**
  - Security groups: sg-c21cf0ba
- Routing:**
  - Target group: New target group
  - Target group name: MyWebServers
  - Port: 80
  - Protocol: HTTP
  - Health check protocol: HTTP
  - Path: /healthy.html
  - Health check port: traffic port
  - Healthy threshold: 5
  - Unhealthy threshold: 2
  - Timeout: 5
  - Interval: 30
  - Success codes: 200
- Targets:** (List of targets is currently empty)

At the bottom right, there are buttons for **Create**, **Cancel**, and **Previous**.

# Step 1 – Backup WP Files



List S3 bucket names with \$ aws s3 ls

```
[[ec2-user@ip-172-31-15-213 html]$ cd /var/www/html/
[[ec2-user@ip-172-31-15-213 html]$ ls
healthy.html      wp-blog-header.php    wp-includes        wp-signup.php
index.php         wp-comments-post.php  wp-links-opml.php  wp-trackback.php
license.txt       wp-config.php       wp-load.php       xmlrpc.php
readme.html       wp-config-sample.php wp-login.php
wp-activate.php   wp-content          wp-mail.php
wp-admin          wp-cron.php        wp-settings.php
[[ec2-user@ip-172-31-15-213 html]$ aws s3 ls
2016-11-24 06:53:08 appliesfromdmptestpolicy
2017-08-04 05:10:27 aws-codedstar-us-east-1-534106927381
2017-08-04 05:11:25 aws-codedstar-us-east-1-534106927381-firstproject-pipe
2016-09-22 15:59:15 aws-nodejs-dev-serverlessdeploymentbucket-j14n2fi6vibb
2017-08-29 10:39:15 aws-second-dev-serverlessdeploymentbucket-xzsdyrnxzrjp
2017-08-31 23:05:54 awsokc-wpcode
2017-08-31 23:06:13 awsokc-wpmedia
2016-10-12 17:44:26 cf-templates-7ji395qq8mmj-us-east-1
2016-02-17 19:19:22 elasticbeanstalk-us-east-1-534106927381
2017-02-02 09:08:19 feedTest
```

# Step 1 – Backup WP Files



Copy WP code to S3 bucket with:

```
$ aws s3 cp --recursive /var/www/html/ s3://awsokc-wpcode
```

```
Downloads — ec2-user@ip-172-31-15-213:/var/www/html — ssh ec2-user@54.2...  
[ec2-user@ip-172-31-15-213 html]$ clear  
[ec2-user@ip-172-31-15-213 html]$ aws s3 cp --recursive /var/www/html/ s3://awsokc-wpcode  
Completed 8 Bytes/~77.4 KiB (67 Bytes/s) with ~13 file(s) remaining (calculating  
upload: ./healthy.html to s3://awsokc-wpcode/healthy.html  
Completed 8 Bytes/~77.4 KiB (67 Bytes/s) with ~12 file(s) remaining (calculating  
Completed 3.7 KiB/~77.4 KiB (30.9 KiB/s) with ~12 file(s) remaining (calculating  
upload: wp-admin/admin-ajax.php to s3://awsokc-wpcode/wp-admin/admin-ajax.php  
Completed 3.7 KiB/~77.4 KiB (30.9 KiB/s) with ~11 file(s) remaining (calculating  
Completed 6.3 KiB/~77.4 KiB (48.2 KiB/s) with ~11 file(s) remaining (calculating  
upload: wp-admin/admin-footer.php to s3://awsokc-wpcode/wp-admin/admin-footer.ph  
p  
Completed 6.3 KiB/~77.4 KiB (48.2 KiB/s) with ~10 file(s) remaining (calculati  
Completed 10.4 KiB/~122.5 KiB (72.9 KiB/s) with ~16 file(s) remaining (calculati  
upload: wp-admin/async-upload.php to s3://awsokc-wpcode/wp-admin/async-uploa  
p  
Completed 10.4 KiB/~122.5 KiB (72.9 KiB/s) with ~15 file(s) remaining (calculati  
Completed 10.8 KiB/~195.1 KiB (71.8 KiB/s) with ~24 file(s) remaining (calculati  
upload: wp-admin/admin-functions.php to s3://awsokc-wpcode/wp-admin/admin-functi  
ons.php  
Completed 10.8 KiB/~195.1 KiB (71.8 KiB/s) with ~23 file(s) remaining (calculati  
Completed 15.4 KiB/~497.9 KiB (89.8 KiB/s) with ~54 file(s) remaining (calculati  
upload: wp-admin/credits.php to s3://awsokc-wpcode/wp-admin/credits.php  
Completed 15.4 KiB/~497.9 KiB (89.8 KiB/s) with ~53 file(s) remaining (calculati  
Completed 17.1 KiB/~772.8 KiB (92.1 KiB/s) with ~61 file(s) remaining (calculati  
upload: wp-admin/admin-post.php to s3://awsokc-wpcode/wp-admin/admin-post.php  
Completed 17.1 KiB/~772.8 KiB (92.1 KiB/s) with ~60 file(s) remaining (calculati  
Completed 24.3 KiB/~772.8 KiB (129.3 KiB/s) with ~60 file(s) remaining (calculat  
upload: wp-admin/admin-header.php to s3://awsokc-wpcode/wp-admin/admin-header.ph  
p
```

# Step 1 – Backup WP Files



Confirm successful copy with:

```
$ aws s3 ls s3://yourbucket
```

```
Downloads — ec2-user@ip-172-31-15-213:/var/www/html — ssh ec2-user@54.2...  
[ec2-user@ip-172-31-15-213 html]$ aws s3 ls s3://awsokc-wpcode  
PRE wp-admin/  
PRE wp-content/  
PRE wp-includes/  
     8 healthy.html  
    418 index.php  
19935 license.txt  
  7413 readme.html  
  5447 wp-activate.php  
   364 wp-blog-header.php  
16227 wp-comments-post.php  
 2853 wp-config-sample.php  
  3097 wp-config.php  
  3286 wp-cron.php  
 2422 wp-links-opml.php  
  3301 wp-load.php  
34327 wp-login.php  
  8048 wp-mail.php  
16200 wp-settings.php  
29924 wp-signup.php  
  4513 wp-trackback.php  
  3065 xmlrpc.php  
[ec2-user@ip-172-31-15-213 html]$
```

# Step 2 – Upload an Image File



First, confirm that there are no images in wp-content directory:

```
Downloads — ec2-user@ip-172-31-15-213:/var/www/html/wp-content — ssh ec...
[[ec2-user@ip-172-31-15-213 html]$ ls
healthy.html      wp-blog-header.php    wp-includes        wp-signup.php
index.php         wp-comments-post.php  wp-links-opml.php  wp-trackback.php
license.txt       wp-config.php       wp-load.php       xmlrpc.php
readme.html       wp-config-sample.php wp-login.php
wp-activate.php   wp-content          wp-mail.php
wp-admin          wp-cron.php        wp-settings.php
[[ec2-user@ip-172-31-15-213 html]$ cd wp-content
[[ec2-user@ip-172-31-15-213 wp-content]$ ls
index.php  plugins  themes
[[ec2-user@ip-172-31-15-213 wp-content]$ ]]
```

# Step 2 – Upload an Image File



Upload an image:

The image shows three sequential screenshots of a WordPress site. The first screenshot on the left is the WordPress login page, where the username 'awsokc2017' is entered in the 'Username or Email Address' field. The second screenshot in the middle shows the WordPress dashboard with a 'Updates 1' notification. The third screenshot on the right shows the 'Media' section of the dashboard, specifically the 'Upload New Media' page. This page features a dashed box for dropping files, a 'Select Files' button, and a note about the multi-file uploader.

Username or Email Address  
awsokc2017

Password  
\*\*\*\*\*

Remember Me

Log In

Lost your password?  
← Back to AWSOKC Lab

Dashboard

Home

Updates 1

Posts

Media

Pages

Comments

Appearance

AWSOKC Lab

Dashboard

Welcome to AWSOKC Lab

We've assembled a team of experts to help you learn more about AWS services and how they can benefit your organization.

Add New

Library

Upload New Media

Drop files here

Select Files

You are using the multi-file uploader. Problems? Try the [browser uploader](#) instead.

Maximum upload file size: 2 MB.

AWSOKC Logo

# Step 2 – Upload an Image File



Confirm new uploads directory

```
Downloads — ec2-user@ip-172-31-15-213:/var/www/html/wp-content/uploads/2...
[ec2-user@ip-172-31-15-213 html]$ ls
healthy.html      wp-blog-header.php    wp-includes      wp-signup.php
index.php         wp-comments-post.php  wp-links-opml.php  wp-trackback.php
license.txt       wp-config.php        wp-load.php     xmlrpc.php
readme.html       wp-config-sample.php wp-login.php
wp-activate.php   wp-content          wp-mail.php
wp-admin          wp-cron.php         wp-settings.php
[ec2-user@ip-172-31-15-213 html]$ cd wp-content
[ec2-user@ip-172-31-15-213 wp-content]$ ls
index.php  plugins  themes
[ec2-user@ip-172-31-15-213 wp-content]$ ls
index.php  plugins  themes  uploads
[ec2-user@ip-172-31-15-213 wp-content]$ cd uploads/
[ec2-user@ip-172-31-15-213 uploads]$ ls
2017
[ec2-user@ip-172-31-15-213 uploads]$ cd 2017/
[ec2-user@ip-172-31-15-213 2017]$ ls
10
[ec2-user@ip-172-31-15-213 2017]$ cd 10/
[ec2-user@ip-172-31-15-213 10]$ ls
AWSOKC_Logo.jpg
[ec2-user@ip-172-31-15-213 10]$
```

# Step 2 – Upload an Image File



Insert Pic into Post

Screenshot of the WordPress admin interface showing the process of inserting an image into a post.

