

DEPLOYING WORDPRESS IN A VPC

1. INTRODUCTION

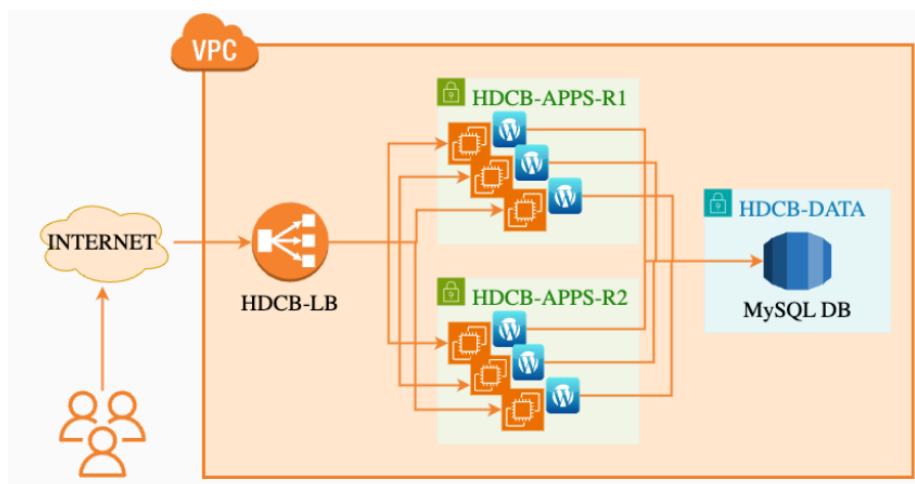
The aim of this exercise is to deploy wordpress in a VPC in order to learn about network configuration in AWS. Through the development of this exercise, we will explore several concepts like VPCs, subnets, Route Tables, Internet Gateway and Load Balancers.

We will also learn how to deploy EC2 and RDS instances.

The idea is to create a VPC that contains three subnets (two subnets for the application, named R1 and R2, and the other for the database).

The users have to be able to make queries through the load balancer and the load balancer will then be used to communicate the traffic to the EC2 instances

This is the architecture of the exercise:



2. NETWORKING

A) VPC AND SUBNETS

Amazon Virtual Private Cloud (VPC) is a virtual network infrastructure provided that allows you to create isolated sections of the AWS cloud where you can launch AWS resources in a logically isolated environment. Subnets, on the other hand, are subdivisions of a VPC's IP address range and are used to segment and organize the resources within a VPC.

We create the VPC named "HDCB-VPC":

VPC > Your VPCs > Create VPC

Create VPC [Info](#)

A VPC is an isolated portion of the AWS Cloud populated by AWS objects, such as Amazon EC2 instances.

VPC settings

Resources to create [Info](#)
Create only the VPC resource or the VPC and other networking resources.

☒ VPC only ☐ VPC and more

Name tag - optional
Creates a tag with a key of 'Name' and a value that you specify.

HDCB-VPC

IPv4 CIDR block [Info](#)
☒ IPv4 CIDR manual input
☐ IPAM-allocated IPv4 CIDR block

IPv4 CIDR
10.0.0.0/16
CIDR block size must be between /16 and /28.

IPv6 CIDR block [Info](#)
☒ No IPv6 CIDR block
☐ IPAM-allocated IPv6 CIDR block
☐ Amazon-provided IPv6 CIDR block
☐ IPv6 CIDR owned by me

Tenancy [Info](#)
Default ▼

After creating the VPC we have to create the subnets that reside inside that VPC:

Create subnet [Info](#)

VPC

VPC ID
Create subnets in this VPC.
vpc-00dd0b83b42f08bc6 (HDCB-VPC)

Associated VPC CIDRs
IPv4 CIDRs
10.0.0.0/16

Subnet settings

Specify the CIDR blocks and Availability Zone for the subnet.

Subnet 1 of 1

Subnet name

Create a tag with a key of 'Name' and a value that you specify.

