Deploying a Web Application on AWS

This lab will provide experience in using multiple AWS services to deploy a web application:

- The web application will be hosted on an**Amazon EC2** instance using PHP
- Data will be stored in an**Amazon DynamoDB** table
- Images will be served from**Amazon S3**

After completing this lab, you will be able to:

- Create an **IAM Role** for use by an Amazon EC2 instance
- Create an **Amazon S3** Bucket
- Create an **Amazon DynamoDB** table
- Create an **Amazon Virtual Private Cloud (VPC)**with an Internet Gateway and a Public Subnet
- Deploy a web application on **Amazon EC2**

Duration

This lab will require approximately**45 minutes** to complete.

Accessing the AWS Management Console

Launch the lab using the Cloudformation template as per the instructions from instructor.

Task 1: Create an IAM Role

In this task, you will create an**IAM Role** and give it permission to access Amazon S3 and Amazon DynamoDB. You will later assign this role to an Amazon EC2 instance that will run your web application.

An **IAM Role** is similar to a User, in that it is given permissions to call AWS services. However, instead of belonging to one person, a role can be *assumed* by an authorized user, service or application. When this is done, **access credentials** are dynamically created, which can then be used to make API calls to AWS services. This is how applications running on an Amazon EC2 instance can obtain credentials to call AWS services such as Amazon S3 and Amazon DynamoDB.

4. [4]In the **AWS Management Console**, on the **Services** menu, click **IAM**.
5. [5]In the left navigation pane, click**Roles**.
6. [6]Click **Create role**.
7. [7]Under **Select role type**, choose **AWS service**, then click **EC2**.
8. [8]Under **Select your use case**, select **EC2** *(Allows EC2 instances to call AWS services on your behalf)*

This indicates that the role will be used by an Amazon EC2 instance.

9. [9]Click **Next: Permissions**.

10. [10]Select (tick) these two roles from the list (you can use the Filter box to locate them quickly):

- `AmazonS3FullAccess`
- `AmazonDynamoDBFullAccess` (Choose the first one that appears)

11. [11]Click **Next: Review**.
12. [12]For **Role name**, type: `WebServerRole`
13. [13]Click **Create role**.

You will use this role later in the lab when you launch an Amazon EC2 instance. The web application running on that instance will then have permission to use Amazon S3 and Amazon DynamoDB.

Task 2: Create an Amazon S3 Bucket

In this task, you will create an**Amazon S3 bucket**. This bucket will be used by your web application to serve images and other static files on web pages.

14. [14]On the **Services** menu, click **S3**.
15. [15]Click **Create bucket**.
16. [16]For **Bucket name**, type a unique bucket name starting with: `webapp-`
    For example, add your initials and some numbers like this:*webapp-cb35*

Ensure your bucket name begins with `webapp-` otherwise the application will be unable to find it.

17. [17]Click **Create**.

Your web application will later copy files into this bucket.

Task 3: Create an Amazon DynamoDB table

In this task, you will create an**Amazon DynamoDB table** that will hold data for the web application.

18. [18]On the **Services** menu, click **DynamoDB**.
19. [19]Click **Create table**.
20. [20]For **Table name**, type `AWS-Services`

Make sure you type the table name exactly as it appears above, including case!

21. [21]In the text box below**Partition key**, type `Category`

Make sure you type the name exactly as it appears above, including case!

22. [22]Select **Add sort key**.
23. [23]In the new text box that appears, type`Name`

Make sure you type the name exactly as it appears above, including case!

24. [24]Click **Create**.

A DynamoDB table has now been created. Your web application will later copy data into this table.

You can continue to the next task. The table will be created in the background.

Task 4: Create an Amazon VPC

In this task, you will create a basic**Amazon Virtual Private Cloud (VPC)**with a single Public Subnet. You will attach an Internet Gateway to the VPC so that resources in the VPC can communicate with the Internet.

Task 4.1: Create a VPC

25. [25]On the **Services** menu, click **VPC**.
26. [26]In the left navigation pane, click **Your VPCs**.
27. [27]Click **Create VPC**.
28. [28]In the **Create VPC** dialog box, configure the following settings (and ignore any settings that aren’t listed):

Name tag	`Lab VPC`
IPv4 CIDR block	`10.200.0.0/16`

29. [29]Click **Yes, Create**.

The **10.200.0.0/16** CIDR ("Classless Inter-Domain Routing") block defines a range of IP addresses for the VPC. The **/16** ending means that the block contains all IP addresses that match**10.200..** addresses.

Task 4.2: Create a Subnet

A **Subnet** is a segment of a VPC which contains resources such as Amazon EC2 instances. Each subnet receives a subset of the IP address range that was assigned to the VPC.

30. [30]In the navigation pane, click **Subnets**.
31. [31]Click **Create Subnet**.
32. [32]In the **Create Subnet** dialog box, configure the following settings (and ignore any settings that aren’t listed):

Name tag	`Public Subnet 1`
VPC	**Lab VPC**
Availability Zone	Select the first zone listed

IPv4 CIDR block	`10.200.10.0/24`

33. [33]Click **Yes, Create**.