The main screen shows the Posts dashboard with one published post titled "AWSOKC - First Post!". The post content area contains the text "I sure hope this lab is working well for you!" and a visual editor toolbar above it.

A modal window titled "Edit Post" is open, showing the post title and content. Below the content, there is an "Add Media" button and a rich text editor toolbar.

An "Insert Media" sidebar is visible on the right, listing options like "Create Gallery", "Create Audio Playlist", "Create Video Playlist", and "Featured Image". The "Featured Image" option is selected.

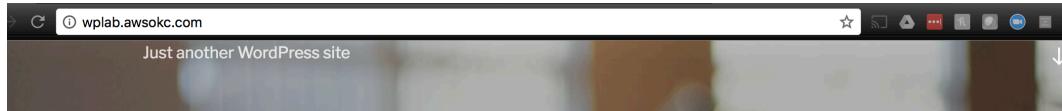
A larger "Insert Media" modal is displayed over the main content area. It shows a preview of the "aws okc" logo image with a checkmark indicating it is selected. Below the preview is a "Insert into post" button.

At the bottom right of the slide, there are two buttons: "Move to Trash" and "Update".

# Step 2 – Upload an Image File



Confirm pic in post



**POSTS**

OCTOBER 5, 2017

AWSOKC – First Post!

I sure hope this lab is working well for you!



Search ...

**RECENT POSTS**

AWSOKC – First Post!

**RECENT COMMENTS**

A WordPress Commenter on AWSOKC – First Post!

**ARCHIVES**

October 2017

Image Address: [http://wplab.awsokc.com/wp-content/uploads/2017/10/AWSOKC\\_BlackBackground.jpg](http://wplab.awsokc.com/wp-content/uploads/2017/10/AWSOKC_BlackBackground.jpg)

# Step 3 – Rewrite Images to S3 / CF



Part 1 – update WP to push images to S3 (served through CloudFront)

Part 2 – Update WP to reference Cloudfront.

First confirm S3 bucket - \$ aws s3 ls

```
[[ec2-user@ip-172-31-15-213 10]$ cd /var/www/html/
[[ec2-user@ip-172-31-15-213 html]$ ls
healthy.html      wp-blog-header.php    wp-includes        wp-signup.php
index.php         wp-comments-post.php  wp-links-opml.php  wp-trackback.php
license.txt       wp-config.php       wp-load.php       xmlrpc.php
readme.html       wp-config-sample.php wp-login.php
wp-activate.php   wp-content          wp-mail.php
wp-admin          wp-cron.php        wp-settings.php
[[ec2-user@ip-172-31-15-213 html]$ aws s3 ls
2016-11-24 06:53:08 appliesfromdmptestpolicy
2017-08-04 05:10:27 aws-codedstar-us-east-1-534106927381
2017-08-04 05:11:25 aws-codedstar-us-east-1-534106927381-firstproject-pipe
2016-09-22 15:59:15 aws-nodejs-dev-serverlessdeploymentbucket-j14n2fi6vibb
2017-08-29 10:39:15 aws-second-dev-serverlessdeploymentbucket-xzsdyrnxzrjp
2017-08-31 23:05:54 awsokc-wpcode
2017-08-31 23:06:13 awsokc-wpmedia
2016-10-12 17:44:26 cf_templates_7j395qq8mmj-us-east-1
2016-02-17 19:19:22 elasticbeanstalk-us-east-1-534106927381
2017-02-02 09:08:19 feedTest
2017-08-29 10:53:15 helloworld-dev-serverlessdeploymentbucket-1qzxuncguhyvt
2016-09-29 20:22:56 inbound-jobdata-1
2017-01-18 05:37:57 jobprocessfromdmptushdev5
2016-11-09 19:06:29 l3-storage-test
2017-08-15 22:18:51 l3-storage-test
```

# Step 3 – Rewrite Images to S3 / CF



S3 Sync Command:

```
$ aws s3 sync --delete /var/www/html/wp-content/uploads/ s3://awsokc-wpmedia
```

```
[ec2-user@ip-172-31-15-213 html]$ aws s3 sync --delete /var/www/html/wp-content/uploads/ s3://awsokc-wpmedia
upload: wp-content/uploads/2017/10/AWSOKC_Logo.jpg to s3://awsokc-wpmedia/2017/10/AWSOKC_Logo.jpg
[[ec2-user@ip-172-31-15-213 html]$ aws s3 ls s3://awsokc-wpmedia
PRE 2017/
[ec2-user@ip-172-31-15-213 html]$ cd]
```

A screenshot of the Amazon S3 console. The top navigation bar shows "Services" and "Resource Groups". Below it, the path "Amazon S3 > awsokc-wpmedia / 2017 / 10" is visible. A search bar contains the placeholder "Type a prefix and press Enter to search. Press ESC to clear.". There are three buttons: "Upload", "Create folder", and "More". A table lists files in the folder, with one file highlighted: "AWSOKC\_Logo.jpg".

Name
AWSOKC_Logo.jpg

# Step 3 – Rewrite Images to S3 / CF



Add URL rewrite rule (while in html directory):

```
$ sudo wget https://s3-eu-west-1.amazonaws.com/acloudguru-wp/htaccess
$ sudo mv htaccess .htaccess
```

```
Downloads — ec2-user@ip-172-31-15-213:/var/www/html — ssh ec2-user@54.2...
[ec2-user@ip-172-31-15-213 html]$ sudo wget https://s3-eu-west-1.amazonaws.com/a
cloudguru-wp/htaccess
--2017-10-05 02:37:40--  https://s3-eu-west-1.amazonaws.com/acloudguru-wp/htacce
ss
Resolving s3-eu-west-1.amazonaws.com (s3-eu-west-1.amazonaws.com)... 52.218.65.2
0
Connecting to s3-eu-west-1.amazonaws.com (s3-eu-west-1.amazonaws.com) |52.218.65.
20|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 167 [binary/octet-stream]
Saving to: 'htaccess'

htaccess          100%[=====]      167  --.-KB/s   in 0s

2017-10-05 02:37:41 (13.2 MB/s) - 'htaccess' saved [167/167]

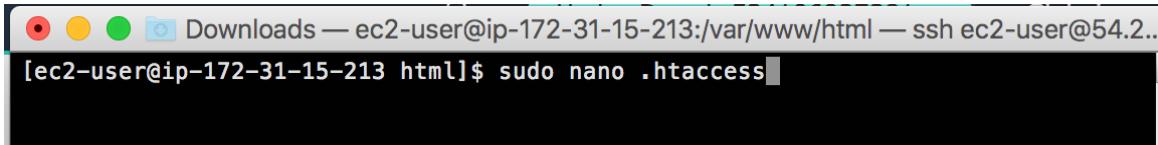
[ec2-user@ip-172-31-15-213 html]$ sudo mv htaccess .htaccess
[ec2-user@ip-172-31-15-213 html]$
```

# Step 3 – Rewrite Images to S3 / CF



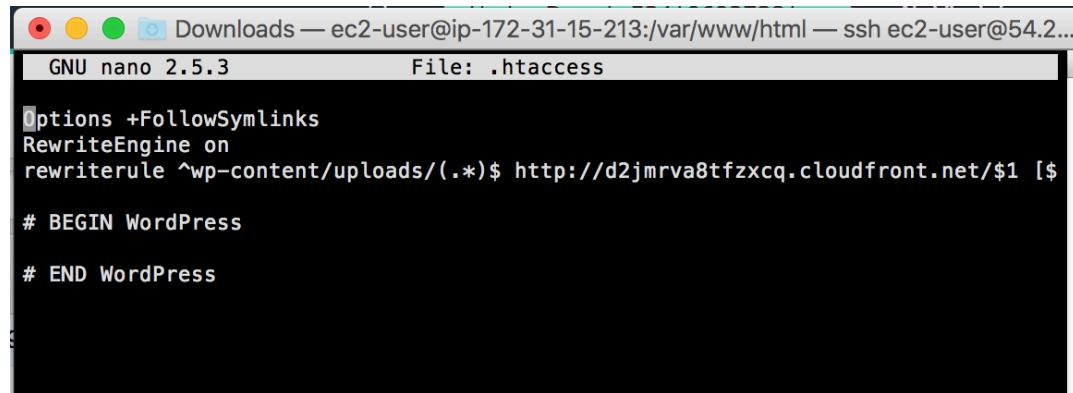
View rewrite rule in .htaccess file:

```
$ sudo nano .htaccess
```



```
Downloads — ec2-user@ip-172-31-15-213:/var/www/html — ssh ec2-user@54.2...
```

```
[ec2-user@ip-172-31-15-213 html]$ sudo nano .htaccess
```



```
GNU nano 2.5.3          File: .htaccess

Options +FollowSymlinks
RewriteEngine on
RewriteRule ^wp-content/uploads/(.*)$ http://d2jmrva8tfzxcq.cloudfront.net/$1 [S]

# BEGIN WordPress

# END WordPress
```