HDCB-APPS-R1

The name can be up to 256 characters long.

Availability Zone [Info](#)

Choose the zone in which your subnet will reside, or let Amazon choose one for you.

US East (N. Virginia) / us-east-1a

IPv4 VPC CIDR block [Info](#)

Choose the VPC's IPv4 CIDR block for the subnet. The subnet's IPv4 CIDR must lie within this block.

10.0.0.0/16

IPv4 subnet CIDR block

10.0.1.0/24 256 IPs

VPC > Subnets > Create subnet

Create subnet [Info](#)

VPC

VPC ID
Create subnets in this VPC.
vpc-00dd0b83b42f08bc6 (HDCB-VPC)

Associated VPC CIDRs
IPv4 CIDRs
10.0.0.0/16

Subnet settings

Specify the CIDR blocks and Availability Zone for the subnet.

Subnet 1 of 1

Subnet name

Create a tag with a key of 'Name' and a value that you specify.

HDCB-APPS-R2

The name can be up to 256 characters long.

Availability Zone [Info](#)

Choose the zone in which your subnet will reside, or let Amazon choose one for you.

No preference

IPv4 VPC CIDR block [Info](#)

Choose the VPC's IPv4 CIDR block for the subnet. The subnet's IPv4 CIDR must lie within this block.

10.0.0.0/16

IPv4 subnet CIDR block

10.0.2.0/24 256 IPs

VPC > Subnets > Create subnet

Create subnet [Info](#)

VPC

VPC ID
Create subnets in this VPC.
vpc-00dd0b83b42f08bc6 (HDCB-VPC)

Associated VPC CIDRs
IPv4 CIDRs
10.0.0.0/16

Subnet settings

Specify the CIDR blocks and Availability Zone for the subnet.

Subnet 1 of 1

Subnet name

Create a tag with a key of 'Name' and a value that you specify.

HDCB-DATA

The name can be up to 256 characters long.

Availability Zone [Info](#)

Choose the zone in which your subnet will reside, or let Amazon choose one for you.

No preference

IPv4 VPC CIDR block [Info](#)

Choose the VPC's IPv4 CIDR block for the subnet. The subnet's IPv4 CIDR must lie within this block.

10.0.0.0/16

IPv4 subnet CIDR block

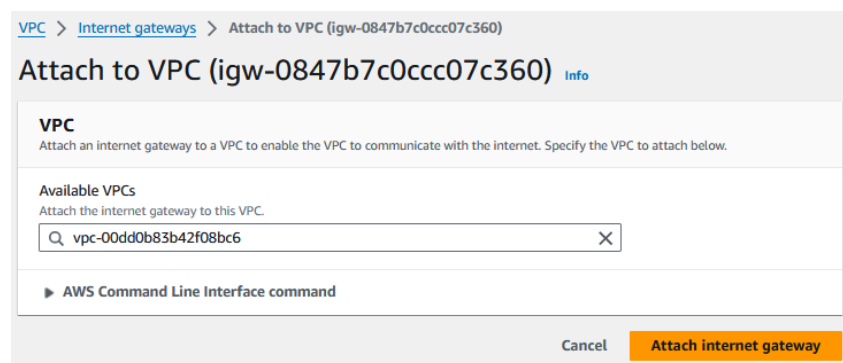
10.0.3.0/24 256 IPs

Load balancers need to use at least two subnets, and to make it more real we will place the subnets in different zones. In that way, if an ability zone is down, the architecture will still work in another availability zone.

These subnets are now completely isolated, so we have to create an Internet Gateway. An Internet Gateway in AWS provides connectivity between a VPC and the internet, enabling resources within the VPC to access internet resources and allowing internet-based users or services to access resources within the VPC.

Internet Gateways are typically used for resources in public subnets within a VPC, where those resources require direct connectivity to the internet. Publicly accessible services, such as web servers or load balancers, often reside in public subnets with an associated route to the Internet Gateway.