A *Public Subnet* is a subnet that can communicate with the Internet. This is where you can place resources that should be publicly accessible. The Subnet has been named **Public Subnet 1**, but it is not yet public. To make it public, the VPC must be connect to an Internet Gateway.

Task 4.3: Create an Internet Gateway

You will now create an**Internet Gateway**, which is the interface between a VPC and the Internet.

34. [34]In the navigation pane, click **Internet Gateways**.
35. [35]Click **Create Internet Gateway**.
36. [36]For **Name tag**, type: `Lab Gateway`
37. [37]Click **Yes, Create**.
38. [38]Select the newly created **Lab Gateway**, and then click **Attach to VPC**.
39. [39]In the **Attach to VPC** dialog box, for **VPC**, select **Lab VPC**.
40. [40]Click **Yes, Attach**.

The Internet Gateway is now attached to your VPC but is not yet being used. You will need to configure a Route Table to direct network traffic to the Internet Gateway.

Task 4.4: Create a Public Route Table

A **Route Table** contains a set of rules called**Routes** that direct network traffic flowing in and out of a subnet. New VPCs automatically receive a Route Table that only routes network traffic *within* the VPC. This is called a *Private Route Table*.

You will create a *Public Route Table* that allows resources to communicate with the Internet via the Internet Gateway.

41. [41]In the navigation pane, click **Route Tables**.
42. [42]Click **Create Route Table**.
43. [43]In the **Create Route Table** dialog box, configure the following settings (and ignore any settings that aren't listed):

Name tag	`Public Route Table`
VPC	**Lab VPC**

44. [44]Click **Yes, Create**.
45. [45]Select the newly created **Public Route Table**, and then click the **Routes** tab in the lower pane of the console. (You can drag the lower pane upwards to make it larger.)
46. [46]Click **Edit**.
47. [47]Click **Add another route**.

You will configure the Route Table to send Internet traffic (identified with the wildcard **0.0.0.0/0**) through the Internet Gateway.

48. [48]For **Destination**, type: `0.0.0.0/0`
49. [49]Click in the **Target** box, and then click the **Lab Gateway** that you created earlier (the ID starts with *igw-*).
50. [50]Click **Save**.

You can now associate the *Public Route Table* with the Subnet you created earlier.

51. [51]With **Public Route Table** still selected, click the **Subnet Associations** tab at the bottom of the screen.
52. [52]Click **Edit**.
53. [53]Select (tick) the checkbox beside **Public Subnet 1**.
54. [54]Click **Save**.

Your subnet is now a *Public Subnet*, which means that resources in the subnet can communicate with the Internet.

Task 5: Launch an Amazon EC2 Instance

In this task, you will create an **Amazon EC2 instance** in the Public Subnet. The term *instance* refers to a virtual machine running in the cloud. You will configure your instance to install a web application when it starts.

The application will communicate with your **Amazon S3 Bucket** and your **Amazon DynamoDB table**. These three services will provide the web application with compute, storage, and database services.

55. [55]On the **Services** menu, click **EC2**.
56. [56]Click **Launch Instance**.

When launching a new instance, you first select an Amazon Machine Image (AMI), which is a preconfigured template for an instance in the cloud. You will be selecting an image that contains the Linux operating system.

57. [57]In the second row **Amazon Linux AMI 2018.03.0 (HVM) **, click **Select**

On the **Choose an Instance Type** page, you can select the *Instance Type*, which determines how much RAM and CPU will be allocated to your instance.

58. [58]Click **Next: Configure Instance Details** to accept the default (**t2.micro**).
59. [59]In the **Configure Instance Details** page, configure the following settings (and ignore any settings that aren’t listed):

Network	**Lab VPC**
Subnet	**Public Subnet 1**
Auto-assign Public IP	**Enable**

IAM role	WebServerRole

Confirm that all the above settings have correctly selected otherwise the web application will not work correctly.

60. [60]Click **Advanced Details** to expand it (you might need to scroll down to see it).

When a script is provided in the **User Data** field, the Amazon EC2 instance will automatically run the script the first time the instance is started. This is an easy way to install software and configure the instance.

You will use a script that automatically loads and configures the web application. This type of script would normally be maintained in a Source Code Repository such as Amazon CodeCommit so that it is correctly stored and version-tracked.