# Step 3 – Rewrite Images to S3 / CF



Update with your Cloudfront distribution ID:

The screenshot shows the AWS CloudFront Distributions page. On the left, there's a sidebar with 'Distributions' selected. The main area is titled 'CloudFront Distributions' and shows a table with one row. The row contains: Delivery Method (Web), ID (E2T4SLV73HP8AK), Domain Name (d3k8zs9wa2jq1w.cloudfront.net), and Comment (aws). There are also 'Delete', 'Enable', and 'Disable' buttons at the top of the table.

```
GNU nano 2.5.3          File: .htaccess          Modified

Options +FollowSymlinks
RewriteEngine on
rewriteRule ^wp-content/uploads/(.*)$ http://d3k8zs9wa2jq1w.cloudfront.net/$1 [$.htaccess

# BEGIN WordPress

# END WordPress
```

CTRL + X  
Y  
Enter

# Step 3 – Rewrite Images to S3 / CF



Change Permissions, then Restart Apache Webserver

```
$ sudo chmod 644 .htaccess  
$ sudo service httpd restart
```

A screenshot of a terminal window titled "Downloads — ec2-user@ip-172-31-15-213:/var/www/html — ssh ec2-user@54.2...". The window shows the following command sequence:

```
[ec2-user@ip-172-31-15-213 html]$ sudo nano .htaccess  
[ec2-user@ip-172-31-15-213 html]$ sudo chmod 644 .htaccess  
[ec2-user@ip-172-31-15-213 html]$ sudo service httpd restart  
Stopping httpd: [OK]  
Starting httpd: [OK]  
[ec2-user@ip-172-31-15-213 html]$
```

The "OK" status messages are highlighted in green.

# Step 3 – Rewrite Images to S3 / CF



Check re-writing:

A screenshot of a WordPress website at [wplab.awsokc.com](http://wplab.awsokc.com). The page shows a post titled "AWSOKC – First Post!" from October 5, 2017. The post content includes the text "I sure hope this lab is working well for you!" and a CloudFront image of the AWS OKC logo. A context menu is open over the logo image, with the "Copy Image Address" option highlighted.

wplab.awsokc.com  
Just another WordPress site

POSTS

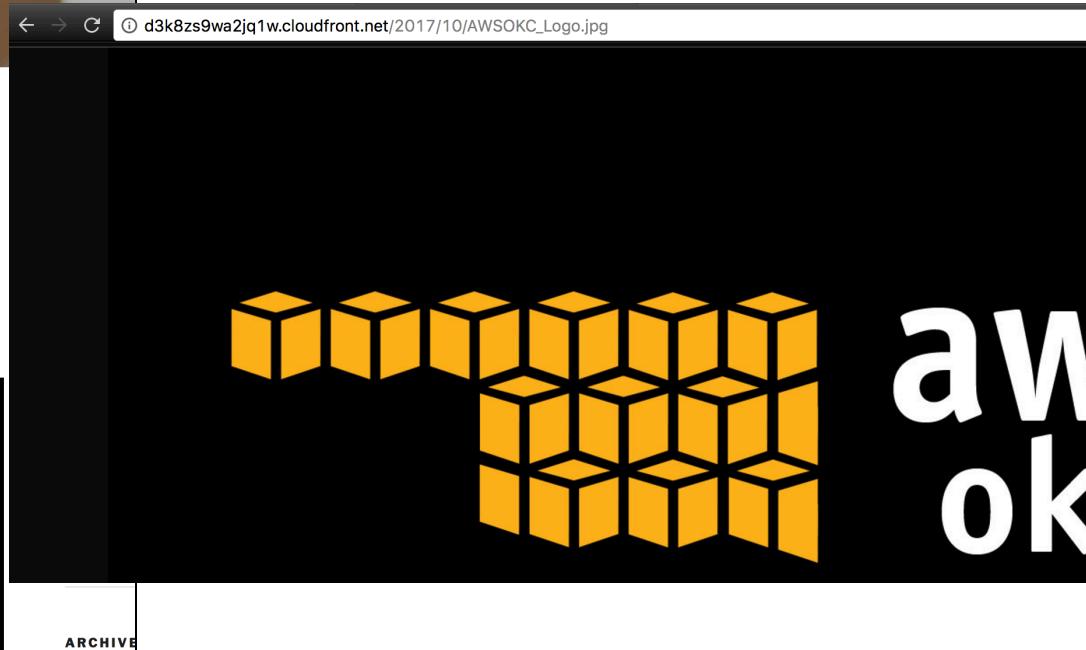
OCTOBER 5, 2017

AWSOKC – First Post!

I sure hope this lab is working well for you!

Open Image in New Tab  
Save Image As...  
Copy Image  
**Copy Image Address**  
Search Google for Image  
Inspect

ARCHIVE



# Step 3 – Rewrite Images to S3 / CF



Now automate the Sync command:

```
$ cd /etc/  
$ sudo nano crontab
```

The screenshot shows a terminal window titled "Downloads — ec2-user@ip-172-31-15-213:/etc — ssh ec2-user@54.221.110.158 -i awslab2017key.pem...". The window contains the following text:

```
SHELL=/bin/bash  
PATH=/sbin:/bin:/usr/sbin:/usr/bin  
MAILTO=root  
HOME=/  
  
# For details see man 4 crontabs  
  
# Example of job definition:  
# .----- minute (0 - 59)  
# | .----- hour (0 - 23)  
# | | .----- day of month (1 - 31)  
# | | | .----- month (1 - 12) OR jan,feb,mar,apr ...  
# | | | | .----- day of week (0 - 6) (Sunday=0 or 7) OR sun,mon,tue,wed,thu,fri,sat  
# | | | | | * user-name command to be executed  
*/5 * * * * root aws s3 sync --delete /var/www/html/wp-content/uploads/ s3://awsokc-wpmedia
```

# Step 3 – Rewrite Images to S3 / CF



Add Cron Jobs:

```
*/5 * * * * root aws s3 sync --delete /var/www/html/wp-content/uploads/ s3://awsokc-wpmedia
*/5 * * * * root aws s3 sync --delete /var/www/html s3://awsokc-wpcode
```

```
GNU nano 2.5.3          File: crontab          Modified

SHELL=/bin/bash
PATH=/sbin:/bin:/usr/sbin:/usr/bin
MAILTO=root
HOME=/

# For details see man 4 crontabs

# Example of job definition:
# .----- minute (0 - 59)
# | .----- hour (0 - 23)
# | | .----- day of month (1 - 31)
# | | | .---- month (1 - 12) OR jan,feb,mar,apr ...
# | | | | .--- day of week (0 - 6) (Sunday=0 or 7) OR sun,mon,tue,wed,thu,fri,sat
# | | | |
# * * * * * user-name command to be executed
*/5 * * * * root aws s3 sync --delete /var/www/html/wp-content/uploads/ s3://awsokc-wpmedia
*/5 * * * * root aws s3 sync --delete /var/www/html s3://awsokc-wpcode
```

CTRL + X  
Y  
Enter

# Step 3 – Rewrite Images to S3 / CF



Restart Cron to force the jobs to rerun:

```
$ sudo service crond restart
```

```
[ec2-user@ip-172-31-15-213 etc]$ sudo service crond restart
Stopping crond:                                     [  OK  ]
Starting crond:                                     [  OK  ]
[ec2-user@ip-172-31-15-213 etc]$
```

# Step 3 – Rewrite Images to S3 / CF



Create file and test sync:

A screenshot of a web browser window showing the WordPress Media Library at the URL 54.221.110.158/wp-admin/upload.php. The left sidebar shows the navigation menu with 'Media' selected. The main area displays a single media item titled 'aws okc', which is a black square icon containing yellow 3D cubes and the text 'aws okc'. Below the image are standard WordPress media library controls: a grid view icon, dropdown menus for 'All media items' and 'All dates', and a 'Bulk Select' button.

A screenshot of a web browser window showing the WordPress Media Library at the URL 54.221.110.158/wp-admin/media.php. The left sidebar shows the navigation menu with 'Media' selected. The main area displays a media item titled 'aws okc', which is a black square icon containing yellow 3D cubes and the text 'aws okc'. The interface includes a 'Media Library' header with an 'Add New' button, a large dashed rectangular area for dropping files, and standard WordPress media library controls below the image.

