**WordPress on AWS**

**Document Summary**

The following document describes the infrastructure for an AWS-hosted WordPress website that I have built.

**Project Summary**

The described infrastructure hosts a personal WordPress blog that I have created

This project uses AWS’s [WordPress Best Practices](https://d1.awsstatic.com/whitepapers/wordpress-best-practices-on-aws.pdf) as a launching pad - with certain concessions made to stay in AWS Free Tier.

**Diagram**

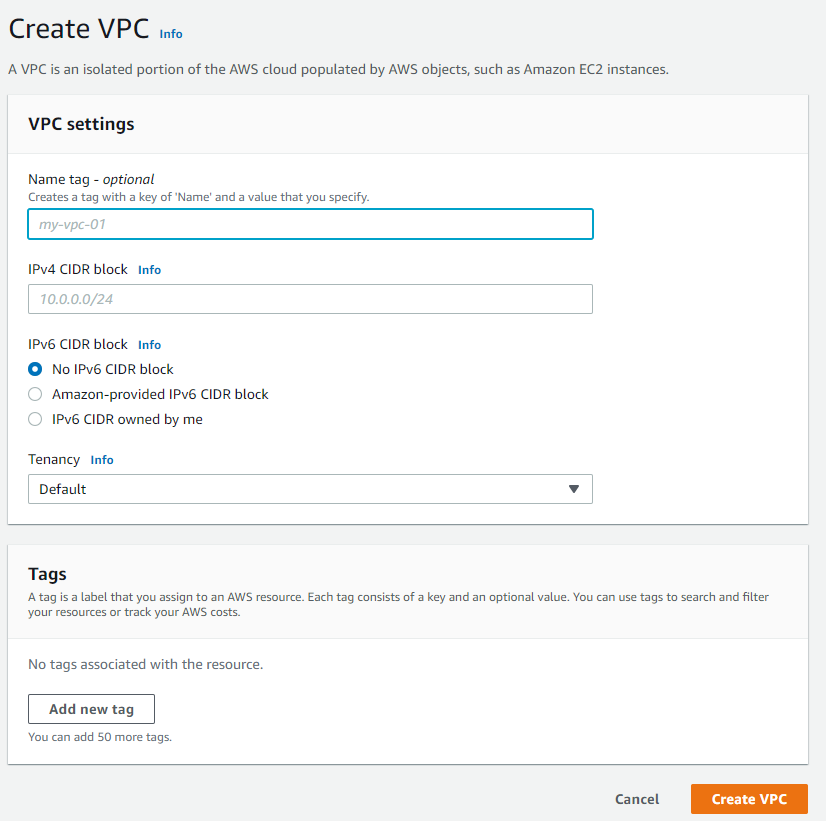
**Major Components**

1. WordPress Web Server - EC2 instance with AWS Linux 2 AMI
2. RDS - Database contents, running MariaDB
3. EFS\* - Web Contents served out of EFS to have consistency across AZs \*[performance note](https://aws.amazon.com/blogs/storage/optimizing-wordpress-performance-with-amazon-efs/)
4. Application LB - Allows load balancing between original instance and scaled in instances
5. S3 Bucket - For hosting static content like pictures

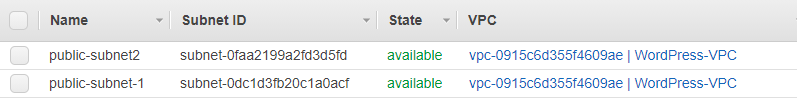
**Install steps**

**Phase 1 - Simple implementation. No scaling.**

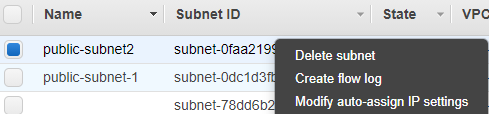
1. **Create VPC -** [**AWS Documentation for Wizard**](https://docs.aws.amazon.com/batch/latest/userguide/create-public-private-vpc.htmll)
   1. Skipping the Wizard for VPC creation practice…
   2. Create VPC

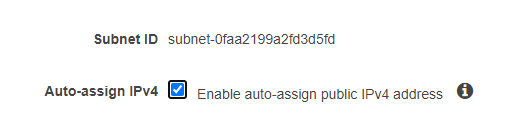
****

* 1. Create subnets that will exist in your VPC
     1. In our case, we will create two public subnets in us-east-1a and us-east-1b. We will need to create additional public subnets if we want to launch instances into us-east-1c - us-east-1f.
     2. We will not currently be making a private subnet but will in the future if we add caching.