After creating it, we have to attach the internet Gateway to our VPC:



Now that we created the Internet Gateway we must connect it to the subnets via the Route Tables.

B) ROUTE TABLES

In AWS, a route table is a set of rules, called routes, that are used to determine where network traffic from the VPC should be directed. Each subnet in a VPC must be associated with a route table, and the route table controls the routing of traffic within that subnet.

We have to create two Route Tables in our VPC, one for the DCB-APPS-R1 subnet and the other for the DCB-APPS-R2 subnet. Both need to access the internet. The database does not need to access the internet, so we do not need to route HDCB-DATA subnet.

We will exemplify the creation of the route table for DCB-APPS-R1, although the same process is used for DCB-APPS-R2:

VPC > Route tables > Create route table

Create route table [Info](#)

A route table specifies how packets are forwarded between the subnets within your VPC, the internet, and your VPN connection.

Route table settings

Name - optional
Create a tag with a key of 'Name' and a value that you specify.

VPC
The VPC to use for this route table.

Tags

A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.

Key **Value - optional**

You can add 49 more tags.

After creating the route tables we make the association with the correspondent subnet:

VPC > Route tables > rtb-0699783fc9ea54153

rtb-0699783fc9ea54153 / HDCB-APPS-R1-RT Actions

Details [Info](#)

Route table ID rtb-0699783fc9ea54153	Main <input checked="" type="checkbox"/> No	Explicit subnet associations subnet-02b85c1d09f517d09 / HDCB-APPS-R1	Edge associations -
VPC vpc-00dd0b83b42f08bc6 HDCB-VPC	Owner ID 654654181466		

Subnet associations

Explicit subnet associations (1)

Name	Subnet ID	IPv4 CIDR	IPv6 CIDR
HDCB-APPS-R1	subnet-02b85c1d09f517d09	10.0.1.0/24	-

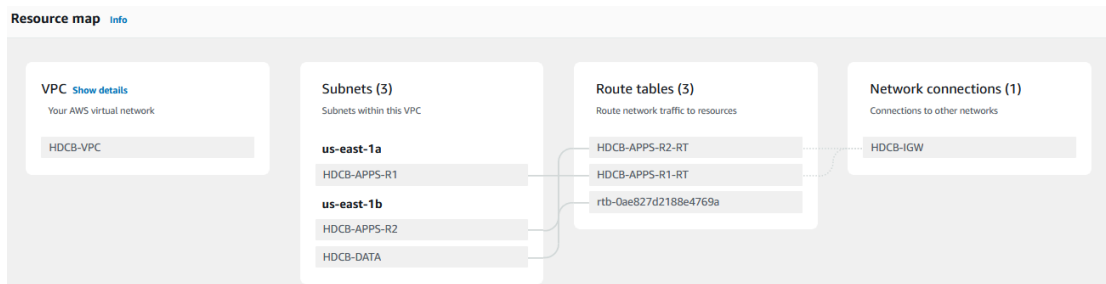
Then we add a new route to allow all traffic throughout the internet gateway specified before:

VPC > Route tables > [rtb-0699783fc9ea54153](#) > Edit routes

Edit routes

Destination	Target	Status
10.0.0.0/16	local	<input checked="" type="checkbox"/> Active
<input type="text" value="0.0.0.0/0"/>	Internet Gateway	-
	igw-0847b7c0ccc07c360	

The following scheme shows the final network configuration:



3. DEPLOYING AN EC2 INSTANCE

We launch an EC2 instance with the name “HDCB-WP”, keeping the default AMI and configurations. In the networking part, we create a new security group as follows:

Network settings [Info](#)

VPC - required [Info](#)

vpc-00dd0b83b42f08bc6 (HDCB-VPC)
10.0.0.0/16

Subnet [Info](#)

subnet-02b85c1d09f517d09 HDCB-APPS-R1
VPC: vpc-00dd0b83b42f08bc6 Owner: 654654181466
Availability Zone: us-east-1a IP addresses available: 251 CIDR: 10.0.1.0/24

