AWS Introduction

AWS Overview

What is Cloud Computing?

Cloud computing is the on-demand delivery of compute power, database storage, applications and other IT resources through a cloud services platform via the Internet with pay-as-you-go pricing.

Cloud computing provides a simple way to access servers, storage, databases and a broad set of application services over the Internet.

6 Advantages of Cloud Computing

1. Trade capital expense for variable expense.
2. Benefit from massive economies of scale.
3. Stop guessing about capacity.
4. Increase speed and agility.
5. Eliminate overhead cost of maintaining data centers
6. Go global in minutes.

Trade capital expense for variable expense

Instead of having to invest heavily in data centers and servers before you know how you're going to use them, you can pay only when you consume computing resources, and pay only for how much you consume.

Benefit from massive economies of scale

By using cloud computing, you can achieve a lower variable cost than you can get on your own. Because usage from hundreds of thousands of customers is aggregated in the cloud, providers such as AWS can achieve higher economies of scale, which translates into lower pay as-you-go price.

Stop guessing about capacity

Eliminate guessing on your infrastructure capacity needs. When you make a capacity decision prior to deploying an application, you often end up either sitting on expensive idle resources or dealing with limited capacity.

Increase speed and agility

In a cloud computing environment, new IT resources are only a click away, which means that you reduce the time to make those resources available to your developers from weeks to just minutes.

Eliminate overhead cost of maintaining data centers

Cloud computing lets you focus on your own customers, rather than on the heavy lifting of racking, stacking, and powering servers (infrastructure).

Go global in minutes

Easily deploy your application in multiple regions around the world with just a few clicks. This means you can provide lower latency and a better experience for your customers at minimal cost.

Paying for Cloud Services

AWS offers you a **pay-as-you-go** approach for pricing for over 160 cloud services. With AWS you pay only for the individual services you need, for as long as you use them, and without requiring long-term contracts or complex licensing.

- You can also use AWS Billing and Cost Management

3 Models of Cloud Computing

Infrastructure as a Service (IaaS)

Infrastructure as a Service (IaaS) is a self-service model for managing remote data center infrastructures. AWS offers IaaS in the form of data centers.

Platform as a Service (PaaS)

Platform as a Service (PaaS) allows organizations to build, run and manage applications without the IT infrastructure. This makes it easier and faster to develop, test and deploy applications.

Software as a Service (SaaS)

Software as a service (SaaS) replaces the traditional on-device software with software that is licensed on a subscription basis. It is centrally hosted in the cloud. A good example is Salesforce.com.

Other Major Cloud Providers

- Microsoft Azure
- Google Cloud Platform (GCP)

AWS RDS

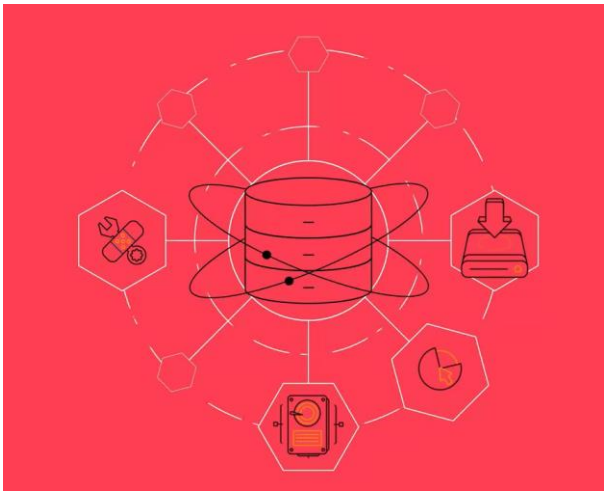
RDS Overview

The following overview will cover:

- What is Amazon RDS?
- Why use RDS?
- Regions & Availability Zones
- Security
- How to interact with Amazon RDS
- Using AWS RDS vs. Installing a database on an AWS EC2 Instance

What Is Amazon Relational Database Service (Amazon RDS)?

Amazon Relational Database Service (Amazon RDS) is a web service that makes it easier to set up, operate, and scale a relational database in the AWS Cloud. It provides cost-efficient, resizable capacity for an industry-standard relational database and manages common database administration tasks.



- RDS **automates** expensive and time consuming tasks such as managing backups, software patching, automatic failure detection, and recovery.
- You can help control who can access your RDS databases by using **AWS Identity and Access Management (IAM)** to define users and permissions.
- RDS is available on several **database instance types**. **Instance types** comprise varying combinations of CPU, memory, storage, and networking capacity and give you the flexibility to choose the appropriate mix of resources for your database.
    - Examples of instance types include `T3`, `T2`, `M6g`, `M5`, etc...You can read more about instance types here.

- RDS is **free to try** and you will be charged based on how much computational power you use per month (pay-as-you-go).
- RDS provides you with six familiar database engines to choose from, including Amazon Aurora, PostgreSQL, MySQL, MariaDB, Oracle Database, and SQL Server.

Why Use RDS?

It's important to distinguish AWS RDS from *other database solutions offered through AWS*.

