Deploying a Web Application on AWS

using Amazon EC2

using Amazon EC2

Task 4: Test the Application

Task 3: Launch an Application Server using Amazon EC2

Task 4: Test the Application

Task 3: Launch an Application Server using Amazon EC2

Task 4: Test the Application

Task 3: Launch an Application Server using Amazon EC2

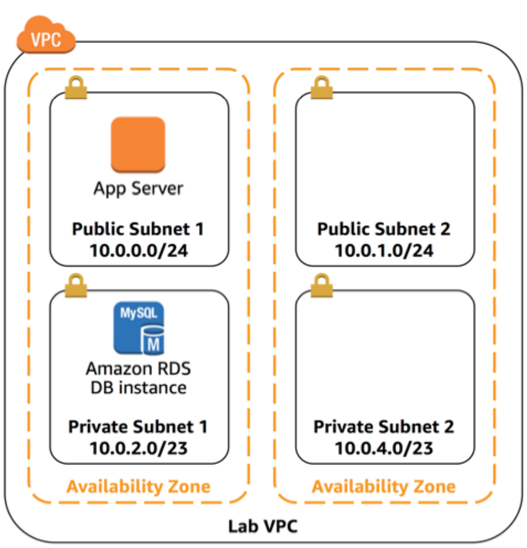
Task 4: Test the Application

Traditional methods of deploying servers and configuring security are complex and often involve multiple teams and long delays. Fortunately, it is quick and easy to deploy secure infrastructure in the cloud.

In this lab you will:

- Launch a database using **Amazon RDS**
- Launch an application server using **Amazon EC2**
- Automatically install an **application**

The final architecture will be:



Duration

This lab will require approximately **30 minutes** to complete.

Accessing the AWS Management Console

⚠ **Windows Users:** Please use Chrome or Firefox as your web browser for this lab. The lab instructions are **not compatible with *Internet Explorer*** due to a difference in the Amazon RDS console.

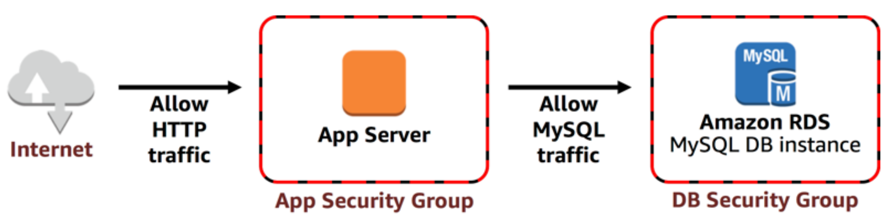
- Click **Start Lab** to launch the lab.
- Click **Open Console**
- Sign in to the AWS Management Console using the credentials shown to the left of these instructions.

⚠ Please do not change the Region during this lab.

Task 1: Configure Security

Security should be implemented at every layer of your architecture — in the application, on the server, within the network and when connecting to the internet.

In this task, you will define **Security Groups** for the Amazon EC2 application server and Amazon RDS database instance:



A *security group* acts as a virtual firewall that controls the traffic for one or more instances. When you launch an instance, you associate one or more security groups with the instance. You add *rules* to each security group that allow traffic to or from its associated instances. You can modify the rules for a security group at any time; the new rules are automatically applied to all instances that are associated with the security group.

First, you will create the **App Security Group**. It will be configured to permit incoming HTTP connections from the Internet.

1. In the **AWS Management Console**, on the **Services** menu, click **EC2**.
2. In the left navigation pane, click **Security Groups**.

   Some existing security groups will be displayed. You will now create a new security group for the application servers.

3. Click **Create Security Group** and configure:

   - **Security group name:** `App-SG`
   - **Description:** `Allow HTTP access`
   - **VPC:** *Lab VPC*

4. Click **Add Rule** then configure:

   - **Type:** *HTTP*
   - **Source:** *Anywhere*
   - Click **Create**

   You will use this Security Group later in the lab when launching an application server.

   Next, you will configure the **Database Security Group**. It will be configured to permit incoming database connections from the App Server.