# Step 3 – Rewrite Images to S3 / CF



Restart Cron:

```
$ sudo service crond restart
```

A screenshot showing two windows side-by-side. The left window is a terminal session on an EC2 instance, displaying the command 'sudo service crond restart' and its output. The right window is a web browser displaying the 'Overview' page of an Amazon S3 bucket named 'awsokc-wpmedia'. The bucket contains three objects: 'AWSOKC\_Logo.jpg', 'Oct 4, 2017 9:30:10 PM', '66.0 KB', 'Standard'; and 'SymphonyTalent-Logo-v.2.22.17.png', 'Oct 4, 2017 10:20:02 PM', '2.6 KB', 'Standard'.

```
[ec2-user@ip-172-31-15-213 html]$ sudo service crond restart
Stopping crond: [ OK ]
Starting crond: [ OK ]
[ec2-user@ip-172-31-15-213 html]$ 
```

Amazon S3 > awsokc-wpmedia / 2017 / 10

Name	Last modified	Size	Storage class
AWSOKC_Logo.jpg	Oct 4, 2017 9:30:10 PM	66.0 KB	Standard
SymphonyTalent-Logo-v.2.22.17.png	Oct 4, 2017 10:20:02 PM	2.6 KB	Standard

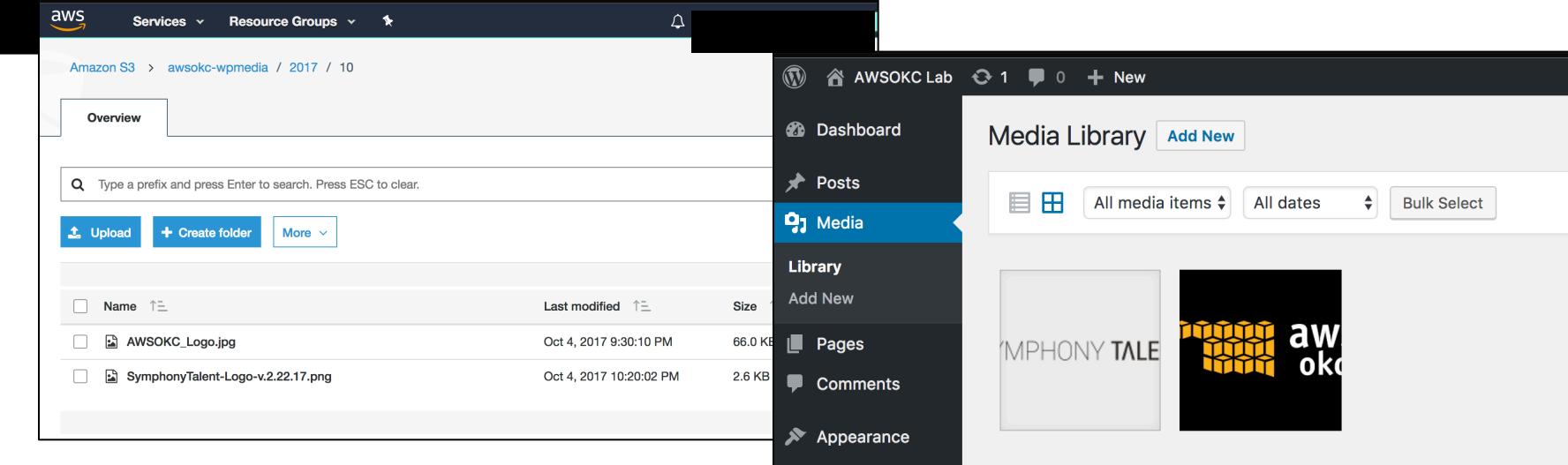
# Step 3 – Rewrite Images to S3 / CF



Restart Cron:

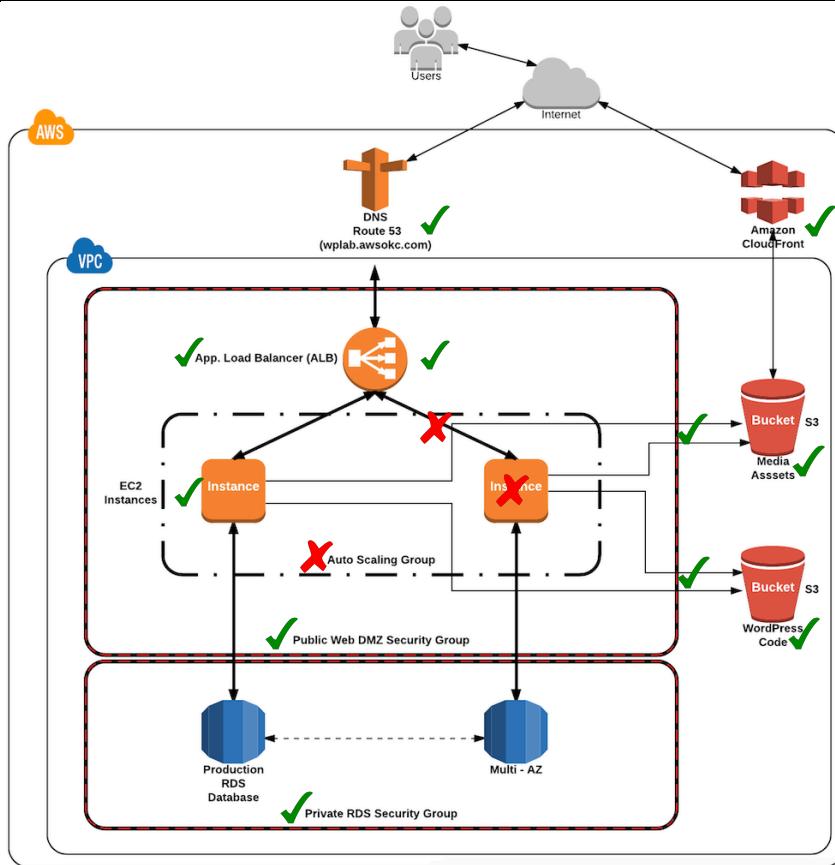
```
$ sudo service crond restart
```

```
[ec2-user@ip-172-31-15-213 html]$ sudo service crond restart
Stopping crond: [OK]
Starting crond: [OK]
[ec2-user@ip-172-31-15-213 html]$
```



The image displays two screenshots side-by-side. On the left is the AWS S3 console showing the 'Media' folder for the 'awsokc-wpmedia' bucket. It lists two files: 'AWSOKC\_Logo.jpg' (uploaded on Oct 4, 2017 at 9:30:10 PM) and 'SymphonyTalent-Logo-v.2.22.17.png' (uploaded on Oct 4, 2017 at 10:20:02 PM). On the right is the WordPress 'Media Library' page for the 'AWSOKC Lab' site. The sidebar menu is visible, showing 'Media' selected. The media library interface includes a search bar, upload buttons ('Upload', '+ Create folder'), and filters for 'All media items' and 'All dates'. Two media items are previewed: a thumbnail for 'AWSOKC\_Logo.jpg' and a larger preview for 'SymphonyTalent-Logo-v.2.22.17.png'.

# Status Check



# Step 4 – Creating our AMI



First remove our EC2 instance from the ELB target group:

The screenshot shows the AWS EC2 Dashboard with the 'Target Groups' section selected. A modal window titled 'Remove' is open, showing the details of the instance being deregistered.

**Main View (EC2 Target Groups):**

- Create target group** button
- Actions** dropdown
- Filter:** Search input
- Targets** table:
  - Name: MyWebServers
  - Port: 80
  - Protocol: HTTP
  - Target type: instance
  - VPC ID: vpc-9aaef8fe
- Target group:** MyWebServers
- Targets** tab (selected)
- Description: The load balancer starts routing requests to a newly registered target as soon as the registration process completes and the target passes the initial health checks. If demand on your targets increases, you can register additional targets. If demand on your targets decreases, you can deregister targets.
- Edit** button
- Registered targets** table:
  - Instance ID: i-09615ff0f36d880eb
  - Name: MyEC2WebServer
  - Port: 80
- Availability Zones** table:
  - Availability Zone: Target count

**Modal Window (Remove):**

Instance	Name	Port	State
i-09615ff0f36d880eb	MyEC2WebServer	80	running

**Buttons:**

- Save button

# Step 4 – Creating our AMI



Discuss need for a Write node in the Architecture in addition to the Web Nodes / ASG

# Step 4 – Creating our AMI



Reconfigure the Cron jobs to Sync down to the EC2 instance instead of up.  
(Make the Web Nodes read nodes – not write nodes)

```
$ cd /etc/  
$ sudo nano crontab
```

