

# Cloud Computing - Laboratory 02 : App scaling on Amazon Web Services

Author : Baptiste Hardrick & David Jaquet

## Create a database using the RDS



- Copy the estimated costs you calculated and add it to your report.
  - As you can see below, the estimated costs is about **\$ 14.71** by month

### Coûts mensuels estimés


instance de base de données	12.41 USD
Stockage	2.30 USD
<b>Total</b>	<b>14.71 USD</b>

- Compare the costs of your RDS instance to a continuously running EC2 instance of the same size using the AWS calculator. (Don't forget to uncheck the **Free Usage Tier** checkbox at the top.)
  - As you can see below, we use an EBS Volume of **20 GB** an it costs **\$ 10.50**. The RDS instance is more expensive.



**Compute: Amazon EC2 Instances:**

Description	Instances	Usage	Type	Billing Option	Monthly Cost
	1	100 % Utilized/Mo	Linux on t2.micro	On-Demand (No Cor)	\$ 8.50
 Add New Row					

**Compute: Amazon EC2 Dedicated Hosts:**

Description	Number of Hosts	Usage	Type	Billing Option
 Add New Row				

**Storage: Amazon EBS Volumes:**

Description	Volumes	Volume Type	Storage	IOPS	Baseline Throughput	Snapshot Storage
	1	General Purpose SSD (gp2)	20 GB	100	128 MBs/sec	0 GB-month of Storage
 Add New Row						

**Summary:**

Amazon EC2 Service (US East (N. Virginia))		\$ 10.50
Compute:	\$ 8.50	
EBS Volumes:	\$ 2.00	
EBS IOPS:	\$ 0.00	
AWS Support (Basic)		\$ 0.00
<b>Total Monthly Payment:</b>		<b>\$ 10.50</b>

- In a two-tier architecture the web application and the database are kept separate and run on different hosts. Imagine that for the second tier instead of using RDS to store the data you would create a virtual machine in EC2 and install and run yourself a database on it. If you were the Head of IT of a medium-size business, how would you argue in favor of using a database as a service instead of running your own database on an EC2 instance? How would you argue against it?
  - Using RDS is better than an a database for multiple reasons :
    - The execution of queries is faster.
    - Best tuples indexing
    - We can easily create some relation between 2 tables

- RDS handle concurrency, so multiple users can access to the databases at the same time
  - RDS handle large datasets more easily
  - It scales well
- Copy the endpoint address of the database into the report.
  - The endpoint address of the database is [hardrick-drupal.cv3ojyhiunqm.us-east-1.rds.amazonaws.com](https://hardrick-drupal.cv3ojyhiunqm.us-east-1.rds.amazonaws.com)

## Additional Information

In this task, we had to create a user for DB Instance. We took this laboratory as an exercise, so the credentials are not secure at all and should not be used in a real situation. The credentials are the following:

Username	Password
admin	adminpassword

## Configure the Drupal master instance to use the RDS database

- Copy the part of **settings.php** that configures the database into the report.
  - You can find below a screenshot of the **settings.php** file :

```
$databases['default']['default'] = array (
  'database' => 'hardrick_drupal',
  'username' => 'admin',
  'password' => 'adminpassword',
  'prefix' => '',
  'host' => 'hardrick-drupal.cv3ojyhiunqm.us-east-1.rds.amazonaws.com',
  'port' => '3306',
  'namespace' => 'Drupal\\Core\\Database\\Driver\\mysql',
  'driver' => 'mysql',
);
$settings['config_sync_directory'] = 'sites/default/files/config_28QZ40xM2SI0lRM-MNpNup1mAvTwkjJkgkDEyhjQFy5n0QDfzBeulQLG0k3Bg80rKYmXV20WCwg/sync';
```

## Additional Information

In this task, we had to create a user for DB Instance. We took this laboratory as an exercise, so the credentials are not secure at all and should not be used in a real situation. The credentials are the following:

Username	Password
newadmin	newadminpassword

## Create a custom virtual

- Copy a screenshot of the AWS console showing the AMI parameters into the report.
  - You can find below a screenshot of the AWS console with the AMI parameters :

Nom d'AMI	ID d'AMI	Source	Propriétaire	Visibilité	Statut	Date de création	Plateforme	Type de périph	Virtualisation
hardrick-Drupal	ami-0912c4207aabf68aa	475486677660/...	475486677660	Privé	available	20 mars 2020 15:57:27 UTC+1	Other Linux	ebs	hvm