- AWS offers 15 database engines including *relational, key-value, document, in-memory, graph, time series, and ledger databases*.
- With **RDS**, you don’t need to worry about database management tasks such as server provisioning, patching, setup, configuration, backups, or recovery. The RDS manages this for you.
- **RDS** is a **relational** database service, therefore it organizes data within tables in rows and columns. Compare this to non-relational or NoSQL databases which use different mechanisms to store and retrieve data through key-value pairs, document models, etc.

See the image below to see how AWS RDS differs from other databases available in AWS.

Regions & Availability Zones

- An **AWS Region** is a highly available data center that houses Amazon cloud computing resources in different areas of the world (for example, North America, Europe, or Asia).
- Each AWS Region contains multiple distinct locations called **Availability Zones**, or AZs.
- Each Availability Zone is engineered to be isolated from failures in other Availability Zones. Each is engineered to provide inexpensive, low-latency network connectivity to other Availability Zones in the same AWS Region.
- By launching instances in separate Availability Zones, you can protect your applications from the failure of a single location. This makes AWS fault-tolerant.
    - **Fault-tolerance** defines the ability for a system to remain in operation even if some of the components used to build the system fail.

Security

- A **security group** controls the access to a DB instance. It does so by allowing access to IP address ranges or Amazon EC2 instances that you specify.

- You can set security groups when configuring your RDS instance.

How to Interact with Amazon RDS

There are several ways that you can interact with Amazon RDS.

1. AWS Management Console

You can manage your DB instances from the console with no programming required. To access the Amazon RDS console, sign in to the AWS Management Console and open the Amazon RDS console at https://console.aws.amazon.com/rds/

2. Command Line Interface

You can use the AWS Command Line Interface (AWS CLI) to access the Amazon RDS API interactively. Install the AWS CLI here. To begin using the AWS CLI with RDS, view the AWS CLI Command Reference.

3. Programmatically Accessing Amazon RDS

While developing an application, developers may use the *AWS Software Development Kits (SDKs)* and utilize the RDS Application Programming Interface (API) to automate tasks for managing DB instances and other objects in Amazon RDS.

Using AWS RDS vs. Installing a database on an AWS EC2 Instance

DB services (like RDS) are *managed* services, meaning that you have limited control over the actual database. (I.e. you can't install particular software you may need).

- **EC2 (Elastic Cloud Compute)**, on the other hand, gives you maximum control over the software stack, database, and operating system.
- **EC2** allows you compute capacity and `SUPER` privileges, increasing flexibility.
- **RDS** is more cost-effective and is automated, but limits your control over the database.
- **RDS** takes care of your database from end-to-end by managing, maintaining, and securing it, eliminating overhead costs and the role of a DBA (Database Administrator).

For a full comparison of AWS RDS vs. EC2 see this article.
Find the EC2 overview module here.

References

- Databases on AWS
- RDS Instance Types

AWS Elastic Compute Cloud (EC2)

Amazon Elastic Compute Cloud (Amazon EC2) is a web service that provides secure, resizable compute capacity in the cloud. It is designed to make web-scale cloud computing easier for developers.

Amazon EC2 provides:

- Virtual computing environments, known as instances
- Preconfigured templates for your instances, known as **Amazon Machine Images (AMIs)**, that package the bits you need for your server (including the operating system and additional software)
- Various configurations of CPU, memory, storage, and networking capacity for your instances, known as **instance types**
- Secure login information for your instances using key pairs (AWS stores the public key, and you store the private key in a secure place)
- A firewall that enables you to specify the protocols, ports, and source IP ranges that can reach your instances using **security groups**

Elastic Web-Scale Computing

You can increase or decrease capacity within minutes and commission one to thousands of instances simultaneously.

Completely Controlled

AWS EC2 give you complete control including root access to each instance and can stop and start instances without losing data and use web service APIs.

Flexible Cloud Hosting Services

You can choose from multiple instance types & operating systems as well as instances with varying memory, CPU, and storage configurations.

Elasticity

The “Elastic” nature of the service allows developers to instantly scale to meet spikes in traffic or demand. When computing requirements unexpectedly change (up or down), Amazon EC2 can instantly respond, meaning that developers have the ability to control how many resources are in use at any given point in time.

Instances

An instance is a virtual server in the cloud. Its configuration at launch is a copy of the AMI that you specified when you launched the instance.

You can launch different types of instances from a single AMI. An instance type essentially determines the hardware of the host computer used for your instance. Each instance type offers different compute and memory capabilities. Select an instance type based on the amount of memory and computing power that you need for the application or software that you plan to run on the instance. For more information about the hardware specifications for each Amazon EC2 instance type, see Amazon EC2 Instance Types.

