Standard Operating Procedure (SOP)

Title

AWS EC2 – Virtual Machine (Instance) Creation

Version

1.0

Purpose

This SOP defines standardized steps to create an **Amazon EC2 Virtual Machine** using both the **AWS Management Console** and **AWS CLI**, including security, networking, and operational best practices.

Scope

Applies to all teams provisioning EC2 instances in AWS accounts for **development, testing, staging, and production** workloads.

Responsibilities

- **Cloud / DevOps Engineer** – EC2 provisioning and configuration
- **Security Team** – Security group and IAM review
- **Operations Team** – Monitoring, backup, and lifecycle management

Prerequisites

- Active AWS account
- IAM permissions:
- ec2:RunInstances
- ec2:Describe*
- ec2:CreateTags
- AWS CLI installed and configured (`aws configure`)
- VPC and subnet available
- Key pair (for Linux/Windows access)

---

1. EC2 Creation Using AWS Management Console

Step-by-Step Procedure

1. Log in to **AWS Management Console**

2. Navigate to **EC2 → Instances**
3. Click **Launch Instance**
4. Enter **Instance Name**
5. Select **Amazon Machine Image (AMI)**
6. Amazon Linux / Ubuntu / RHEL / Windows Server
7. Choose **Instance Type**
8. Example: `t3.micro` (Free Tier eligible)
9. Select or create **Key Pair**
10. `.pem` (Linux/macOS)
11. `.ppk` (Windows – PuTTY)
12. Configure **Network Settings**
13. VPC
14. Subnet
15. Auto-assign Public IP (if required)
16. Configure **Security Group**
17. Linux: Allow SSH (22)
18. Windows: Allow RDP (3389)
19. Restrict source IPs
20. Configure **Storage (EBS)**
    - Root volume size and type (gp3 recommended)
21. Review configuration
22. Click **Launch Instance**

Output

- EC2 instance state: **Running**
- Public/Private IP assigned

---

2. EC2 Creation Using AWS CLI

Command

```
aws ec2 run-instances
  --image-id ami-xxxxxxxx
  --instance-type t3.micro
  --key-name my-keypair
  --security-group-ids sg-xxxxxxxx
  --subnet-id subnet-xxxxxxxx
  --tag-specifications 'ResourceType=instance,Tags=[{Key=Name,Value=MyEC2}]'
```

Verify Instance

```
aws ec2 describe-instances --filters "Name=tag:Name,Values=MyEC2"
```

3. Accessing the EC2 Instance

Linux Instance

```
ssh -i my-keypair.pem ec2-user@<public-ip>
```

Windows Instance

- Download **RDP file** from EC2 console
- Decrypt administrator password using key pair
- Connect via **Remote Desktop (mstsc)**

---

4. Security Best Practices

- Use **least privilege IAM roles**
- Do not allow `0.0.0.0/0` on SSH/RDP in production
- Use **Security Groups as firewalls**
- Enable **IMDSv2 only**
- Encrypt EBS volumes (default)

---

5. Networking Best Practices

- Use **private subnets** for backend servers
- Use **NAT Gateway** for outbound access
- Use **Elastic Load Balancer** for public traffic
- Enable **VPC Flow Logs**

---

6. Cost Optimization Best Practices

- Right-size instance types
- Stop or terminate unused EC2 instances
- Use **Savings Plans / Reserved Instances**
- Use **Auto Scaling Groups** where applicable

---

7. Operational Best Practices

- Apply standard **resource tags**
- Name

- Environment
- Owner
- CostCenter
- Enable **CloudWatch monitoring & alarms**
- Take **EBS snapshots** regularly
- Patch OS using SSM Patch Manager

---

8. Validation Checklist

- EC2 instance is running
- SSH/RDP connectivity verified
- Security group rules validated
- Monitoring enabled
- Tags applied

---

9. Approval

Role	Name	Date
Cloud Architect		
Security Lead		