Auto-assign public IP [Info](#)

Enable

Firewall (security groups) [Info](#)

A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

☒ Create security group ☐ Select existing security group

Security group name - required

HDCB-WP-5G

Description - required [Info](#)

Allow SSH and HTTP traffic

Inbound Security Group Rules

Security group rule 1 (TCP, 22, 0.0.0.0/0)

Type [Info](#) ssh Protocol [Info](#) TCP Port range [Info](#) 22

Source type [Info](#) Anywhere Source [Info](#) 0.0.0.0/0 Description - optional [Info](#) e.g. SSH for admin desktop

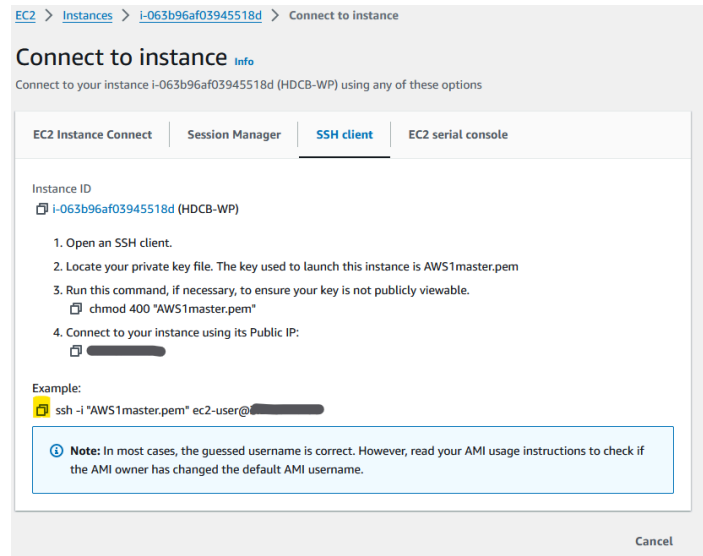
Security group rule 2 (TCP, 80, 0.0.0.0/0)

Type [Info](#) HTTP Protocol [Info](#) TCP Port range [Info](#) 80

Source type [Info](#) Custom Source [Info](#) 0.0.0.0/0 Description - optional [Info](#) e.g. SSH for admin desktop

In AWS, security groups act as virtual firewalls for EC2 instances and other resources within the VPC. They basically control inbound and outbound traffic at the instance level, and this enables us to specify which traffic is allowed to reach our instances.

Afterwards, we go to our terminal and connect to the instance by copying the example command and using our private key:



In this virtual machine, we will install wordpress:

vi install_wp.sh

We type “i” and insert the following code:

```
#!/bin/bash
# sudo bash install_wp.sh

# Check if the script is run with sudo
if [ "$(id -u)" -ne 0 ]; then
    echo "Please run this script with sudo or as root."
    exit 1
fi

# Install necessary packages
sudo dnf install -y wget php-mysqlnd httpd php-fpm php-mysqli php-json php php-devel php-gd

# Download and extract WordPress
cd /tmp
wget https://wordpress.org/latest.tar.gz
tar -xzf latest.tar.gz

# Start and Enable Apache
sudo systemctl start --now httpd

# Move WordPress files to the web directory
cp -r wordpress/* /var/www/html/

# Configure Apache
sed -i 's/AllowOverride None/AllowOverride All/g' /etc/httpd/conf/httpd.conf

# Set permissions
chown -R apache:apache /var/www
chmod 2775 /var/www

# Restart Apache
systemctl restart httpd
```

To save the bash script we press esc and type :wq, and then press enter.