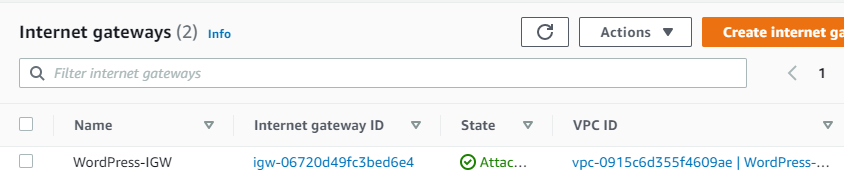
****

* 1. By **default,** our public subnets will not automatically assign a public IP address to EC2 instances created in those subnets. We can update that with the “Modify auto-assign IP settings” option

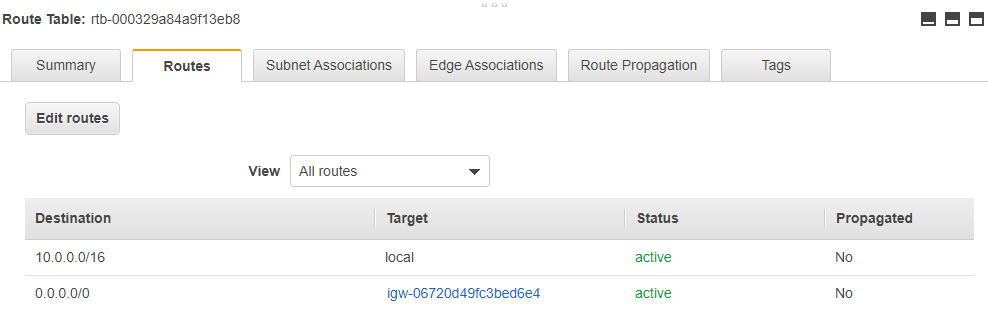
****

****

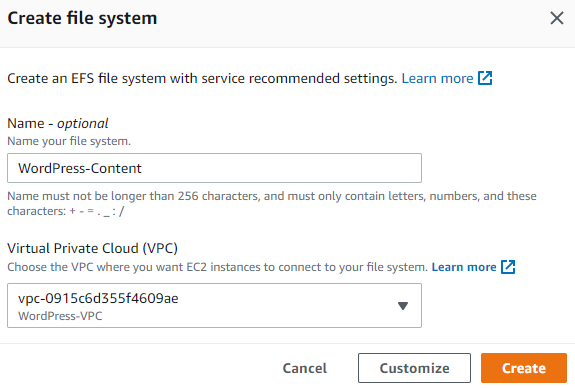
* 1. Create an Internet Gateway to our new VPC