The screenshot shows a terminal window titled "Downloads — ec2-user@ip-172-31-15-213:/etc — ssh ec2-user@54.221.110.158 -i awslab2017key.pem...". The window displays the contents of the crontab file being edited with the GNU nano 2.5.3 editor. The terminal command history at the bottom shows the user navigating to /etc and opening crontab for editing.

```
SHELL=/bin/bash
PATH=/sbin:/bin:/usr/sbin:/usr/bin
MAILTO=root
HOME=/

# For details see man 4 crontabs

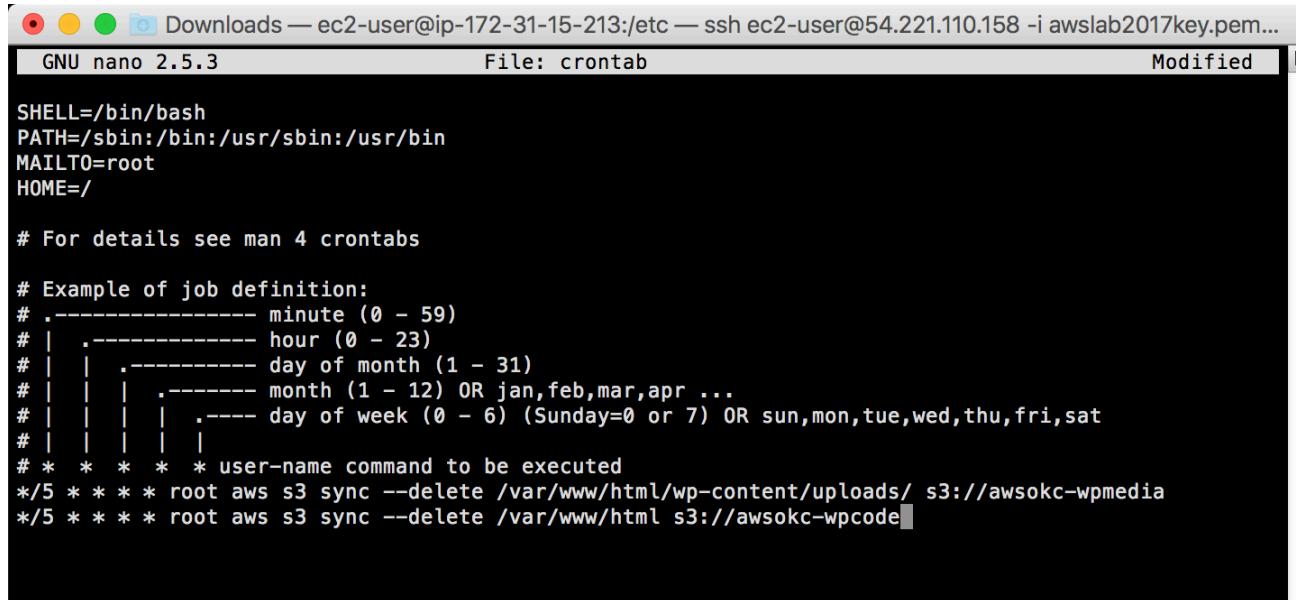
# Example of job definition:
# .----- minute (0 - 59)
# | .---- hour (0 - 23)
# | | .--- day of month (1 - 31)
# | | | .-- month (1 - 12) OR jan,feb,mar,apr ...
# | | | | .-- day of week (0 - 6) (Sunday=0 or 7) OR sun,mon,tue,wed,thu,fri,sat
# |
# * * * * * user-name command to be executed
*/5 * * * * root aws s3 sync --delete /var/www/html/wp-content/uploads/ s3://awsokc-wpmedia
```

# Step 4 – Creating our AMI



Add Cron Jobs:

```
*/5 * * * * root aws s3 sync --delete s3://awsokc-wpmedia /var/www/html/wp-content/uploads/  
*/5 * * * * root aws s3 sync --delete s3://awsokc-wpcode /var/www/html
```



A screenshot of a terminal window titled "Downloads — ec2-user@ip-172-31-15-213:/etc — ssh ec2-user@54.221.110.158 -i awslab2017key.pem...". The window shows the contents of the crontab file being edited with the nano text editor. The terminal title bar includes icons for red, yellow, green, and blue circles, and a file icon.

```
SHELL=/bin/bash
PATH=/sbin:/bin:/usr/sbin:/usr/bin
MAILTO=root
HOME=/

# For details see man 4 crontabs

# Example of job definition:
# .----- minute (0 - 59)
# | .----- hour (0 - 23)
# | | .----- day of month (1 - 31)
# | | | .---- month (1 - 12) OR jan,feb,mar,apr ...
# | | | | .--- day of week (0 - 6) (Sunday=0 or 7) OR sun,mon,tue,wed,thu,fri,sat
# | | | |
# * * * * * user-name command to be executed
*/5 * * * * root aws s3 sync --delete /var/www/html/wp-content/uploads/ s3://awsokc-wpmedia
*/5 * * * * root aws s3 sync --delete /var/www/html s3://awsokc-wpcode
```

CTRL + X  
Y  
Enter

# Step 4 – Creating our AMI



Restart Cron to force the jobs to rerun:

```
$ sudo service crond restart
```

```
[ec2-user@ip-172-31-15-213 etc]$ sudo service crond restart
Stopping crond: [OK]
Starting crond: [OK]
[ec2-user@ip-172-31-15-213 etc]$
```

# Step 4 – Creating our AMI



Now stop the EC2 instance and create the AMI:

The image shows two screenshots of the AWS Management Console illustrating the process of stopping an instance and creating an AMI.

**Top Screenshot:** The Actions menu is open for an instance named "MyEC2Web...". The "Stop" option is highlighted with a red box. The instance status is shown as "running".

Column 1	Column 2
Name	i-09615ff0f

**Bottom Screenshot:** The instance is now listed as "stopped" in the instance list. The Actions menu is open again, and the "Create Image" option is highlighted with a red box. A tooltip below the menu item says "Bundle Instance (instance store AMI)".

Column 1	Column 2
us-east-1b	stopped

# Step 4 – Creating our AMI



## Give Image Name

**Create Image**

Instance ID	i-09615ff0f36d880eb
Image name	AWSOKC-WP-Image
Image description	AWSOKC-WP-Image
No reboot	<input type="checkbox"/>

**Instance Volumes**

Volume Type	Device	Snapshot	Size (GiB)	Volume Type	IOPS	Throughput (MB/s)	Delete on Termination	Encrypted
Root	/dev/xvda	snap-080eb3cb2eda29974	8	General Purpose SSD (GP2)	100 / 3000	N/A	<input checked="" type="checkbox"/>	Not Encrypted

**Add New Volume**

Total size of EBS Volumes: 8 GiB  
When you create an EBS image, an EBS snapshot will also be created for each of the above volumes.

**Create Image**

**Create Image**

✓ Create Image request received.  
View pending image ami-6516d21f

Any snapshots backing your new EBS image can be managed on the [snapshots screen](#) after successful image creation.

**Close**

# Step 4 – Creating our AMI



AMI Creation:

The screenshot shows the AWS EC2 Dashboard. On the left, a sidebar menu includes EC2 Dashboard, Events, Tags, Reports, Limits, INSTANCES (with sub-options Instances, Spot Requests, Reserved Instances, Scheduled Instances, Dedicated Hosts), IMAGES (with sub-option AMIs selected), and ELASTIC BLOCK STORE. The main content area has tabs for Launch and Actions. A search bar at the top says "Owned by me" and "search : AWSOKC". A table lists one item: Name: AWSOKC-WP..., AMI ID: ami-6516d21f, Source: 534106927381..., Owner: 534106927381, Visibility: Private, Status: pending, Creation Date: October 4, 2017 at 11:11. To the right of the table is a large callout box with the word "Status" and "available" in green text.

Name	AMI Name	AMI ID	Source	Owner	Visibility	Status	Creation Date
AWSOKC-WP...	ami-6516d21f	534106927381/...	534106927381	Private	pending	pending	October 4, 2017 at 11:11

Status  
available

# Step 4 – Creating our AMI



Now that we have our AMI – you can terminate the stopped instance.

A screenshot of the AWS EC2 Dashboard. The left sidebar shows navigation links like EC2 Dashboard, Events, Tags, Reports, Limits, INSTANCES, Instances (which is selected), Spot Requests, Reserved Instances, Scheduled Instances, and Dedicated Hosts. The main area displays a list of instances with one entry: "MyEC2Web..." (Instance ID: i-0c466e80). Below the instance list, it says "Instance: i-0c466e80ce3fc08bed (MyEC2WebServer) Private IP: 172.31.7.82". Above the instance list is a search bar with "search : MyEC2" and a "Connect" button. To the right of the search bar is an "Actions" button with a dropdown menu open. The dropdown menu includes options: Connect, Get Windows Password, Launch More Like This, Instance State (which is highlighted in orange), Instance Settings, Image, Networking, CloudWatch Monitoring, Start, Stop, Reboot, and Terminate. The "Terminate" option is also highlighted in orange. The status of the instance is shown as "stopped" with a red circle icon. The top navigation bar includes "Services", "Resource Groups", "N. Virginia", and "S".

# Step 5 – Creating our ASG / LC



Create Auto-Scaling Group / Launch Config:

The screenshot shows two views of the AWS Auto Scaling service. On the left, the main 'Welcome to Auto Scaling' page is displayed. A red box highlights the 'Launch Configurations' link under the 'AUTO SCALING' section in the sidebar. On the right, a modal window titled 'Create Auto Scaling Group' is open, also with a red box highlighting the 'Launch Configurations' link in its sidebar.

**Welcome to Auto Scaling**

You can use Auto Scaling to manage Amazon EC2 capacity automatically. You can launch new instances to support your application, operate a healthy group of instances, and scale down when you don't need as many instances.

**Auto Scaling Group: 1**

**Create Auto Scaling group**

Note: To create your Auto Scaling groups in a different region, select your region from the Region dropdown in the top navigation bar.

**Benefits of Auto Scaling**

**Reusable Instance Templates**

Provision instances based on a reusable template you define, called a launch configuration.

**Automated Provisioning**

Keep your Auto Scaling group healthy and balanced, whether you need one instance or 1,000.

**Create Auto Scaling Group**

To create an Auto Scaling group, you will first need to choose a template that your Auto Scaling group will use when it launches instances for you, called a launch configuration. Choose a launch configuration or create a new one, and then apply it to your group.

Later, if you want to use a different template, you can create another launch configuration and apply it to this group, even if you already have instances running in it. Using this method, you can update the software that your group uses when it launches new instances.

Create a new launch configuration

Create an Auto Scaling group from an existing launch configuration

# Step 5 – Creating our ASG / LC



Select your new AMI:

The screenshot shows the AWS Lambda console interface for creating a new launch configuration. The top navigation bar includes the AWS logo, Services dropdown, Resource Groups dropdown, a bell icon, a blue progress bar, N. Virginia region selector, and Support dropdown.