Image: ami-0912c4207aabf68aa

Détails

Autorisations

Balises

ID d'AMI

ami-0912c4207aabf68aa

Propriétaire

475486677660

Statut

available

Date de création

20 mars 2020 15:57:27 UTC+1

Architecture

x86\_64

Type d'image

machine

Description

Drupal connected to RDS database

Type de périphérique racine

ebs

ID du noyau

-

Périphériques de stockage en mode bloc

/dev/sda1=snap-01a471135221e1e06:8:true:gp2

Nom d'AMI

hardrick-Drupal

Source

475486677660/hardrick-Drupal

Raison de l'état

-

Platform details

-

Usage operation

-

Type de virtualisation

hvm

Nom du périphérique racine

/dev/sda1

ID de disque RAM

-

Codes produit

-

## Create a load balancer

- On your local machine resolve the DNS name of the load balancer into an IP address using the nslookup command (Linux or Windows). Write the DNS name and the resolved IP Address(es) into the report.
  - As you can see in screenshot below, the DNS name of our load balancer is [hardrick-loadBalancer-410735606.us-east-1.elb.amazonaws.com](https://hardrick-loadBalancer-410735606.us-east-1.elb.amazonaws.com) and the IP addresses are 52.7.249.117 and 52.6.4.29.

```
(base) batach31@ubuntu:~$ nslookup hardrick-loadBalancer-410735606.us-east-1.elb.amazonaws.com
Server:         127.0.0.53
Address:        127.0.0.53#53

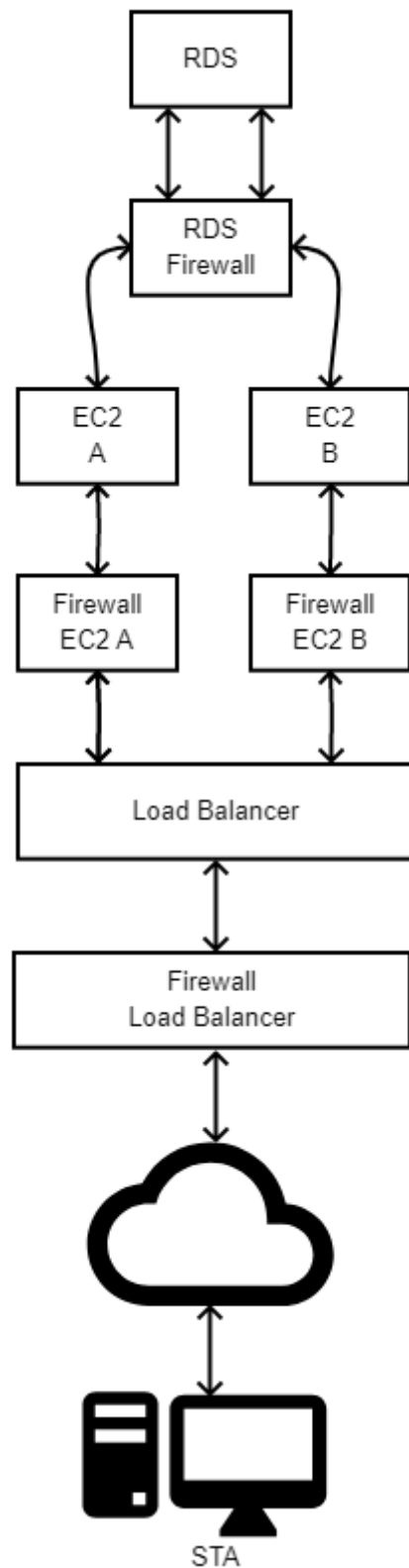
Non-authoritative answer:
Name:   hardrick-loadBalancer-410735606.us-east-1.elb.amazonaws.com
Address: 52.7.249.117
Name:   hardrick-loadBalancer-410735606.us-east-1.elb.amazonaws.com
Address: 52.6.4.29
```