****

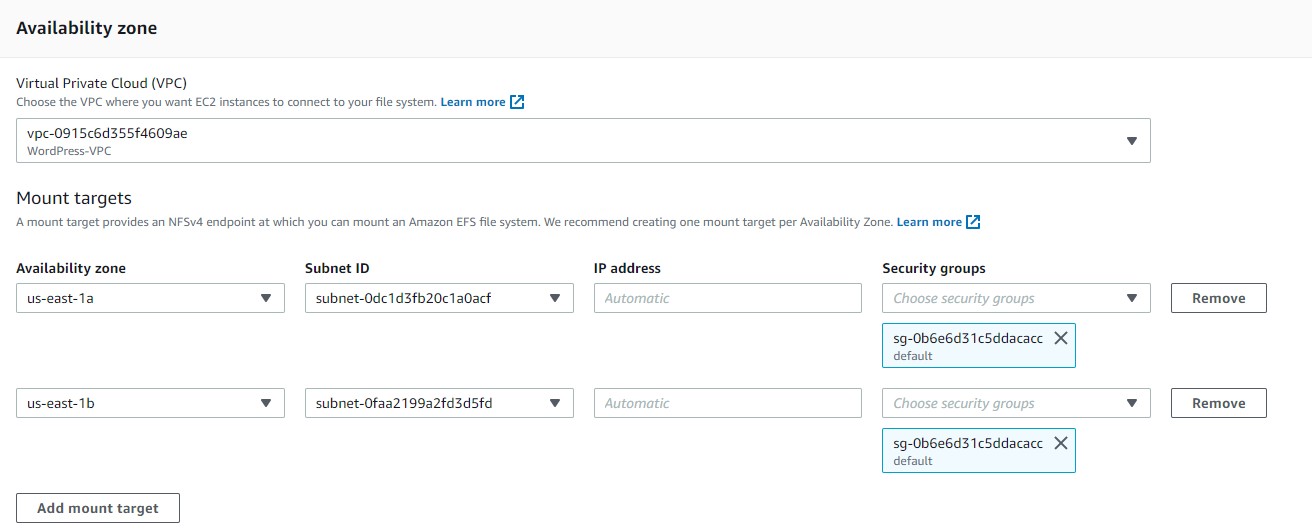
* 1. **Attach** the IGW to our routing table as pictured below

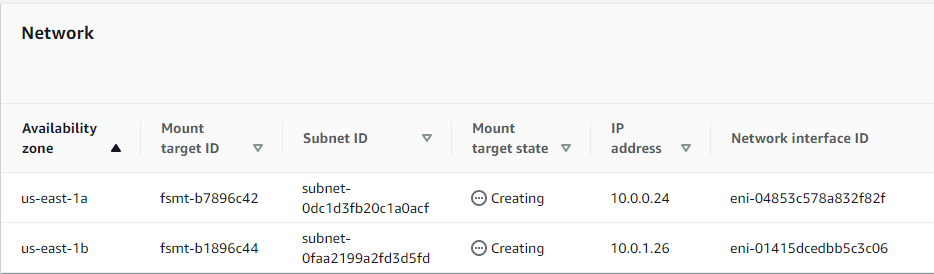
****

1. **Create EFS system and mount points**
   1. Click Create File System and enter a name and VPC

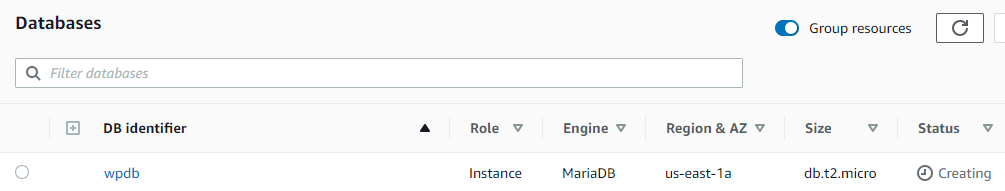
****

* 1. In Network, click Manage in the top right. This will take you to the page where you can create Mount Points
  2. Create a mount point for each subnet you will need. In our case for us-east-1a and us-east-1b.
     1. We will update the Security Groups for these after we create our web server.

