**1. Introduction to AWS EC2**

AWS EC2 (Elastic Compute Cloud) provides scalable computing capacity in the cloud. It allows you to launch virtual servers, known as EC2 instances, which you can configure with various hardware and software configurations.

**Key EC2 Concepts**

* **Instance**: A virtual server in the cloud.
* **AMI (Amazon Machine Image)**: Preconfigured templates for EC2 instances that include the OS and optional software.
* **Instance Type**: Defines the hardware configuration of the instance (CPU, memory, storage, etc.).
* **Key Pair**: Used for SSH access to your EC2 instance (public and private key pair).
* **Security Group**: Acts as a firewall, controlling inbound and outbound traffic to EC2 instances.
* **Elastic IP**: A static, public IP address that can be associated with an instance.
* **EBS (Elastic Block Store)**: Persistent storage that can be attached to EC2 instances.

**2. Steps to Launch an EC2 Instance**

**Step 1: Log in to the AWS Management Console**

Go to [AWS Management Console](https://aws.amazon.com/console/) and log in to your account.

**Step 2: Navigate to EC2 Dashboard**

1. In the AWS Console, search for **EC2** in the search bar and select it.
2. This takes you to the EC2 dashboard, where you can manage instances, security groups, key pairs, and more.

**Step 3: Launch an EC2 Instance**

1. **Click on "Launch Instance"**:
   * In the EC2 Dashboard, click the "Launch Instance" button to begin the setup process.
2. **Choose an Amazon Machine Image (AMI)**:
   * Select an AMI (such as Amazon Linux 2, Ubuntu, or a Windows Server).
   * AMIs can be free-tier eligible, custom-made, or found in the AWS Marketplace.
3. **Choose an Instance Type**:
   * Select the appropriate instance type based on CPU, memory, and other requirements.
   * The **t2.micro** instance is eligible for the AWS free tier.
4. **Configure Instance Details**:
   * Number of instances: Enter how many instances you want to launch.
   * Network: Choose the default VPC or create a custom one.
   * Subnet: Choose a subnet in which to launch the instance.
   * Auto-assign public IP: Select "Enable" if you want the instance to be publicly accessible.
   * IAM Role: Attach an IAM role if your instance needs access to other AWS services.
   * Other advanced settings (such as shutdown behavior, monitoring, and tenancy) can be configured if needed.
5. **Add Storage**:
   * The root volume is configured by default with EBS. You can add more volumes or increase the storage size as needed.
   * EBS volumes can be General Purpose (SSD) or Provisioned IOPS (SSD) based on performance needs.
6. **Add Tags**:
   * Tags are key-value pairs used to categorize and manage your instances. Example: Key: Name, Value: MyInstance.
7. **Configure Security Group**:
   * Create a new security group or select an existing one.
   * Configure **inbound rules** to allow SSH (port 22) for Linux or RDP (port 3389) for Windows.
   * Specify the IP range (CIDR) that can access the instance. For public access, set 0.0.0.0/0, but this is not recommended for production.
8. **Review and Launch**:
   * Review the instance configuration.
   * When ready, click "Launch".
9. **Choose an Existing Key Pair or Create a New One**:
   * Select or create a key pair for SSH access to the instance. Download the .pem file for the key pair if you're creating a new one.
   * Click "Launch Instances".

**3. Access Your EC2 Instance**

**For Linux Instances (using SSH)**

1. Open your terminal.
2. Run the following SSH command:

bash

Copy code

ssh -i /path/to/your-key.pem ec2-user@public-ip-address

* + Replace /path/to/your-key.pem with the path to your .pem file.
  + Replace public-ip-address with the public IP or DNS name of your instance.

1. You will now have SSH access to your EC2 instance.

**For Windows Instances (using RDP)**

1. Open **Remote Desktop Connection** on your local machine.
2. Enter the public IP address of your Windows instance.
3. Download the password from the EC2 console (requires the key pair to decrypt).
4. Use the administrator username (usually "Administrator") and the decrypted password to log in.