- In the Apache access log identify the health check accesses from the load balancer and copy some samples into the report.
  - You can find below the messages of the load balancer's health check. The messages have been recorded in the logs.

```
172.31.5.3 - - [20/Mar/2020:15:31:28 +0000] "GET / HTTP/1.1" 200 3440 "-" "ELB-HealthChecker/2.0"
172.31.32.106 - - [20/Mar/2020:15:31:58 +0000] "GET / HTTP/1.1" 200 3440 "-" "ELB-HealthChecker/2.0"
172.31.5.3 - - [20/Mar/2020:15:31:58 +0000] "GET / HTTP/1.1" 200 3440 "-" "ELB-HealthChecker/2.0"
172.31.32.106 - - [20/Mar/2020:15:32:26 +0000] "GET / HTTP/1.1" 200 3440 "-" "ELB-HealthChecker/2.0"
172.31.5.3 - - [20/Mar/2020:15:32:28 +0000] "GET / HTTP/1.1" 200 3440 "-" "ELB-HealthChecker/2.0"
172.31.32.106 - - [20/Mar/2020:15:32:56 +0000] "GET / HTTP/1.1" 200 3440 "-" "ELB-HealthChecker/2.0"
172.31.5.3 - - [20/Mar/2020:15:32:58 +0000] "GET / HTTP/1.1" 200 3440 "-" "ELB-HealthChecker/2.0"
172.31.32.106 - - [20/Mar/2020:15:33:26 +0000] "GET / HTTP/1.1" 200 3440 "-" "ELB-HealthChecker/2.0"
172.31.5.3 - - [20/Mar/2020:15:33:28 +0000] "GET / HTTP/1.1" 200 3440 "-" "ELB-HealthChecker/2.0"
172.31.32.106 - - [20/Mar/2020:15:33:56 +0000] "GET / HTTP/1.1" 200 3440 "-" "ELB-HealthChecker/2.0"
172.31.5.3 - - [20/Mar/2020:15:33:58 +0000] "GET / HTTP/1.1" 200 3440 "-" "ELB-HealthChecker/2.0"
172.31.32.106 - - [20/Mar/2020:15:34:26 +0000] "GET / HTTP/1.1" 200 3440 "-" "ELB-HealthChecker/2.0"
172.31.5.3 - - [20/Mar/2020:15:34:28 +0000] "GET / HTTP/1.1" 200 3440 "-" "ELB-HealthChecker/2.0"
172.31.32.106 - - [20/Mar/2020:15:34:56 +0000] "GET / HTTP/1.1" 200 3440 "-" "ELB-HealthChecker/2.0"
172.31.5.3 - - [20/Mar/2020:15:34:58 +0000] "GET / HTTP/1.1" 200 3440 "-" "ELB-HealthChecker/2.0"
172.31.32.106 - - [20/Mar/2020:15:35:26 +0000] "GET / HTTP/1.1" 200 3440 "-" "ELB-HealthChecker/2.0"
172.31.5.3 - - [20/Mar/2020:15:35:28 +0000] "GET / HTTP/1.1" 200 3440 "-" "ELB-HealthChecker/2.0"
172.31.32.106 - - [20/Mar/2020:15:35:56 +0000] "GET / HTTP/1.1" 200 3440 "-" "ELB-HealthChecker/2.0"
172.31.5.3 - - [20/Mar/2020:15:35:58 +0000] "GET / HTTP/1.1" 200 3440 "-" "ELB-HealthChecker/2.0"
172.31.32.106 - - [20/Mar/2020:15:36:26 +0000] "GET / HTTP/1.1" 200 3440 "-" "ELB-HealthChecker/2.0"
172.31.5.3 - - [20/Mar/2020:15:36:28 +0000] "GET / HTTP/1.1" 200 3440 "-" "ELB-HealthChecker/2.0"
172.31.32.106 - - [20/Mar/2020:15:36:56 +0000] "GET / HTTP/1.1" 200 3440 "-" "ELB-HealthChecker/2.0"
172.31.5.3 - - [20/Mar/2020:15:36:58 +0000] "GET / HTTP/1.1" 200 3440 "-" "ELB-HealthChecker/2.0"
172.31.32.106 - - [20/Mar/2020:15:37:26 +0000] "GET / HTTP/1.1" 200 3440 "-" "ELB-HealthChecker/2.0"
172.31.5.3 - - [20/Mar/2020:15:37:28 +0000] "GET / HTTP/1.1" 200 3440 "-" "ELB-HealthChecker/2.0"
172.31.32.106 - - [20/Mar/2020:15:37:56 +0000] "GET / HTTP/1.1" 200 3440 "-" "ELB-HealthChecker/2.0"
172.31.5.3 - - [20/Mar/2020:15:37:58 +0000] "GET / HTTP/1.1" 200 3440 "-" "ELB-HealthChecker/2.0"
79.127.104.75 - - [20/Mar/2020:15:38:23 +0000] "GET / HTTP/1.1" 200 11173 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_11_6) AppleWebKit/605.1.76 (KHTML, like Gecko) Version/9.1.2 Safari/601.7.7"
172.31.32.106 - - [20/Mar/2020:15:38:27 +0000] "GET / HTTP/1.1" 200 3440 "-" "ELB-HealthChecker/2.0"
172.31.5.3 - - [20/Mar/2020:15:38:28 +0000] "GET / HTTP/1.1" 200 3440 "-" "ELB-HealthChecker/2.0"
172.31.32.106 - - [20/Mar/2020:15:38:57 +0000] "GET / HTTP/1.1" 200 3440 "-" "ELB-HealthChecker/2.0"
172.31.5.3 - - [20/Mar/2020:15:38:58 +0000] "GET / HTTP/1.1" 200 3440 "-" "ELB-HealthChecker/2.0"
```