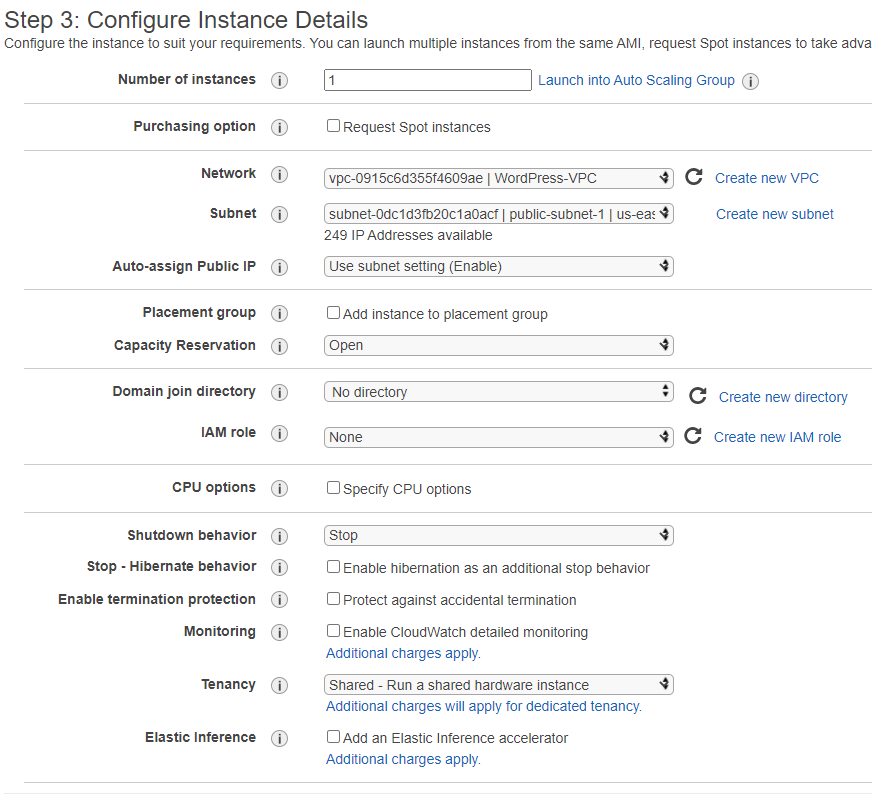
****

****

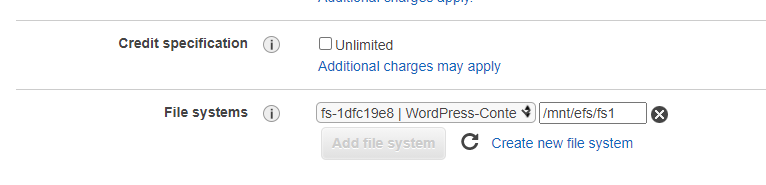
1. **Create RDS Database**
   1. We are mostly taking the defaults, but changing in a few key areas:
      1. MariaDB for Free Tier support
      2. **Make sure** to select the correct VPC
      3. Create with an existing database - and write it down (wpdb)
   2. This will be scaled at a later step.

****

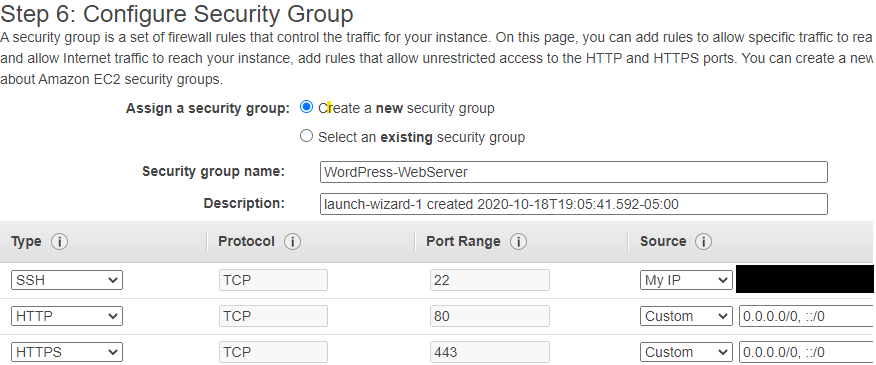
1. **Create EC2 Instance**
   1. Again, mostly taking defaults.
   2. Make sure to select the **correct VPC**!

****

* 1. Mount your EFS drive under the “File systems” section

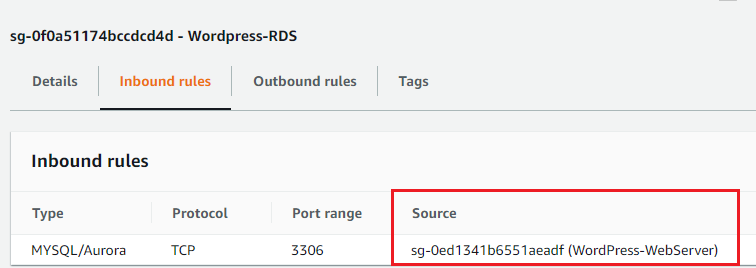
****

* 1. Create a Security Group allowing connections from SSH, HTTP, HTTPS, and RDS (from your existing RDS SG)