**4. EC2 Management Tasks**

**Stopping and Starting Instances**

* **Stop**: Stops the instance but retains the root EBS volume. You can restart the instance later.
* **Start**: Restarts a stopped instance.
* **Reboot**: Reboots the instance without data loss.
* **Terminate**: Permanently deletes the instance and associated storage unless you configured the storage to persist.

**Monitoring Instances**

* **CloudWatch**: Monitor instance metrics such as CPU utilization, disk I/O, and network traffic. Basic monitoring is enabled by default.

**Elastic IPs**

* You can assign an **Elastic IP** to ensure your instance has a static public IP that persists across stops and starts.

**Auto Scaling**

* You can use **Auto Scaling Groups** to automatically launch or terminate instances based on predefined metrics (such as CPU load).

**5. EC2 Storage Options**

**EBS (Elastic Block Store)**

* Provides persistent block-level storage volumes for EC2 instances. You can attach/detach volumes from running instances.
* **EBS Types**:
  + **gp3/gp2 (General Purpose SSD)**: Suitable for most workloads.
  + **io2/io1 (Provisioned IOPS SSD)**: High-performance storage for I/O-intensive applications.
  + **st1 (Throughput Optimized HDD)**: For big data or large, sequential workloads.
  + **sc1 (Cold HDD)**: Lowest-cost storage for infrequently accessed data.

**EFS (Elastic File System)**

* Provides a scalable file storage system that can be mounted by multiple EC2 instances simultaneously. Ideal for shared data between instances.

**Instance Store**

* Provides ephemeral storage that is physically attached to the host hardware. Data is lost when the instance is stopped or terminated.

**6. EC2 Pricing and Cost Management**

**Pricing Models**

* **On-Demand**: Pay for instances by the hour or second without long-term commitments.
* **Reserved Instances**: Commit to a 1- or 3-year term for significant cost savings.
* **Spot Instances**: Bid for unused EC2 capacity, which can be interrupted when AWS needs the capacity back, but comes at a much lower cost.
* **Savings Plans**: Flexible pricing model that provides savings in exchange for a consistent amount of compute usage (measured in dollars per hour).

**Cost Management**

* Use **AWS Cost Explorer** and **Budgets** to monitor and forecast your EC2 usage and costs.
* Right-size your instances by selecting appropriate instance types and utilizing auto-scaling to optimize cost.

**7. Terminating an EC2 Instance**

When you no longer need an instance:

1. Go to the **EC2 Dashboard**.
2. Select the instance(s) you want to terminate.
3. Click on "Actions" > "Instance State" > "Terminate".

Note: Once terminated, the instance data is lost unless you've stored it on EBS or other external storage solutions.

**8. Best Practices for EC2**

**Security Best Practices**

* Always use **Key Pairs** and avoid password-based SSH access.
* Restrict access via **Security Groups** by using the least privilege principle (open ports only to necessary IP ranges).
* Use **IAM roles** to manage permissions securely.
* Enable **CloudTrail** to log API activity and track access to instances.

**Backup and Recovery**

* Regularly back up important data using **EBS snapshots**.
* Consider using **AMI** for creating a template of a configured instance for backup or launch of identical instances.

**Performance Optimization**

* Choose appropriate instance types based on workload (e.g., T3 for general-purpose, C5 for compute-intensive tasks).
* Use **Auto Scaling** to automatically adjust the number of instances based on demand.

**9. EC2 Alternatives**

* **AWS Lambda**: For running code without provisioning servers (serverless).
* **Amazon Lightsail**: Simplified EC2 for small projects or applications.
* **ECS/EKS**: For container-based workloads.

**Conclusion**

AWS EC2 is a powerful and flexible compute service that allows you to run applications in the cloud with ease. By following this tutorial, you should now be familiar with launching, managing, and optimizing EC2 instances for various workloads.