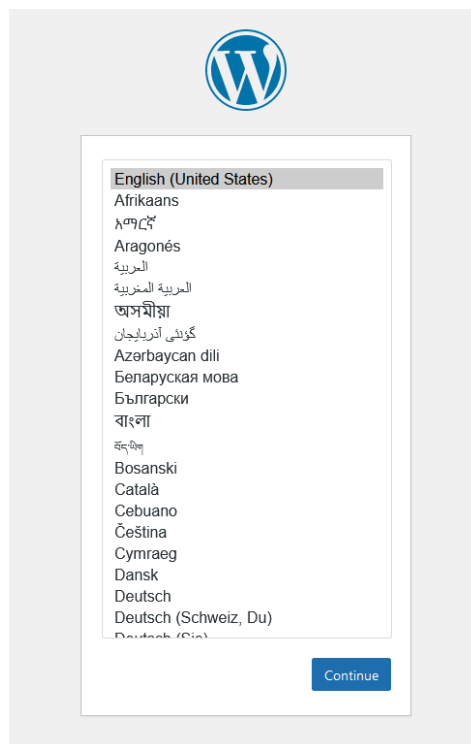
To install wordpress we have to run the bash script we just saved. We do that by executing the following command:

```
sudo bash install_wp.sh
```

Then, we copy the public IP of the instance and go to the browser:

http://PUBLIC_IP

We will see that we are accessing the installation of wordpress:



4. RDS AND DATABASE TESTING

Amazon RDS is a fully managed relational database service that simplifies the process of deploying, operating, and scaling relational databases in the cloud.

We go to the RDS service and create a database:

RDS > Create database

Create database

Choose a database creation method [Info](#)

☒ **Standard create**
 You set all of the configuration options, including ones for availability, security, backups, and maintenance.

☐ **Easy create**
 Use recommended best-practice configurations. Some configuration options can be changed after the database is created.

Engine options

Engine type [Info](#)

☐ Aurora (MySQL Compatible)

☐ Aurora (PostgreSQL Compatible)

☒ MySQL

☐ MariaDB

Connectivity [Info](#) [↻](#)

Compute resource
 Choose whether to set up a connection to a compute resource for this database. Setting up a connection will automatically change connectivity settings so that the compute resource can connect to this database.

☒ **Don't connect to an EC2 compute resource**
 Don't set up a connection to a compute resource for this database. You can manually set up a connection to a compute resource later.

☐ **Connect to an EC2 compute resource**
 Set up a connection to an EC2 compute resource for this database.

Virtual private cloud (VPC) [Info](#)
 Choose the VPC. The VPC defines the virtual networking environment for this DB instance.

HDCB-VPC (vpc-00dd0b83b42f08bc6)
 3 Subnets, 2 Availability Zones

Only VPCs with a corresponding DB subnet group are listed.

☒ After a database is created, you can't change its VPC.

DB subnet group [Info](#)
 Choose the DB subnet group. The DB subnet group defines which subnets and IP ranges the DB instance can use in the VPC that you selected.

Create new DB Subnet Group

Public access [Info](#)

☐ **Yes**
 RDS assigns a public IP address to the database. Amazon EC2 instances and other resources outside of the VPC can connect to your database. Resources inside the VPC can also connect to the database. Choose one or more VPC security groups that specify which resources can connect to the database.

☒ **No**
 RDS doesn't assign a public IP address to the database. Only Amazon EC2 instances and other resources inside the VPC can connect to your database. Choose one or more VPC security groups that specify which resources can connect to the database.

VPC security group (firewall) [Info](#)
 Choose one or more VPC security groups to allow access to your database. Make sure that the security group rules allow the appropriate incoming traffic.

☐ **Choose existing**
 Choose existing VPC security groups

☒ **Create new**
 Create new VPC security group

New VPC security group name

HDCB-RDS-SG

Additional configuration
 Database options, encryption turned on, backup turned on, backtracking turned off, maintenance, CloudWatch Logs, delete protection turned off.

Database options

Initial database name [Info](#)

wordpress_db

If you do not specify a database name, Amazon RDS does not create a database.