   First, you will copy the ID of the *App-SG* that you just created.

5. Select ☑ **App-SG**.
6. Copy the **Group ID** displayed in the **Description** tab in the lower half of the page.

Tip: Simply hover over the Group ID and click Copy.

It will look similar to: *sg-1234abcd*

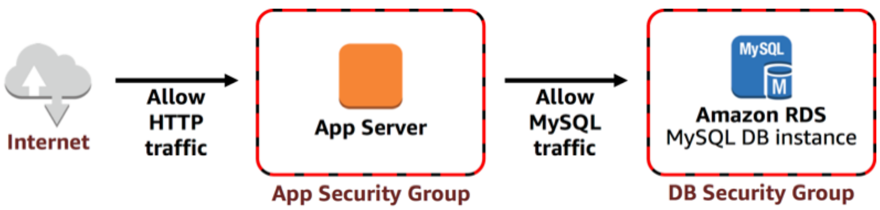
7. Click **Create Security Group** and configure:

   - **Security group name:** `DB-SG`
   - **Description:** `Allow DB access`
   - **VPC:** *Lab VPC*
   - Click **Add Rule** then configure:

     - **Type:** *MYSQL/Aurora*
     - **Source:** Paste the *Group ID* that you copied in the previous step

   - Click **Create**

This configuration means that *the Database security group (DB-SG) is permitting Inbound access from the Application security group (App-SG)*.



The ability for one security group to refer to another security group is a powerful capability. It means that additional EC2 instances can be granted access to the database by simply associating them with *App-SG*. Any instance associated with the *App-SG* will then be permitted to communicate to the database (or, more accurately, to any database associated with the *DB-SG* security group).

You will use this Database security group in the next task.

Task 2: Create an Amazon RDS Database

Database

Traditionally, creating a database can be a complex process requiring either a Database Administrator or a Systems Administrator. In the cloud, the task is made simple by using Amazon Relational Database Service (Amazon RDS).

In this task, you will create a MySQL database in your VPC. MySQL is a popular open-source relational database management system (RDBMS), so there are no software licensing fees.

⚠ **Windows Users:** Please use Chrome or Firefox as your web browser for this lab. The lab instructions are **not compatible with *Internet Explorer*** due to a difference in the Amazon RDS console.

8. On the **Services ⌄** menu, click **RDS**.

9. In the left navigation pane, click **Databases**.

10. Click **Create database**

11. On the **Select engine** page:

    - Select ⦿ **MySQL**
    - Click **Next**

    Several use-cases will be presented, ranging from enterprise-class databases through to Dev/Test systems. You will also see mention of **Amazon Aurora**, which is a MySQL-compatible system that has been re-architected for the cloud. If your company uses large-scale MySQL or PostgreSQL databases, Amazon Aurora can provide enhanced performance.

12. On the **Choose use case** page:

    - Select ⦿ **Dev/Test - MySQL**
    - Click **Next**

    You can now select a database configuration, including software version, instance class, storage and login settings. The **Multi-AZ deployment** option can automatically create a replica of the database in a second Availability Zone for High Availability. In this lab, however, you will use a single database instance.

13. On the **Specify DB details** page, configure:

- **DB instance class:** *db.t2.micro* (at the top of the list)
- **DB instance identifier:** `inventory-db`
- **Master username:** `master`
- **Master password:** `lab-password`
- **Confirm password:** `lab-password`
- Click **Next**

You can next **Configure Advanced Settings**, including networking, security, backup and monitoring.

14. In the **Network & Security** section, configure:

- **Virtual Private Cloud (VPC):** *Lab VPC*
- **Subnet group:** *lab-db-subnet-group* (*This was created for you during lab setup*)
- **VPC security groups:**
    - Click ⦿ **Choose existing VPC security groups**
    - Add **DB-SG** (which you created earlier)
    - Remove ✖ **default**

15. In the **Database options** section, configure:

- **Database name:** `inventory`

This is the logical name of the database that will be used by the application.

