Project 6- (Web Solution with WordPress)

1. I configured a RedHat EC2 instance for the webserver using the AWS console and launched the instance on the terminal
2. I created 3 volumes in the same AZ (US-East 1) and the attached the 3 volumes to the web server Instance) one after the other on the AWS console
3. The command lsblk was used to list the volumes (new and old) and initial partition for the old volume present on the file system
4. The command df -h command was used to show the storage and free space on the server
5. The sudo g disk command was used to create partition on the three new disk/volumes
6. I ran the command lsblk to view the newly configured partition of the 3 disks and their respective storage sizes
7. The LVM2 package was installed using the command sudo yum install lvm2
8. I ran the command lvmdiskscan to check for available partitions
9. I used the command sudo pvcreate /dev/xvdf1 /dev/xvdg1 /dev/xvdh1 to create physical volumes PV
10. I ran the command sudo pvs to verify the physical volume was successfully created
11. I used the command sudo vgcreate vg-webdata /dev/xvdf1 /dev/xvdg1 /dev/xvdh1 to create the volume group
12. The sudo vgs command was ran to show the volume group was successful created and running
13. The command sudo lvcreate was used to to create two logical volumes (apps-iv and logs-iv)

sudo lvcreate -n apps-lv -L 14G webdata-vg

sudo lvcreate -n logs-lv -L 14G webdata-vg

1. I ran the command sudo lvs to verify the logical volumes were successfully created and running
2. The entire set up was verified using the command sudo vgdisplay -v #view complete setup - VG, PV, and LV

sudo lsblk

1. I used the command mkfs.ext4 to format the logical volumes with ext4 file system

sudo mkfs -t ext4 /dev/vg-webdata/apps-lv

sudo mkfs -t ext4 /dev/vg-webdata/logs-lv

1. I created a directory to store the website files using the command sudo mkdir -p /var/www/html
2. I created a directory to store backup of log data file using the command sudo mkdir -p /home/recovery/logs
3. I mounted **/var/www/html** on **apps-lv** logical volume using the command sudo mount /dev/vg-webdata/apps-lv /var/www/html/
4. I used the [rsync](https://linux.die.net/man/1/rsync) utility to backup all the files in the log directory **/var/log** into **/home/recovery/logs** (*This is required before mounting the file system*) using the command sudo rsync -av /var/log/. /home/recovery/logs/
5. I mounted **/var/log** on **logs-lv** logical volume using the command sudo mount /dev/vg-webdata/logs-lv /var/log
6. I restored the log files back into **/var/log** directory using the command sudo rsync -av /home/recovery/logs/. /var/log
7. I updated the /etc/fstab file so that the mount configuration will persist after restart of the server using the UUID of the device and running the following commands sudo blkid and sudo vi /etc/fstab
8. The configuration was tested and the daemon was reloaded using the command sudo mount -a and sudo systemctl daemon-reload
9. The set up was verified by running the command df -h

Database Server set up

1. I launched a second RedHat EC2 instance on the AWS console that will serve as a database server
2. I repeated the same steps as used for the web server, but in this case instead of the apps-lv, I created db-lv and mounted it to /db directory instead of /var/www/html

Installing WordPress on my EC2 server

1. I updated the repository using the command sudo yum -y update
2. I installed wget, Apache and it’s dependencies with the command

sudo yum -y install wget httpd php php-mysqlnd php-fpm php-json

1. I Started/activated Apache using the command

sudo systemctl enable httpd

sudo systemctl start httpd

1. I installed PHP and it’s dependencies with the following commands

sudo yum install https://dl.fedoraproject.org/pub/epel/epel-release-latest-8.noarch.rpm

sudo yum install yum-utils http://rpms.remirepo.net/enterprise/remi-release-8.rpm

sudo yum module list php

sudo yum module reset php

sudo yum module enable php:remi-7.4

sudo yum install php php-opcache php-gd php-curl php-mysqlnd

sudo systemctl start php-fpm

sudo systemctl enable php-fpm

setsebool -P httpd\_execmem 1

1. I restarted Apache using the command

sudo systemctl restart httpd

1. I downloaded wordpress and copy wordpress to var/www/html using the command

mkdir wordpress

cd wordpress

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

sudo tar xzvf latest.tar.gz

sudo rm -rf latest.tar.gz

cp wordpress/wp-config-sample.php wordpress/wp-config.php

cp -R wordpress /var/www/html/

1. I configured SELinux Policies with the following commands

sudo chown -R apache:apache /var/www/html/wordpress

sudo chcon -t httpd\_sys\_rw\_content\_t /var/www/html/wordpress -R

sudo setsebool -P httpd\_can\_network\_connect=1

Installing MySQL on Database server EC2

1. I installed MySQL using the commands

sudo yum update

sudo yum install mysql-server

1. I verified the service is up and running by using the commands sudo systemctl status mysqld and if not running I can use the following commands

sudo systemctl restart mysqld

sudo systemctl enable mysqld

#### Configure DB to work with WordPress

1. I configured the Databse to work with wordpress using the following commands

sudo mysql

CREATE DATABASE wordpress;

CREATE USER `myuser`@`<Web-Server-Private-IP-Address>` IDENTIFIED BY 'mypass';

GRANT ALL ON wordpress.\* TO 'myuser'@'<Web-Server-Private-IP-Address>';

FLUSH PRIVILEGES;

SHOW DATABASES;

exit

#### Configure WordPress to connect to the remote database.

1. I opened MySQL port (3306) database server EC2 on the AWS console
2. I allowed access to the DB server only from my web server IP’s address and the inbound rule configuration specify source as /32
3. I installed MySQL client and test that you can connect from your Web Server to your DB server by using mysql-client by using the following commands

sudo yum install mysql

sudo mysql -u admin -p -h <DB-Server-Private-IP-address>

1. I verified if I can successfully execute SHOW DATABASES; command and see a list of existing databases.
2. I changed permissions and configuration so Apache could use WordPress
3. I enabled TCP port 80 in Inbound Rules configuration for my Web Server EC2 (enable from everywhere 0.0.0.0/0 or from your workstation’s IP)
4. I tried to access from my browser the link to my WordPress http://<Web-Server-Public-IP-Address>/wordpress/ which works fine.
5. I filled out my DB credentials.