DB parameter group [Info](#)

default:mysql8.0

Option group [Info](#)

default:mysql-8-0

Backup

☒ **Enable automated backups**
 Creates a point-in-time snapshot of your database

Next, we edit the security group that was automatically created. We delete the inbound rule and create a new one that allows all traffic coming from the wordpress security group:

EC2 > Security Groups > sg-021f217634aa8fb3d - HDCB-RDS-SG > Edit inbound rules

Edit inbound rules [info](#)

Inbound rules control the incoming traffic that's allowed to reach the instance.

Inbound rules [info](#)

Security group rule ID	Type info	Protocol info	Port range info	Source info
-	Custom TCP	TCP	3306	Custom

This means that all the instances attached to this HDCB-WP-SG security group are going to be allowed to send traffic to this port. In this case, we are only accepting inbound traffic so we can delete the outbound traffic set by default.

Now, let's test that everything works by trying to connect to the database using this configuration:

Below you should enter your database connection details. If you are not sure about these, contact your host.

Database Name
The name of the database you want to use with WordPress.

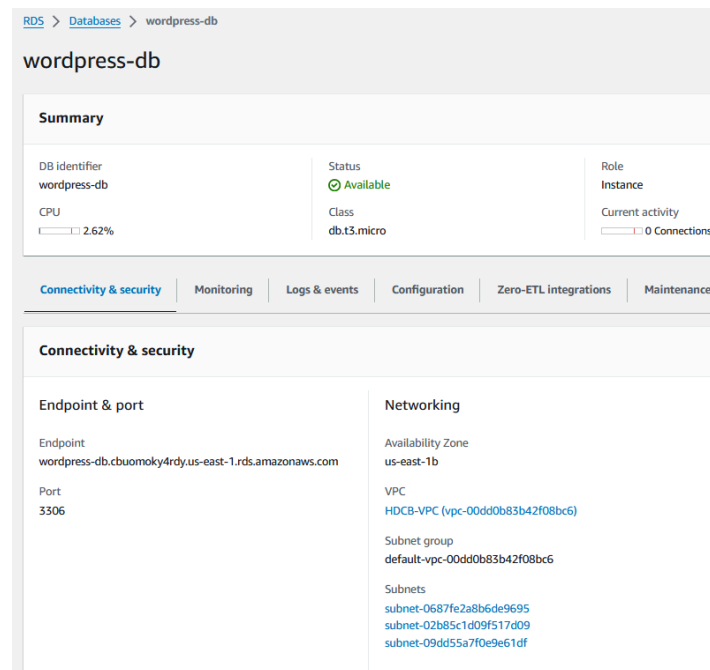
Username
Your database username.

Password
Your database password.

Database Host
You should be able to get this info from your web host, if localhost does not work.

Table Prefix
If you want to run multiple WordPress installations in a single database, change this.

Note that the database host is the endpoint we got when creating the database:



5. LOAD BALANCER CONFIGURATION

When we deploy an EC2 instance, the IP is dynamic, which means that if we stop the instance or re-launch it, this IP can change. For a web service, having a dynamic IP is not the best thing, since we would have to reconfigure the web service for every time our IP changes. In this case, we can create an elastic IP (which is static) and assign it to our instances.

A load balancer is something that has an elastic IP and also is able to distribute the traffic across different services. Load balancers are critical components in distributed computing and networking architectures since they help to distribute incoming network traffic across multiple servers (or EC2 instances in our case) to ensure efficient use of resources, optimize performance, and enhance the availability and reliability of applications.

We create an application load balancer:

EC2 > Load balancers > Compare and select load balancer type

Compare and select load balancer type

A complete feature-by-feature comparison along with detailed highlights is also available. [Learn more](#)

Load balancer types

Application Load Balancer [Info](#)

Choose an Application Load Balancer when you need a flexible feature set for your applications with HTTP and HTTPS traffic. Operating at the request level, Application Load Balancers provide advanced routing and visibility features targeted at application architectures, including microservices and containers.