****

* 1. Create a Key Pair and save it on your local machine

1. **Configuration steps before installing WordPress**
   1. Update the Security Group that we attached to RDS earlier to allow access from our Web Server

****

**Phase Two Installing LEMP/WordPress onto our EC2 Instance**

1. **Installing NGINX and WordPress**
   1. The following Bash snippet will install php components, NGINX, and copy the WordPress installation files

#!/bin/bash

sudo yum update -y

sudo amazon-linux-extras install -y php7.2 nginx1

sudo systemctl start NGINX

sudo systemctl enable NGINX

sudo wget http://wordpress.org/latest.tar.gz

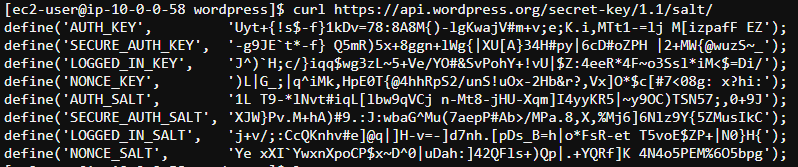
sudo tar xzvf latest.tar.gz

sudo cp wp-config-sample.php wp-config.php

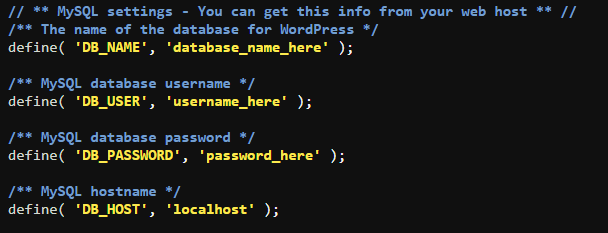
sudo cp -R ./ /var/www/wordpress

sudo chown -R NGINX: /var/www/wordpress

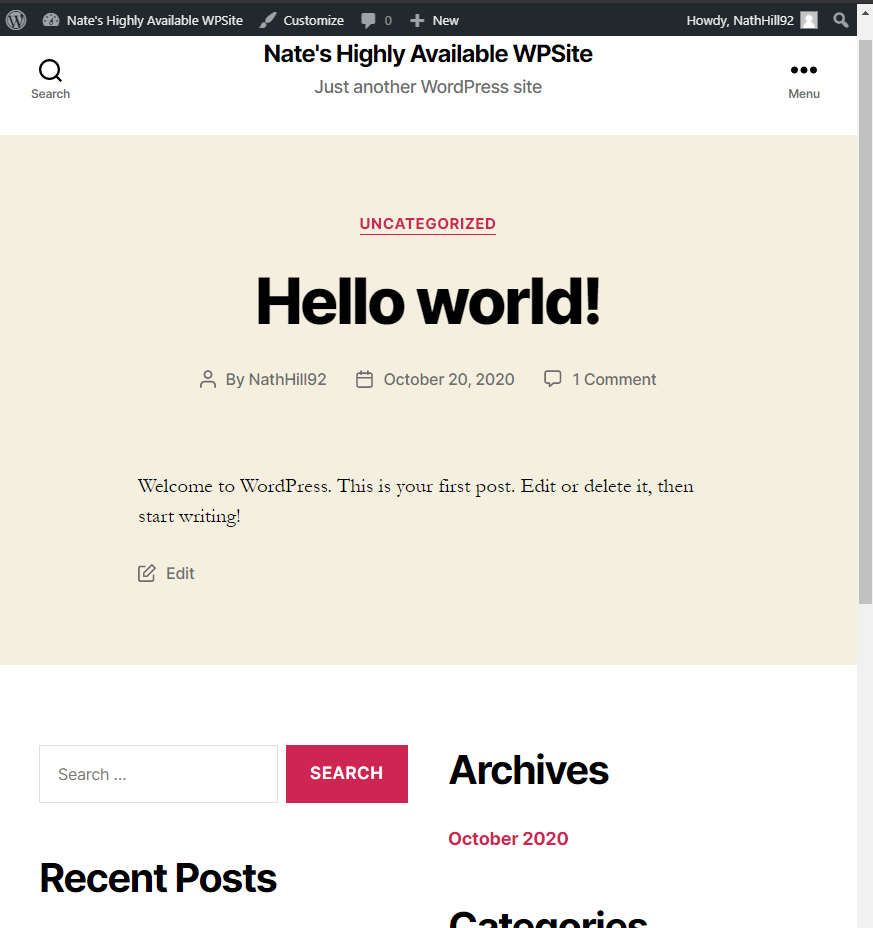
* 1. Next we will need to create a new Salt file for encrypting our WordPress contents.
     1. Copy contents from “curl <https://api.wordpress.org/secret-key/1.1/salt/>”