61. [61]Click **Copy Code Block** below, and paste it into the**User Data** field. To be on the safer side you can copy the bash script from bottom of the Lab01-webapp-CommandRef.txt.

```
#!/bin/bash
# Install Apache Web Server and PHP
yum remove -y httpd php
yum install -y httpd24 php56
# Download Lab files
wget https://us-west-2-aws-training.s3.amazonaws.com/awsu-ilt/AWS-100-ARC/v5.2/lab-1-web
unzip lab1src.zip -d /tmp/
mv /tmp/lab1src/*.php /var/www/html/
# Download and install the AWS SDK for PHP
wget https://github.com/aws/aws-sdk-php/releases/download/3.15.9/aws.zip
unzip aws -d /var/www/html
# Determine Region
AZ=`curl --silent http://169.254.169.254/latest/meta-data/placement/availability-zone/`
REGION=${AZ::-1}
# Copy files to Amazon S3 bucket with name webapp-*
BUCKET=`aws s3api list-buckets --query "Buckets[?starts_with(Name, 'webapp-')].Name | [0
aws s3 cp /tmp/lab1src/jquery/ s3://$BUCKET/jquery/ --recursive --acl public-read --regi
aws s3 cp /tmp/lab1src/images/ s3://$BUCKET/images/ --recursive --acl public-read --regi
aws s3 ls s3://$BUCKET/ --region $REGION --recursive
# Configure Region and Bucket to use
sed -i "2s/%region%/$REGION/g" /var/www/html/*.php
sed -i "3s/%bucket%/$BUCKET/g" /var/www/html/*.php
# Copy data into DynamoDB table
aws dynamodb batch-write-item --request-items file:///tmp/lab1src/scripts/services1.json
aws dynamodb batch-write-item --request-items file:///tmp/lab1src/scripts/services2.json
aws dynamodb batch-write-item --request-items file:///tmp/lab1src/scripts/services3.json
# Turn on web server
chkconfig httpd on
service httpd start
```

Examine the script. It is performing the following steps:

- Install Apache web server (httpd) and the PHP language
- Download and unzip a file containing scripts for the web application
- Download and install the AWS SDK for PHP
- Copy files to the Amazon S3 bucket that has a name starting with*webapp-*
- Copy data into the DynamoDB table
- Turn on the web server

A User Data script will only run the first time that an instance is started, making it an ideal way to configure instances and load initial data.

62. [62]Click **Next: Add Storage**.

You do not require additional storage on this instance, so you are leaving the settings as their default.

63. [63]Click **Next: Add Tags**.

Tags are *metadata* associated with an instance. They can be used to name an instance, associate it with cost centers and identify its purpose.

64. [64]Click **Add Tag**.
65. [65]Under **Key**, type `Name`
66. [66]Under **Value**, type: `Web Server`
67. [67]Click **Next: Configure Security Group**.

You will now define a *Security Group*, which acts as a virtual firewall that controls traffic to the instance. Your instance will be a web server, so you will need to permit access via port 80 (HTTP).

68. [68]Specify the following:

Security group name	`Web Server`
Description	`Web Server Security Group`

There should be an existing**SSH** rule. Leave that rule as-is.

69. [69]Click **Add Rule**.
70. [70]In the *new row* that appears, specify the following:

Type	**HTTP**
Source	**Anywhere**

71. [71]Click **Review and Launch**.
72. [72]Review the settings, then click **Launch**.
73. [73]When prompted, accept the *qwikLABS* keypair, select the acknowledgement check box, then click **Launch Instances**.
74. [74]Click **View Instances** (you might need to scroll down to see it).

Your Amazon EC2 instance will take a few minutes to start. It will then run the*User Data script*, which will install software and activate the web application.

75. [75]Wait for the **Web Server** to show **Status Checks:** ***2/2 checks passed***

This will take a few minutes. You can occasionally click the refresh icon in the top-right corner.

Your instance is now ready for you to access!

Task 6: Examine your Resources

In this task, you will access the web application and examine the resources loaded by the User Data script.

76. [76]Select the **Web Server** instance.
77. [77]On the **Description** tab in the lower pane, copy the**IPv4 Public IP** to your clipboard.
78. [78]Open a new browser tab, paste the IP address from your clipboard and hit Enter.

The web application should appear.

79. [79]In the web application, click any category to reveal service icons.

You will see:

- A list of services, which has been retrieved from the**DynamoDB table** the you created
- Service icons being served from the**Amazon S3 bucket** that you created

If your web application does not work, or if the images are not correctly appearing, ask your instructor for assistance in diagnosing the configuration.

The images displayed in the web application are being served directly from your Amazon S3 bucket.

80. [80]Right-click an icon and choose "Copy image address" (the wording in your browser might vary).
81. [81]Paste the link into a **new browser tab** and notice that the URL is referencing your Amazon S3 bucket.

The service names were retrieved from your Amazon DynamoDB table.

82. [82]Return to the AWS Management Console tab in your browser.
83. [83]On the **Services** menu, click **DynamoDB**.
84. [84]In the navigation pane, click **Tables**.
85. [85]Click your **AWS-Services** table.
86. [86]Click the **Items** tab.

The items in the DynamoDB table will be displayed. These items were loaded by the User Data script that ran when your instance started.

Optional: Try the Challenge Quiz!

87. [87]Return to the web application. In the top-right corner of the webpage, click the**challenge me** button.

Your web application includes a Quiz that uses the resources in your Amazon S3 bucket and the items in your DynamoDB table. Take the challenge by dragging the Services listed in the left container and drop them in the correct Service category in the right container. Good luck!

(**Hint**: They can be grouped by color!)

Lab Complete.

Congratulations! You have completed the lab.