[Create](#)

Network Load Balancer [Info](#)

Choose a Network Load Balancer when you need ultra-high performance, TLS offloading at scale, centralized certificate deployment, support for UDP, and static IP addresses for your applications. Operating at the connection level, Network Load Balancers are capable of handling millions of requests per second securely while maintaining ultra-low latencies.

[Create](#)

Gateway Load Balancer [Info](#)

Choose a Gateway Load Balancer when you need to deploy and manage a fleet of third-party virtual appliances that support GENEVE. These appliances enable you to improve security, compliance, and policy controls.

[Create](#)

Load balancer name
Name must be unique within your AWS account and can't be changed after the load balancer is created.

A maximum of 32 alphanumeric characters including hyphens are allowed, but the name must not begin or end with a hyphen.

Scheme [Info](#)
Scheme can't be changed after the load balancer is created.

☒ **Internet-facing**
An internet-facing load balancer routes requests from clients over the internet to targets. Requires a public subnet. [Learn more](#)

☐ **Internal**
An internal load balancer routes requests from clients to targets using private IP addresses.

IP address type [Info](#)
Select the type of IP addresses that your subnets use.

☒ **IPv4**
Recommended for internal load balancers.

☐ **Dualstack**
Includes IPv4 and IPv6 addresses.

Network mapping [Info](#)
The load balancer routes traffic to targets in the selected subnets, and in accordance with your IP address settings.

VPC [Info](#)
Select the virtual private cloud (VPC) for your targets or you can [create a new VPC](#). Only VPCs with an internet gateway are enabled for selection. The selected VPC can't be changed after the load balancer is created. To confirm the VPC for your targets, view your [target groups](#).

[vpc-06a5d8a13a42f98bd6](#) [IPv4: 10.0.0.0/16](#) [C](#)

Mappings [Info
Select at least two Availability Zones and one subnet per zone. The load balancer routes traffic to targets in these Availability Zones only. Availability Zones that are not supported by the load balancer or the VPC are not available for selection.](#)

☒ **us-east-1a (use1-az1)**

Subnet: HDCB-APPS-R1

IPv4 address
Assigned by AWS

☒ **us-east-1b (use1-az2)**

Subnet: HDCB-APPS-R2

We also create a security group for the load balancer. We want everyone to be able to access the load balancer from the internet (HTTP) using port 80:

EC2 > Security Groups > Create security group

Create security group [Info](#)

A security group acts as a virtual firewall for your instance to control inbound and outbound traffic. To create a new security group, complete the fields below.

Basic details

Security group name [Info](#)

Name cannot be edited after creation.

Description [Info](#)

VPC [Info](#)

Inbound rules [Info](#)

Type Info	Protocol Info	Port range Info	Source Info
<input type="text" value="HTTP"/>	TCP	80	Anywhere-IPv4

However, the outbound traffic should only be redirected to the wordpress instances, so the destination should be the security group of wordpress:

Outbound rules [Info](#)

Type Info	Protocol Info	Port range Info	Destination Info
<input type="text" value="HTTP"/>	TCP	80	Custom

Finally, we have to create a **target group**. Target groups define where the load balancer directs traffic after receiving it. They play a crucial role in distributing traffic across multiple targets and ensuring the availability and scalability of applications hosted behind the load balancer.

In this case our target is the EC2 instance.

EC2 > Target groups > Create target group

Step 1
[Specify group details](#)

Step 2
Register targets

Register targets

This is an optional step to create a target group. However, to ensure that your load balancer routes traffic to this target group you must register your targets.