The main steps are numbered 1. Choose AMI through 6. Review. Step 1 is currently selected.

The title "Create Launch Configuration" is displayed, along with a sub-instruction: "An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. You can select an AMI provided by AWS, our user community, or the AWS Marketplace; or you can select one of your own AMIs."

A search bar at the top right contains the text "awsokc". Below it, a table lists AMIs under the heading "My AMIs".

AMIs	Action
 AWSOKC-WP-Image - ami-a836f2d2 AWSOKC-WP-Image Root device type: ebs Virtualization type: hvm Owner: 534106927381	<button>Select</button> 64-bit

On the left sidebar, there are filters for Quick Start, My AMIs (selected), AWS Marketplace, Community AMIs, Ownership (Owned by me checked, Shared with me unchecked), and Architecture.

# Step 5 – Creating our ASG / LC



Keep size as t2.micro:

The screenshot shows the AWS EC2 'Create Launch Configuration' wizard at step 2. The top navigation bar includes 'Services', 'Resource Groups', 'N. Virginia', and 'Support'. Below the navigation, tabs for steps 1 through 6 are visible, with '2. Choose Instance Type' being the active tab. A descriptive text block about Amazon EC2 instance types follows. Filter options 'All instance types' and 'Current generation' are present. A note states 'Currently selected: t2.micro (Variable ECUs, 1 vCPUs, 2.5 GHz, Intel Xeon Family, 1 GiB memory, EBS only)'. A table lists various instance types under 'General purpose': t2.nano, t2.micro (selected), t2.small, t2.medium, and t2.large. The t2.micro row is highlighted with a blue border and has a green banner indicating it is 'Free tier eligible'. The table columns include Family, Type, vCPUs, Memory (GiB), Instance Storage (GB), EBS-Optimized Available, and Network Performance. At the bottom are 'Cancel', 'Previous', and 'Next: Configure details' buttons.

	Family	Type	vCPUs	Memory (GiB)	Instance Storage (GB)	EBS-Optimized Available	Network Performance
<input type="checkbox"/>	General purpose	t2.nano	1	0.5	EBS only	-	Low to Moderate
<input checked="" type="checkbox"/>	General purpose	t2.micro Free tier eligible	1	1	EBS only	-	Low to Moderate
<input type="checkbox"/>	General purpose	t2.small	1	2	EBS only	-	Low to Moderate
<input type="checkbox"/>	General purpose	t2.medium	2	4	EBS only	-	Low to Moderate
<input type="checkbox"/>	General purpose	t2.large	2	8	EBS only	-	Low to Moderate

# Step 5 – Creating our ASG / LC



Give it a name and IAM role:

The screenshot shows the "Create Launch Configuration" step in the AWS Launch Configuration wizard. The "Configure details" tab is selected. The "Name" field contains "AWSOKC-WPlab-LC", and the "IAM role" field contains "S3-Admin-Access". Both fields are highlighted with red boxes. Below the fields, there are sections for "Purchasing option" (checkbox for Request Spot Instances) and "Monitoring" (checkbox for CloudWatch detailed monitoring). A "Advanced Details" section is expanded, containing a note about changing launch configurations. At the bottom, there are "Cancel", "Previous", "Skip to review", and "Next: Add Storage" buttons.

1. Choose AMI 2. Choose Instance Type 3. Configure details 4. Add Storage 5. Configure Security Group 6. Review

Create Launch Configuration

Name: AWSOKC-WPlab-LC

Purchasing option:  Request Spot Instances

IAM role: S3-Admin-Access

Monitoring:  Enable CloudWatch detailed monitoring

Advanced Details

Later, if you want to use a different launch configuration, you can create a new one and apply it to any Auto Scaling group. Existing launch configurations cannot be edited.

Cancel Previous Skip to review Next: Add Storage

# Step 5 – Creating our ASG / LC



Use default storage size:

The screenshot shows the AWS EC2 Launch Instance wizard at Step 4: Add Storage. The top navigation bar includes 'Services' (dropdown), 'Resource Groups' (dropdown), a bell icon, a dropdown for 'N. Virginia', and 'Support' (dropdown). Below the navigation, a progress bar shows steps 1 through 6: Choose AMI, Choose Instance Type, Configure details, Add Storage (highlighted in orange), Configure Security Group, and Review.

**Create Launch Configuration**

Your instance will be launched with the following storage device settings. You can attach additional EBS volumes and instance store volumes to your instance, or edit the settings of the root volume. You can also attach additional EBS volumes after launching an instance, but not instance store volumes.

<https://docs.aws.amazon.com/console/ec2/launchinstance/storage> about storage options in Amazon EC2.

Volume Type	Device	Snapshot	Size (GiB)	Volume Type	IOPS	Throughput	Delete on Termination	Encrypted
Root	/dev/xvda	snap-01597c91a34323e95	8	General Purpose (SSD)	100 / 3000	N/A	<input checked="" type="checkbox"/>	No

**Add New Volume**

**Free tier eligible customers can get up to 30 GB of EBS storage.** [Learn more](#) about free usage tier eligibility and usage restrictions.

At the bottom are buttons for 'Cancel', 'Previous', 'Skip to review' (highlighted in blue), and 'Next: Configure Security Group'.

# Step 5 – Creating our ASG / LC



Use the Web-DMZ security group:

The screenshot shows the AWS Launch Configuration creation process at step 5: Configure Security Group. The top navigation bar includes the AWS logo, Services dropdown, Resource Groups dropdown, a bell icon, a blue circular progress bar, N. Virginia dropdown, and Support dropdown.

The main content area has a breadcrumb trail: 1. Choose AMI, 2. Choose Instance Type, 3. Configure details, 4. Add Storage, 5. Configure Security Group (which is underlined in orange), and 6. Review.

### Create Launch Configuration

A security group is a set of firewall rules that control the traffic for your instance. On this page, you can add rules to allow specific traffic to reach your instance. For example, if you want to set up a web server and allow Internet traffic to reach your instance, add rules that allow unrestricted access to the HTTP and HTTPS ports. You can create a new security group or select from an existing one below. [Learn more](#) about Amazon EC2 security groups.

Assign a security group:  Create a **new** security group  Select an **existing** security group

<input type="checkbox"/>	sg-ce14f9b4	symmetricds-aurora	vpc-9aaef8fe	SymmetricDS Aurora
<input checked="" type="checkbox"/>	sg-cde3b2bd	Web-DMZ2	vpc-9aaef8fe	Web-DMZ2

Inbound rules for sg-cde3b2bd Selected security groups: sg-cde3b2bd.

Type	Protocol	Port Range	Source
HTTP	TCP	80	0.0.0.0/0
SSH	TCP	22	0.0.0.0/0

Buttons at the bottom: Cancel, Previous, and Review (highlighted in blue).

# Step 5 – Creating our ASG / LC



Create:

Screenshot of the AWS Create Launch Configuration wizard, Step 6: Review.

The top navigation bar shows: AWS, Services, Resource Groups, N. Virginia, Support.

The progress bar indicates: 1. Choose AMI, 2. Choose Instance Type, 3. Configure details, 4. Add Storage, 5. Configure Security Group, 6. Review.

**Create Launch Configuration**

Review the details of your launch configuration. You can go back to edit the details of each section before you finish.

**⚠ Improve security of instances launched using your launch configuration, AWSOKC-WPlab-LC. Your security group, Web-DMZ2, is open to the world.**

Your instances may be accessible from any IP address. We recommend that you update your security group rules to allow access from known IP addresses only. You can also open additional ports in your security group to facilitate access to the application or service you're running, e.g., HTTP (80) for web servers. [Edit security groups](#)

**AMI Details** [Edit AMI](#)

**AWSOKC-WP-Image - ami-a836f2d2**  
AWSOKC-WP-Image  
Root device type: ebs Virtualization Type: hvm

**Instance Type** [Edit instance type](#)

Instance Type	ECUs	vCPUs	Memory GiB	Instance Storage (GiB) GiB	EBS-Optimized Available	Network Performance
t2.micro	Variable	1	1	EBS only	-	Low to Moderate

[Cancel](#) [Previous](#) [Create launch configuration](#)

# Step 5 – Creating our ASG / LC



Select your key pair:

**Select an existing key pair or create a new key pair** X

A key pair consists of a **public key** that AWS stores, and a **private key file** that you store. Together, they allow you to connect to your instance securely. For Windows AMIs, the private key file is required to obtain the password used to log into your instance. For Linux AMIs, the private key file allows you to securely SSH into your instance.

Note: The selected key pair will be added to the set of keys authorized for this instance. Learn more about [removing existing key pairs from a public AMI](#).

Choose an existing key pair

Select a key pair

awslab2017key

I acknowledge that I have access to the selected private key file (awslab2017key.pem), and that without this file, I won't be able to log into my instance.

Cancel Create launch configuration

# Step 5 – Creating our ASG / LC



Now we'll create the Auto Scaling Group:

The screenshot shows the "Create Auto Scaling Group" wizard in the AWS Management Console. The current step is "1. Configure Auto Scaling group details".