Create an EC2 Instance

1. Sign into AWS as a root user. Go to **EC2 Dashboard**. Select `Launch Instance`.
2. Choose an **Amazon Machine Image (AMI)** - select `Amazon Linux 2 AMI (HVM), SSD Volume Type` - 64 bit (x86).
3. Choose an **Instance Type** - select t2 micro (Free tier eligible)
4. Skip through steps 3-5 until you get to `6. Configure Security Group`.
5. **Configure Security Group**: configure your security group so that you can access your EC2 instance from SSH - change `Source` to `Anywhere`.
6. Proceed to launch until you are prompted to `Select an existing key pair or create a new key pair`.
7. Name your key pair `mynewkeypair`. Click `Download`, then click `Launch Instance`.
8. Your EC2 Instance has now been launched. Go to the SSH Into EC2 Instance to learn how to connect to your EC2 through SSH client.

References

- AWS EC2 Documentation

SSH Into EC2 Instance

How to SSH into an EC2

Your EC2 instance will be running in a public subnet. Your EC2 instance has a public address on the internet which you are able to access from wherever you have internet access.

The relevant protocol by which we are able to access the instance is by using **Secure Shell (SSH)** which is the **TCP protocol port 22**.

1. On your AWS Management Console, go to EC2 > Instances > and click on the instance that you've just configured. Click `Connect` > `SSH Client`.
2. Open Command Prompt on Windows/Linux or Terminal on Mac.
3. Run the command given as `Example` under the connection instructions:

4. You may see the following message

```
The authenticity of host 'ec2-18-188-136-133.us-east-2.compute.amazonaws.com (18.188.136.133)' can't be established. ECDSA key fingerprint is SHA256:9Je3cWvvII8iQDuranju1h1lqJ+5+8cgThuDgzCzmns. Are you sure you want to continue connecting (yes/no)?
```

...run `yes`.

You are now connected from your internet client to your EC2 instance!

References

- AWS Guide to Connecting to EC2 Instance via SSH

Security Groups

A **security group** acts as a virtual firewall for your EC2 instances to control incoming and outgoing traffic based on their IP address.

- Security group rules enable you to filter traffic based on protocols and port numbers.
- Security groups are stateful — if you send a request from your instance, the response traffic for that request is allowed to flow in regardless of inbound security group rules. Responses to allowed inbound traffic are allowed to flow out, regardless of outbound rules.
- You can 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.

When demoing configuring an EC2 instance, you should set the security groups before you launch your instance. You can set your first Security Group to `Anywhere`, so that it is easiest to connect from any IP address.

Changing an instance's security groups

After you launch an instance into a VPC, you can change the security groups that are associated with the instance. You can change the security groups for an instance when the instance is in the running or stopped state.

To change the security groups for an instance using the console

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, choose `Instances`.
3. Open the context (right-click) menu for the instance and choose `Networking` > `Change Security Groups`.
4. In the `Change Security Groups` dialog box, select one or more security groups from the list and choose `Assign Security Groups`.

Amazon Machine Image (AMI)

An Amazon Machine Image (AMI) provides the information required to launch an instance. Think of it as a template for an EC2 Instance.

An AMI includes the following:

- One or more EBS snapshots.
- Launch permissions that control which AWS accounts can use the AMI to launch instances.
- A block device mapping that specifies the volumes to attach to the instance when it's launched.

References

- AMI Types

Amazon Elastic Block Store (Amazon EBS)

Amazon Elastic Block Store (Amazon EBS) provides block level storage volumes for use with EC2 instances.

- EBS volumes behave like raw, unformatted block devices. You can mount these volumes as devices on your instances.
- EBS volumes that are attached to an instance are exposed as storage volumes that persist independently from the life of the instance.
- You can create a file system on top of these volumes, or use them in any way you would use a block device (such as a hard drive).
- You can dynamically change the configuration of a volume attached to an instance.
- With Amazon EBS, you pay only for what you use.

EBS Volumes

An Amazon EBS volume is a durable, block-level storage device that you can attach to your instances. After you attach a volume to an instance, you can use it as you would use a physical hard drive. You can use EBS volumes as primary storage for data that requires frequent updates, such as the system drive for an instance or storage for a database application.

- EBS volumes are created in a specific **Availability Zone**, and can then be attached to any instances in that same Availability Zone.

EBS Snapshots

You can back up the data on your Amazon EBS volumes to **Amazon S3** by taking point-in-time snapshots. Snapshots are incremental backups, which means that only the blocks on the device that have changed after your most recent snapshot are saved.

How to Create an EBS Volume for AWS EC2

1. Sign into AWS using your administrator account.
2. Navigate to the EC2 Console.
3. Choose an EC2 setup region from the Region drop-down list at the top of the page.
4. Select `Volumes` in the Navigation pane.
5. Click `Create Volume`.
6. Click `Create`.
7. Choose `Actions→ Create Snapshot`.
8. Type `EBS.Backup` in the Name field, type `Test Backup` in the Description field, and then click `Create`.
9. Click `Close`. The volume is ready to use.

When you finish this example, you can delete the volume you created by selecting its entry in the list and choosing Actions →Delete Volume. In a real-world setup, you can attach this volume to any EC2 instance or detach it when it’s no longer needed.

References

- AWS EBS Documentation