Available instances (1/1)

<input checked="" type="checkbox"/>	Instance ID	Name	State	Security groups	Zone	Private IPv4 address	Subnet ID	Launch time
<input checked="" type="checkbox"/>	i-063b96a703945518d	HDCB-WP	Running	HDCB-WP-SG	us-east-1a	10.0.1.224	subnet-02b85c1d009f517009	March 17, 2024, 18:19 (UTC+)

1 selected

Ports for the selected instances
 Ports for routing traffic to the selected instances.

1-65535 (separate multiple ports with comma)

Finally, we deploy the load balancing. Once we have the load balancing working, we can configure wordpress using this elastic IP.

When the load balancer is active, we can copy the DNS and paste it into the browser:

EC2 > Load balancers > HDCB-LB

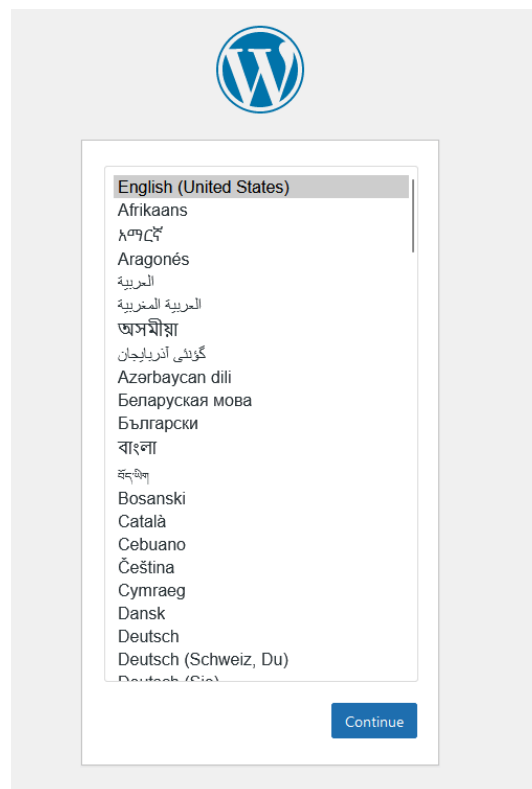
HDCB-LB

Introducing resource map for Application Load Balancers
 Resource map is a visual representation of the relationships between load balancer resources and provides the ability to view, explore, and troubleshoot the architecture of your load balancer. Resource map can be viewed on the console to improve your experience.

▼ Details

Load balancer type	Status	VPC
Application	Active	vpc-00dd0b83b42f08bc6
Scheme	Hosted zone	Availability Zones
Internet-facing	Z355XDOTRQ7X7K	subnet-0687fe2a8b6de9695 us-east-1b (use1-az2)
		subnet-02b85c1d09f517d09 us-east-1a (use1-az1)
Load balancer ARN	DNS name	
arn:aws:elasticloadbalancing:us-east-1:654654181466:loadbalancer/app/HDCB-LB/587c5e27ddcb280d	HDCB-LB- east-1.elb.amazonaws.com (A Record)	

If we do that, we will also be redirected to the wordpress installation page:



We can follow the steps and install it.

Finally, we have to modify the inbound rules for the wordpress security group, only to allow traffic from the load balancer security group:

EC2 > Security Groups > sg-0c269c70819d9b266 - HOCB-WP-SG > Edit inbound rules

Edit inbound rules [info](#)

Inbound rules control the incoming traffic that's allowed to reach the instance.

Inbound rules [info](#)

Security group rule ID	Type info	Protocol info	Port range info	Source info	Description - optional info	
sg-027d9527cb7b9748a	SSH	TCP	22	Custom	<input type="text" value="0.0.0.0/0"/>	<input type="button" value="Delete"/>
-	Custom TCP	TCP	0	Custom	<input type="text" value="sg-08588e456a9cbebe9"/> <input type="text" value="sg-08588e456a9cbebe9"/>	<input type="button" value="Delete"/>

6. REFERENCES

This exercise was proposed and provided by Jordi Mateo, professor of the subject High Performance and Distributed Computing for Big Data (URV).

<https://github.com/JordiMateoUdL>