**Launch Configuration:** AWSOKC-WPlab-LC

**Group name:** AWSOKC-WPlab-ASG (highlighted with a red box)

**Group size:** Start with 2 instances (highlighted with a red box)

**Network:** vpc-9aaef8fe (172.31.0.0/16) (default)

**Subnet:** A dropdown menu showing several subnet options, one of which is highlighted with a red box.

- subnet-6ab70966(172.31.80.0/20) | Default in us-east-1f
- subnet-11ef7b74(172.31.64.0/20) | Default in us-east-1d
- subnet-5955ee2f(172.31.0.0/20) | Default in us-east-1b
- subnet-43b37b69(172.31.48.0/20) | Default in us-east-1a
- subnet-7362054e(172.31.32.0/20) | Default in us-east-1e
- subnet-2cd60b74(172.31.16.0/20) | Default in us-east-1c

**Next: Configure scaling policies**

# Step 5 – Creating our ASG / LC



Configure advanced options:

The screenshot shows the "Create Auto Scaling Group" wizard at step 1. The "Advanced Details" section is expanded. Several fields have red boxes around them: the "Target Groups" input field containing "MyWebServers", the "Health Check Type" radio button group with "ELB" selected, and the "Health Check Grace Period" input field containing "10". At the bottom right are "Cancel" and "Next: Configure scaling policies" buttons.

1. Configure Auto Scaling group details    2. Configure scaling policies    3. Configure Notifications    4. Configure Tags    5. Review

Create Auto Scaling Group

Cancel and Exit

Advanced Details

Load Balancing  Receive traffic from one or more load balancers [Learn about Elastic Load Balancing](#)

Classic Load Balancers

Target Groups  MyWebServers

Health Check Type  ELB  EC2

Health Check Grace Period  10 seconds

Monitoring Amazon EC2 Detailed Monitoring metrics, which are provided at 1 minute frequency, are not enabled for the launch configuration AWSOKC-WPlab-LC. Instances launched from it will use Basic Monitoring metrics, provided at 5 minute frequency.

Instance Protection

Cancel Next: Configure scaling policies

# Step 5 – Creating our ASG / LC



Set scaling policies:

The screenshot shows the AWS Auto Scaling Group creation wizard at Step 2: Configure scaling policies. The page title is "Create Auto Scaling Group". It has two radio button options: "Keep this group at its initial size" (unchecked) and "Use scaling policies to adjust the capacity of this group" (checked). Below this, it says "Scale between 2 and 4 instances. These will be the minimum and maximum size of your group." A modal window titled "Increase Group Size" is open. Inside the modal, there are fields for "Name" (set to "Increase Group Size"), "Execute policy when" (set to "No alarm selected"), and "Take the action" (set to "Add 0 instances"). A red box highlights the "Add new alarm" button, which has a circular icon with a letter "C". At the bottom of the modal, there are sections for "Instances need" (set to "300 seconds to warm up after each step") and "Create a simple scaling policy". At the very bottom of the page, there are navigation buttons: "Cancel", "Previous", "Review" (highlighted in blue), and "Next: Configure Notifications".

# Step 5 – Creating our ASG / LC



Set scaling policies:

### Create Alarm

You can use CloudWatch alarms to be notified automatically whenever metric data reaches a level you define.

To edit an alarm, first choose whom to notify and then define when the notification should be sent.

**Send a notification to:** ScaleUpEvent [cancel](#)

**With these recipients:** nathan@symphonytalent.com

**Whenever:** Average of CPU Utilization

**Is:**  $\geq$  75 Percent

**For at least:** 1 consecutive period(s) of 5 Minutes

**Name of alarm:** awsec2-AWSOKC-WPlab-ASG-CPU-Utilization

**CPU Utilization Percent**

The chart displays CPU Utilization Percent on the Y-axis (0 to 60) against time on the X-axis (10/5 02:00 to 10/5 06:00). A single data series is shown as a solid red line, representing the average CPU utilization for the ASG. The value remains constant at 75% throughout the observed period.

AWSOKC-WPlab-ASG

[Cancel](#) [Create Alarm](#)

# Step 5 – Creating our ASG / LC



Set scaling policies:

### Create Alarm

You can use CloudWatch alarms to be notified automatically whenever metric data reaches a level you define.

To edit an alarm, first choose whom to notify and then define when the notification should be sent.

**Send a notification to:**  [cancel](#)

**With these recipients:**

**Whenever:**  Average  of

**Is:**  >=   Percent

**For at least:**  consecutive period(s) of  5 Minutes

**Name of alarm:**

**CPU Utilization** Percent

The chart displays CPU Utilization Percent over time. The Y-axis ranges from 0 to 25. The X-axis shows dates: 10/5, 02:00, 04:00, and 06:00. A single red horizontal line is drawn at the 25% utilization level. Below the chart, a legend indicates a blue square represents "AWSOKC-WPlab-ASG".

Date	CPU Utilization (%)
10/5 02:00	25
10/5 04:00	25
10/5 06:00	25

# Step 5 – Creating our ASG / LC



Set scaling policies:

Screenshot of the AWS Auto Scaling Group creation process, specifically Step 5: Set scaling policies.

The top navigation bar shows: AWS Services Resource Groups N. Virginia Support.

The current step is 2. Configure scaling policies.

**Create Auto Scaling Group**

**Increase Group Size**

Name: Increase Group Size

Execute policy when: awsec2-AWSOKC-WPlab-ASG-CPU-Utilization [Edit](#) [Remove](#)  
breaches the alarm threshold: CPUUtilization >= 75 for 300 seconds  
for the metric dimensions AutoScalingGroupName = AWSOKC-WPlab-ASG

Take the action: Add 2 instances when 75 <= CPUUtilization < +infinity  
[Add step](#)

Instances need: 300 seconds to warm up after each step

[Create a simple scaling policy](#)

**Decrease Group Size**

Name: Decrease Group Size

Execute policy when: awsec2-AWSOKC-WPlab-ASG-High-CPU-Utilization [Edit](#) [Remove](#)  
breaches the alarm threshold: CPUUtilization >= 25 for 300 seconds  
for the metric dimensions AutoScalingGroupName = AWSOKC-WPlab-ASG

Take the action: Remove 1 instances when 25 <= CPUUtilization < +infinity  
[Add step](#)

[Create a simple scaling policy](#)

Buttons at the bottom: Cancel Previous Review Next: Configure Notifications

# Step 5 – Creating our ASG / LC



Set scaling policies:

The screenshot shows the 'Create Auto Scaling Group' wizard at step 3, 'Configure Notifications'. The top navigation bar includes 'Services', 'Resource Groups', 'N. Virginia', and 'Support'. The steps are: 1. Configure Auto Scaling group details, 2. Configure scaling policies, 3. Configure Notifications (highlighted in orange), 4. Configure Tags, 5. Review.

**Create Auto Scaling Group**  
Configure your Auto Scaling group to send notifications to a specified endpoint, such as an email address, whenever a specified event takes place, including: successful launch of an instance, failed instance launch, instance termination, and failed instance termination.

If you created a new topic, check your email for a confirmation message and click the included link to confirm your subscription. Notifications can only be sent to confirmed addresses.

**Send a notification to:** ASG-Notifications [use existing topic](#) ×

**With these recipients:**

**Whenever instances:**

- launch
- terminate
- fail to launch
- fail to terminate

[Add notification](#)

[Cancel](#) [Previous](#) **Review** [Next: Configure Tags](#)

# Step 5 – Creating our ASG / LC



Set scaling policies:

The screenshot shows the 'Create Auto Scaling Group' wizard at step 4: 'Configure Tags'. The top navigation bar includes 'Services', 'Resource Groups', 'N. Virginia', and 'Support'. Below the navigation, five steps are listed: 1. Configure Auto Scaling group details, 2. Configure scaling policies, 3. Configure Notifications, 4. Configure Tags (which is the current step), and 5. Review.

**Create Auto Scaling Group**

A tag consists of a case sensitive key-value pair that you can use to identify your group. For example, you could define a tag with Key = Environment and Value = Production. You can optionally choose to apply these tags to instances in the group when they launch. [Learn more](#).

Key	Value	Tag New Instances
Name	PROD-WEBSERVER-WPLAB	<input checked="" type="checkbox"/>

**Add tag** 49 remaining

At the bottom right are 'Cancel', 'Previous', and 'Review' buttons.

# Step 5 – Creating our ASG / LC



Set scaling policies:

Screenshot of the AWS Auto Scaling Group creation wizard, Step 5: Review.

The page shows the configuration details for the Auto Scaling Group (ASG) named "AWSOKC-WPlab-ASG".