Feel free to look at the many other options displayed on the page, but leave them set to their default values. Options include automatic backups, the ability to export log files and automatic version upgrades. The ability to activate such features via tick-boxes shows the power of using a *fully-managed database solution* rather than having to install, backup and maintain the database yourself.

16. Click **Create database** (at the bottom of the page).

You will receive a message that Your DB instance is being created.

This will take a few minutes but **there is no need to wait**. Please continue with the next task.

Task 4: Test the Application
Task 3: Launch an Application Server using Amazon EC2
Task 4: Test the Application
Lab Complete
Task 3: Launch an Application Server using Amazon EC2
Task 4: Test the Application
Lab Complete
Task 2: Create an Amazon RDS Database
Task 3: Launch an Application Server using Amazon EC2
using Amazon EC2
Task 4: Test the Application
Task 3: Launch an Application Server using Amazon EC2
using Amazon EC2
Task 4: Test the Application
Task 3: Launch an Application Server using Amazon EC2
using Amazon EC2
using Amazon EC2
using Amazon EC2
Task 4: Test the Application
Lab Complete
Task 2: Create an Amazon RDS Database
Task 3: Launch an Application Server using Amazon EC2
using Amazon EC2
using Amazon EC2
Task 4: Test the Application
Task 3: Launch an Application Server using Amazon EC2
Task 4: Test the Application
Task 3: Launch an Application Server using Amazon EC2
Task 4: Test the Application

Task 3: Launch an Application Server using Amazon EC2

You are now ready to launch an **Amazon EC2 instance** to run the application. As part of the launch, you will provide a *configuration script* that will automatically install an application. You will also associate the instance with the *Application security group* that you created earlier in the lab. This will permit HTTP (web) access from the Internet.

17. On the **Services** menu, click **EC2**.

18. Click **Launch Instance**

Step 1: Choose an Amazon Machine Image (AMI)

This step allows you to choose an AMI, which contains a copy of the disk volume that will be used to launch the instance.

💬 Examine the list of AMIs that are displayed, showing many versions of Microsoft Windows and Linux. These disk images are regularly updated to incorporate security patches and software that helps you use AWS services. You can also create your own AMI that includes your own data and applications, or you can select pre-built commercial applications from the **AWS Marketplace**.

Your application will use *Amazon Linux 2*.

19. Beside the **Amazon Linux 2 AMI** in the top row, click **Select**

Step 2: Choose an Instance Type

This step allows you to choose an **Instance Type**, which determines the resources that will be allocated to your EC2 instance. Each Instance Type allocates a combination of virtual CPUs, memory, disk storage and network performance.

Instance Types are divided into **families** such as Compute-optimized, Memory-optimized and Storage-Optimized. The name of the instance Type includes a family identifier, such as **t2** and **m4**. The number indicates the *generation* of the instance, so **m5** is newer than **m4**.

Your application will use a **t2.micro** Instance Type, which is a small instance that can burst above baseline performance when it is busy. It is ideal for development, testing and for applications that have bursty workloads.

20. Select ☑ **t2.micro**

21. Click **Next: Configure Instance Details**

Step 3: Configure Instance Details

This step allows you to configure instance details, such as the number of instances to launch and the network configuration. You can hover over the ℹ icons to view a description of each field.

You will launch the instance in a public subnet within the *Lab VPC* network.

22. Configure these settings:

You will launch the instance in a public subnet within the *Lab VPC* network.

22. Configure these settings:

- **Network:** *Lab VPC*
- **Subnet:** *Public Subnet 1* (Make sure it says *Public*)

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": "ssm:*",
            "Resource": "arn:aws:ssm:*:*:parameter/inventory-
app/*",
            "Effect": "Allow"
        }
    ]
}
```

In this case, the role grants permission to access the *inventory-app* settings within the *AWS Systems Manager Parameter Store*, which will be used to store configuration settings.

23. Scroll down, then expand ▸ **Advanced Details**.

    A field for **User data** will appear.

    When you launch an instance, you can pass a configuration script via the *User data* field. The script can be used to perform configuration tasks and install software.

Lab Complete