## REAL World Application: [Summing up]

## A highly available WordPress blog:

The setup:

### Create ec2 to s3 role using IAM

|  |
| --- |
|  |

### From your VPC window, Create two security groups

* 1. Title: **webzone** ,
     1. Ports :
        1. HTTP(80),
        2. HTTPS(443) and
        3. SSH(22)
  2. Tile**: dbzone** ,
     1. Ports :
        1. mysql(3306)

### Goto RDS and Create a MYSql instance with following configuration params

* 1. in **Single Availability Zone**
  2. Instance Class: **t2.micro**
  3. Size : **5GB**
  4. Multi AZ : **No**
  5. Storage: General Purpose
  6. DB Instance Identifier: wordpressdb
  7. Username: wordpressdb
  8. Password : wordpressdb
  9. Publically accessible : **No**
  10. Security Group : **dbzone**

**Once created note the endpoint. In my case it is wordpressdb.c9pbehojgmiz.us-east-1.rds.amazonaws.com:3306**

|  |
| --- |
|  |

### Create a Load Balancer

1. Port : 80
2. Health Check: Defaults and
   * 1. Ping path: /elb.html
     2. Response timeout: 5
     3. HC interval : 30
     4. Unhealthy threshold :4
     5. Healthy threshold :4
3. Security group : **webzone**
4. No instances selected

### Goto Route 53 , Create Hosted Zones and Record sets

1. Create Hosted Zone (using my registered domain name. You should use one of yours)
   1. Zone name: **cloudarchitects.co.in** (same as your registered domain)
2. Name server records need to be updated to your DOMAIN PROVIDER Site(in my case it is GODADDY)
3. Create a Record set
   1. RS name : cloudarchitects.com
   2. Type: A IPV4
   3. Alias Yes
   4. Select your Load balancer
   5. Routing Policy : Simple
   6. Evaluate target Health Chk:No
4. Create another Record set (optional)
   1. RS name :www. cloudarchitects.com
   2. Type: A IPV4
   3. Alias Yes
   4. Select your Load balancer
   5. Routing Policy : Simple
   6. Evaluate target Health Chk:No

### Goto S3 , Create buckets

1. Create two buckets (you may change the names if below are not available)
   1. Name: wpcodejpa
   2. Name : wpcdnjpa

### Go to CLOUDFRONT , Create distribution

* 1. Create a cloud front Distribution:
     1. Type : Web
     2. Origin Domain Name: select “**wpcdnjpa**” bucket name
     3. Restrict bucket access **: Yes** (forcing to use Cloudfront distribution url instead of S3 for uploads etc)
     4. **Origin Access identity : Create a New Identity**
     5. **Grant Read Permissions on the bucket : Yes**
     6. Rest all defaults
  2. Click “Create Distribution”
  3. It will take some time

|  |
| --- |
|  |
|  |

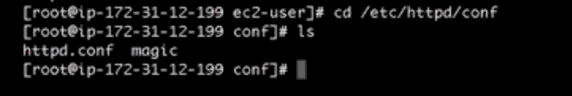
### Create an Amazon Linux instance

* 1. AMI : Amazon Linux
  2. Type t2.micro
  3. Role ec2-s3
  4. Security gp : webzone

### Add above EC2 instance to the Load Balancer you created in step 4

### Connect to the instance using Putty and public IP

* 1. Install Apache server and mysql client (yum install httpd php php-mysql -y)
  2. yum update –y
  3. go to apache configuration file and change
     1. cd /etc/httpd/conf/



* + 1. Duplicated httpd.conf
       1. cp httpd.conf myhttpd.conf
    2. open httpd.conf nano httpd.conf
       1. change the following
          1. Search for (CTRL W) a line starting with “Document Root”

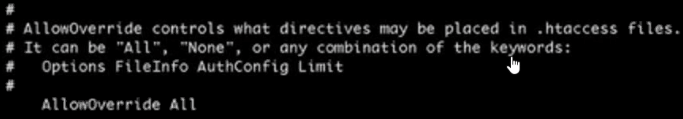
It should have /var/www/html

* + - * 1. Search for (CTRL W) a line starting with “**AllowOverride Controls**”