**Auto Scaling Group Details:**

- Group name: AWSOKC-WPlab-ASG
- Group size: 2
- Minimum Group Size: 2
- Maximum Group Size: 4
- Subnet(s): subnet-6ab70966, subnet-11ef7b74, subnet-5955ee2f, subnet-43b37b69, subnet-7362054e, subnet-2cd60b74
- Load Balancers:
  - Target Groups: MyWebServers
  - Health Check Type: ELB
  - Health Check Grace Period: 10
  - Detailed Monitoring: No
  - Instance Protection: None

**Scaling Policies:**

- Increase Group Size: With alarm = awsec2-AWSOKC-WPlab-ASG-CPU-Utilization; Add 2 instances and 300 seconds for instances to warm up
- Decrease Group Size: With alarm = awsec2-AWSOKC-WPlab-ASG-High-CPU-Utilization; Remove 1 instances

**Notifications:**

- ASG-Notifications (nathan@symphonytalent.com): launch, terminate, fail to launch, fail to terminate

**Tags:**

Cancel Previous Create Auto Scaling group

# Step 5 – Creating our ASG / LC



See new instances coming online:

A screenshot of the AWS EC2 Instances page. The left sidebar shows navigation options like EC2 Dashboard, Events, Tags, Reports, Limits, Instances (which is selected), Spot Requests, Reserved Instances, Scheduled Instances, Dedicated Hosts, Images, AMIs, and Bundle Tasks. The main content area has tabs for Launch Instance, Connect, and Actions. A search bar shows "search : Prod" and an "Add filter" button. Below is a table with columns: Name, Instance ID, Instance Type, Availability Zone, Instance State, Status Checks, Alarm Status, and Public DNS (IPv4). Two instances are listed:

Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS (IPv4)
PROD-WEBSERVER-WPLAB	i-0ced4206e1339b23f	t2.micro	us-east-1c	running	Initializing	None	ec2-34-228-226-122.co...
PROD-WEBSERVER-WPLAB	i-0335bfb0077a19f30	t2.micro	us-east-1d	running	Initializing	None	ec2-52-3-225-94.comp...

A message at the bottom says "Select an instance above" with three small icons.

# Step 5 – Creating our ASG / LC



Once instances are healthy – verify site availability:

Screenshot of the AWS EC2 Dashboard showing two healthy instances:

Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS (IPv4)
PROD-WEBSERVER-WPLAB	i-0ced4206e1339b23f	t2.micro	us-east-1c	running	2/2 checks ...	None	ec2-34-228-226-122.co...
PROD-WEBSERVER-WPLAB	i-0335bf0077a19f30	t2.micro	us-east-1d	running	2/2 checks ...	None	ec2-52-3-225-94.comp...



# Step 6 – Test Availability / Load



## Test 1 – Terminate Instances:

The screenshot shows the AWS EC2 Dashboard with two instances listed:

Name	Instance ID	Type	Zone	Status	Checks	Alarm Status
PROD-WEBSERVER-WPLAB	i-0ced4206e1339b23f	t2.micro	us-east-1c	terminated	0/2	None
PROD-WEBSERVER-WPLAB	i-05b93b42988df27c8	t2.micro	us-east-1a	running	0/2	None
PROD-WEBSERVER-WPLAB	i-0335fb0077a19f30	t2.micro	us-east-1d	running	2/2	None

In the top instance's context menu, the "Terminate" option is highlighted.

# Step 6 – Test Availability / Load



## Test 2 – Testing High Load:

The screenshot shows the AWS EC2 Instances page. The left sidebar navigation includes: EC2 Dashboard, Events, Tags, Reports, Limits, INSTANCES (with Instances selected), Spot Requests, Reserved Instances, Scheduled Instances, Dedicated Hosts, IMAGES (with AMIs selected), Bundle Tasks, ELASTIC BLOCK STORE (with Volumes selected), Snapshots, and NETWORK & SECURITY (with Security Groups selected). The main content area displays a table of instances. The table has columns: Name, Instance ID, Instance Type, Availability Zone, Instance State, Status Checks, Alarm Status, and Public DNS (IPv4). There are three entries:

Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS (IPv4)
PROD-WEBSERVER-WPLAB	i-0ced4206e1339b23f	t2.micro	us-east-1c	terminated		None	
PROD-WEBSERVER-WPLAB	i-05b93b42988df27c8	t2.micro	us-east-1a	running	2/2 checks ...	None	ec2-52-90-91-88.c
PROD-WEBSERVER-WPLAB	i-0335bfb0077a19f30	t2.micro	us-east-1d	running	2/2 checks ...	None	ec2-52-3-225-94.c

Below the table, details for instance i-0335bfb0077a19f30 are shown. The "Description" tab is selected. The instance details are:

Description	Value
Instance ID	i-0335bfb0077a19f30
Instance state	running
Instance type	t2.micro
Elastic IPs	-
Availability zone	us-east-1d
Security groups	Web-DMZ2. view inbound rules
Scheduled events	No scheduled events

On the right side, network information is displayed:

Public DNS (IPv4)	ec2-52-3-225-94.compute-1.amazonaws.com
IPv4 Public IP	52.3.225.94
IPv6 IPs	-
Private DNS	ip-172-31-65-94.ec2.internal
Private IPs	172.31.65.94
Secondary private IPs	-
VPC ID	vpc-9aaef8fe

# Step 6 – Test Availability / Load



SSH and run Stress Testing Tool:

```
$ sudo su
```

```
$ stress --cpu 100
```

A screenshot of a terminal window titled "Downloads — root@ip-172-31-65-94:/home/ec2-user — ssh ec2-user@52.3.225.94 -i awslab2017key....". The terminal shows the following text:

```
LM-140227:Downloads nathan$ ssh ec2-user@52.3.225.94 -i awslab2017key.pem
The authenticity of host '52.3.225.94 (52.3.225.94)' can't be established.
ECDSA key fingerprint is SHA256:tizw58db+fS6IcYrwNGbfLug0FT0JnbBdt2p0mr0rcw.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '52.3.225.94' (ECDSA) to the list of known hosts.
Last login: Thu Oct  5 05:52:36 2017 from 68.229.253.41

      _\   _ )   Amazon Linux AMI
     _\ \_ |__|_ _\

https://aws.amazon.com/amazon-linux-ami/2017.09-release-notes/
[ec2-user@ip-172-31-65-94 ~]$ sudo su
[root@ip-172-31-65-94 ec2-user]# stress --cpu 100
stress: FAIL: [2777] (244) unrecognized option: -cpu
[root@ip-172-31-65-94 ec2-user]# stress --cpu 100
stress: info: [2778] dispatching hogs: 100 cpu, 0 io, 0 vm, 0 hdd
```

# Step 6 – Test Availability / Load

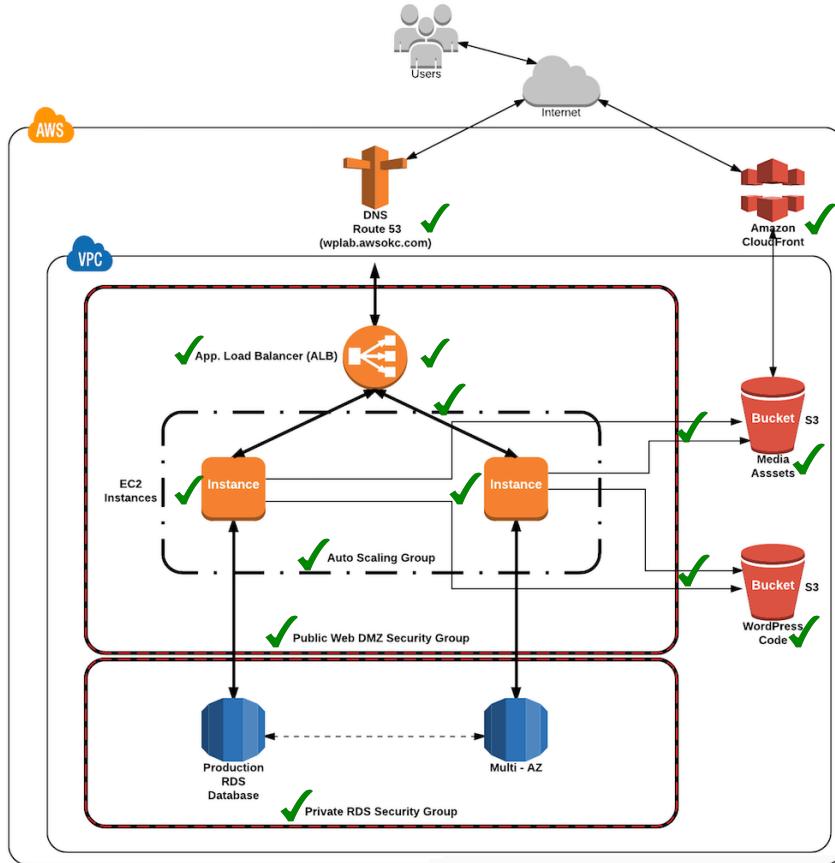


Test 2 – Testing High Load:

The screenshot shows the AWS EC2 Dashboard with the following details:

- Instances:** Five instances are listed, all named "PROD-WEBSERVER-WPLAB".
  - i-0dd516435e8eb8... (us-east-1b): running
  - i-0ced4206e1339b23f (us-east-1c): terminated
  - i-0c30aa300a85ae6f1 (us-east-1e): running
  - i-05b93b42988df27c8 (us-east-1a): running
  - i-0335fb0077a19f30 (us-east-1d): running
- CloudWatch alarms:** No alarms configured.
- CloudWatch metrics:** Basic monitoring. Metrics shown: CPU Utilization (Percent), Disk Reads (Bytes), and Disk Read Operations (Operations). All three metrics show a sharp increase starting around the 1-hour mark.

# Status Check



# Power Down Recommendations:

Don't forget to terminate your resources:

- Auto-Scaling Group and Launch Config
- AMI
- Any Running EC2 Instances
- ALB (**This costs you money!**)
- RDS Instance (**This could cost you money depending on config**)
- CloudFront Distribution
- S3 Buckets
- Security Groups
- IAM Role (S3-Admin-Access)
- Route53 DNS Entries (if you used them)