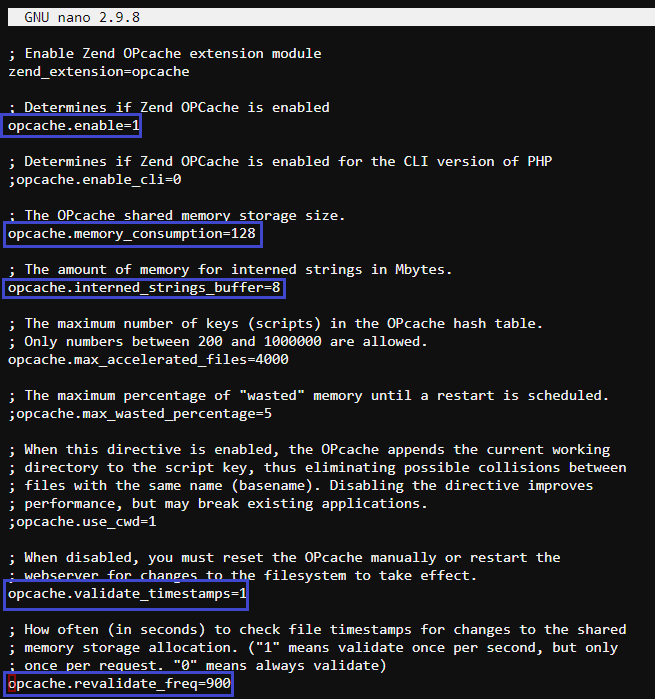
* + 1. And place them into the wp-config file
    2. Leave the config file open for the next section
  1. Update DB values with the data you put into RDS earlier



* 1. Create user and password
  2. You’re done!

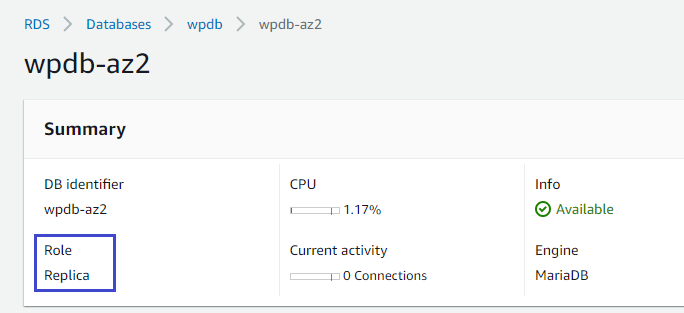


1. **Enable OPcache/bytecache to improve performance of PhP - especially over EFS**
   1. See <https://aws.amazon.com/blogs/storage/optimizing-wordpress-performance-with-amazon-efs/>
   2. Install OPcache module - “sudo yum install php-opcache
   3. Update config file for OPcache “sudo nano /etc/php.d/10-opcache.ini
   4. Enable the following settings - revalidate frequency is how long to hold onto the cache - 15 minutes for our uses

****

**Lets make it scale**

1. Create a Read Replica for our RDS instance for each Availability zone that we will deploy to. Use the same Security Group as your original instance
   1. Note - WP does not support this configuration normally. We will need to use an extension.

****

1. In the EC2 menu, right click our webserver and click **Create Image**. This will allow us to scale multiple instances with the same configuration that we just set.
2. Create a **Launch Template** in the EC2 menu