Change **AllowOverride None to**

**AllowOverride All**

**(this is to enable apache to sense the updates/uploads done thro’ a Cloud front distribution url)**

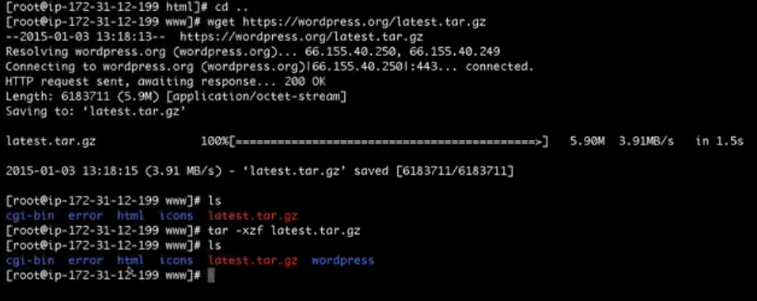


* + - * 1. Start apache (service httpd start)
    1. Go to /var/www/html and create a file called “elb.html” and store a word “great success” in it. This is for load balancer health check. (as done in Step 4)

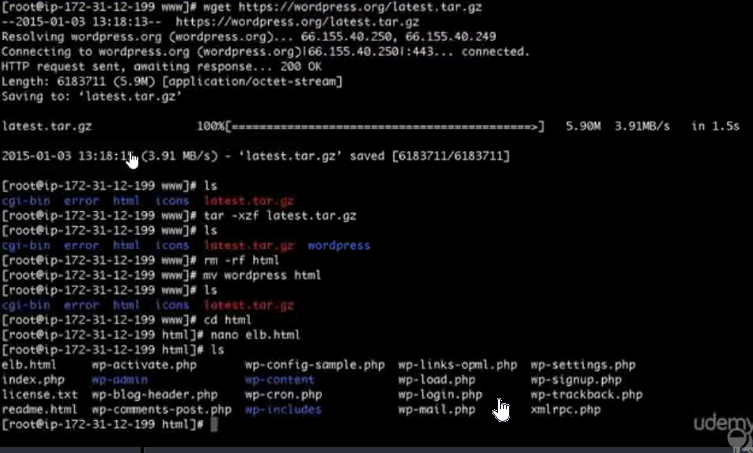
### Now test the site using “cloudarchitects.co.in/elb.html”, you should get “great success” displayed on the screen.

### INSTALL WORDPRESS in your instance

* 1. In your EC2 instance go to /var/www folder (cd /var/www)
  2. Download “Wordpress” using wget [*https://wordpress.org/latest.tar.gz*](https://wordpress.org/latest.tar.gz)
  3. Untar the file using tar xzf latest.tar.gz



* 1. rm -rf /var/www/html
  2. mv wordpress html
  3. go to html folder and create elb.html file again with “great success” as the contect



* 1. Be in /var/www folder and run below commands
     1. chown apache.apache html
     2. chmod –R 755 html

### Launch your WORDPRESS site using cloudarchitects.co.in

|  |
| --- |
| select English (united States) and Continue  In the following screen select the button with title “Lets Go” |
| Update the following screen with your database name,user Name, password and the DB endpoint.  In our case dbname,user name and password is “wordpressdb” and the end point was  **wordpressdb.c9pbehojgmiz.us-east-1.rds.amazonaws.com:3306**      Once done, “Submit” |
| Click “Run the Install” button |
| In the resulting screen: Update the site information as given below and press “Install Wordpress” button |
| Now Login |
|  |
| Now create a POST of your choise. Below is mine |
|  |
| Now access the site using load balancer URL |
| Now try with route 53 domain URL |
|  |

### Now copy and replicate your EC2 instance’s /var/www/html folder to the s3://wpcodejpa bucket

|  |
| --- |
| aws s3 cp –recursive /var/www/html/ s3://wpcodejps |
| If you want you can create a scheduled job to replicate the html folder with S3 and vice versa for any subfolder  crontab –e ( or cd /etc and nano crontab)  \*/2 \* \* \* \* root /usr/bin/aws s3 sync /var/www/html/ s3://wpcodejpa/  \*/2 \* \* \* \* root /usr/bin/aws s3 sync --recursive /var/www/html/wp-content/uploads s3://wpcdnjpa/  \*/3 \* \* \* \* root /usr/bin/aws s3 sync s3://wpcodejpa/ /var/www/html |
| Now make http redirect happening from your apache to cloudfront (note your cloudfront domain name)   Back in your instance running apache, create a new file (/var/www/html/.htaccess) and paste the below content in it  Options +FollowSymlinks  RewriteEngine on  rewriterule ^wp-content/uploads/(.\*)$ http://dsxombkwbumc2.cloudfront.net/$1 [r=301.nc]  # BEGIN WordPress  # END WordPress |
| You are ready . Launch your site in a browser and you’ll notice that the images are now distributed using cloudfront CDNIf you need you may scale it up or down using an autoscaling group and policies. |