## Launch a second instance from the custom image

- Draw a diagram of the setup you have created showing the components (instances, database, load balancer, client) and how they are connected. Include the security groups as well.
  - Below, the diagram :



- Using the [Simple Monthly Calculator](#) calculate the monthly cost of this setup. You can ignore traffic costs. (Make sure you don't forget to include a component in the calculation. Also don't forget to uncheck the **Free Usage Tier** checkbox at the top.)
  - Below the total cost :

Services

Estimate of your Monthly Bill (\$ 50.21)

☒ Show First Month's Bill (include all one-time fees, if any)

Below you will see an estimate of your monthly bill. Expand each line item to see cost breakout of each service. To save this bill and input values, click on 'Save and Share' button. To remove the service from the estimate, jump back to the service and clear the specific service's form.

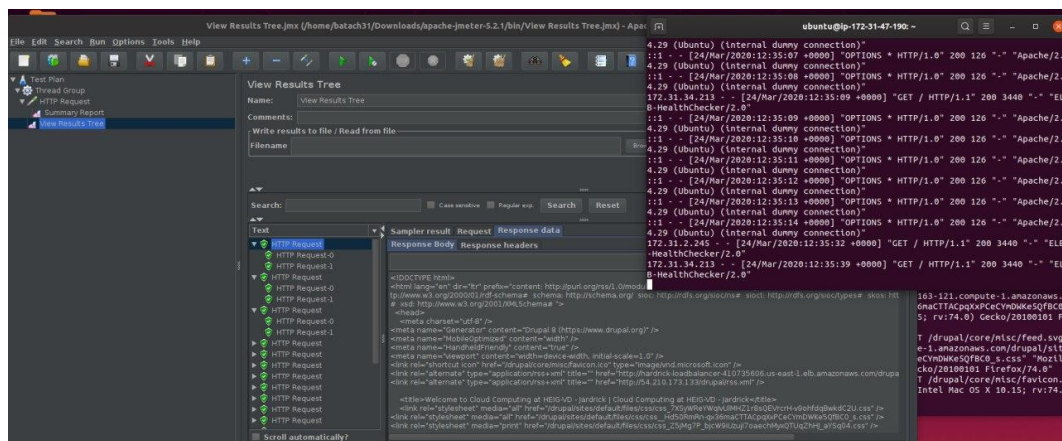
Export to CSV

Save and Share

<div> <div>Amazon EC2 Service (US East (N. Virginia))</div> <div>Compute:</div> </div>	<div>\$17.00</div>	<div>\$17.00</div>
<div> <div>Amazon RDS Service (US East (N. Virginia))</div> <div>DB Instances:</div> <div>Storage:</div> </div>	<div>\$12.45</div> <div>\$2.30</div>	<div>\$14.75</div>
<div> <div>Amazon Elastic Load Balancing (US East (N. Virginia))</div> <div>Classic LBs:</div> <div>Data Processed by Classic LBs:</div> </div>	<div>\$18.30</div> <div>\$0.16</div>	<div>\$18.46</div>
<div> <div>AWS Support (Basic)</div> <div>Support for all AWS services:</div> </div>	<div>\$0.00</div>	<div>\$0.00</div>
<div>Total Monthly Payment:</div>	<div>\$0.00</div>	<div>\$50.21</div>

## Test the distributed application

- Document your observations. Include screenshots of JMeter and the AWS console monitoring output.
  - You can find below our first load balancing test. The main instance receive all the requests



Then we edit the number of users (threads) that execute a request. You can find in the screenshot below the the result of JMeter test with 1000 users. As you can see, there is **4.46%** of errors.

Summary Report									
Name: Summary Report									
Comments:									
Write results to file / Read from file									
Filename									
Log/Display Only: Errors Successes Configure									
Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	Received Kbit/sec	Sent Kbit/sec
HTTP Request	1120	1904	453	8194	1380.48	4.46%	1.8/sec	17.24	0.58
TOTAL	1120	1904	453	8194	1380.48	4.46%	1.8/sec	17.24	0.58

To increase the errors and have a better representation of the load balancing, we execute the same test as before but every threads iterate 1000 times. You can see in the screenshot below the result of our test. As you can see, the errors rise to **99.91%**.



Summary Report

Name:

Summary Report

Comments:

Write results to file / Read from file

Filename

Browse...

Log/Display Only: 

ErrorsSuccesses

Configure

Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	Received KB/sec	Sent KB/sec	Avg. Bytes
HTTP Request	35009	26	0	23348	633.84	99.91%	1229.2/sec	2715.04	0.49	2261.9
TOTAL	35009	26	0	23348	633.84	99.91%	1229.2/sec	2715.04	0.49	2261.9

When the user receive a time out, he gets an empty error has shown below :

View Results Tree

Name: View Results Tree

Comments:

Write results to file / Read from file

Filename

Browse... Log/Display Only: ☐ Errors ☐ Successes

Search:  ☐ Case sensitive ☐ Regular exp.

Text

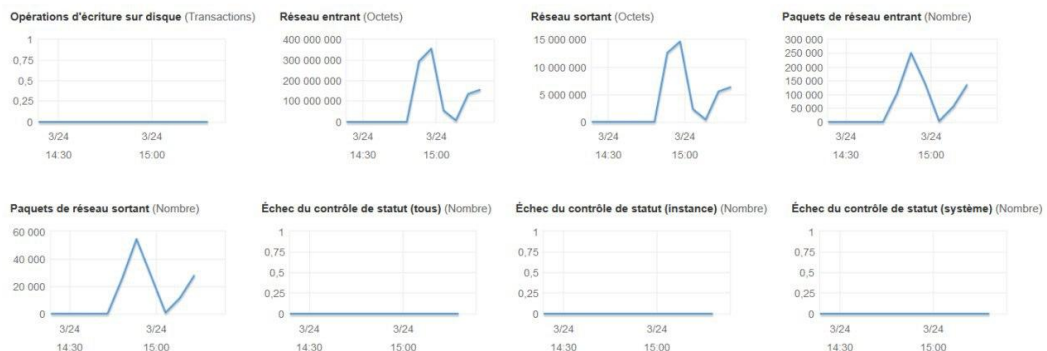
Sampler result Request Response data

Thread Name:Thread Group 1-250  
Sample Start:2020-03-24 08:21:09 PDT  
Load time:0  
Connect Time:0  
Latency:0  
Size in bytes:2254  
Sent bytes:0  
Headers size in bytes:0  
Body size in bytes:2254  
Sample Count:1  
Error Count:1  
Data type ("text"/"bin"/""):text  
Response code:Non HTTP response code: java.net.UnknownHostException  
Response message:Non HTTP response message: hardrick-loadBalancer-410735606.us-east-1.elb.amazonaws.com

HTTPSampleResult fields:  
ContentType:  
DataEncoding: null

Raw Parsed

By monitoring the instances, we can see that instance send a response to all packets received. This proves the packets time out.



- When you resolve the DNS name of the load balancer into IP addresses while the load balancer is under high load what do you see? Explain.
  - As you can see in the screenshot below, we have a time out. The load balancer is overwhelmed and can no longer handle new connections. So, the server will manage the connection already established and ignores new incoming connections.

```
(base) batach31@ubuntu:~/Desktop$ nslookup hardrick-loadBalancer-410735606.us-east-1.elb.amazonaws.com
;; connection timed out; no servers could be reached
```

- Did this test really test the load balancing mechanism? What are the limitations of this simple test? What would be necessary to do realistic testing?
  - We haven't really test the load balancing mechanism because only the main instance seems to have some requests. We should define some rules in the load balancer to indicate when the load balancer have to send some requests to the second instance.