Amazon Machine Images (AMI) - Complete Guide

What is an AMI?

An Amazon Machine Image (AMI) is a template that contains the software configuration (operating system, application server, and applications) required to launch an EC2 instance. AMIs serve as the basic unit of deployment for services delivered using EC2.

Types of AMIs

1. **Public AMIs** - Available to all AWS users
2. **Private AMIs** - Only available to your AWS account
3. **Shared AMIs** - Shared with specific AWS accounts
4. **AWS Marketplace AMIs** - Pre-configured software from third-party vendors

Creating an AMI

Method 1: From an Existing EC2 Instance

Using AWS Console:

1. Navigate to EC2 Dashboard
2. Select **Instances** from the left menu
3. Select the instance you want to create an AMI from
4. Click **Actions → Image and templates → Create image**
5. Configure the following:
   - **Image name**: Provide a descriptive name
   - **Image description**: Optional but recommended
   - **No reboot**: Check if you want to create the AMI without rebooting (not recommended for consistency)
   - **Instance volumes**: Configure volumes to include
   - **Tags**: Add tags for organization
6. Click **Create image**

Using AWS CLI:

```
bash
```

```
aws ec2 create-image \
    --instance-id i-1234567890abcdef0 \
    --name "MyServerAMI-2024-01-06" \
    --description "Production web server AMI" \
    --no-reboot
```

Method 2: From a Snapshot

Using AWS Console:

1. Go to **EC2 → Snapshots**
2. Select the snapshot
3. Click **Actions → Create image from snapshot**
4. Configure the AMI settings
5. Click **Create**

Using AWS CLI:

```
bash

aws ec2 register-image \
    --name "MyAMI-from-snapshot" \
    --root-device-name /dev/xvda \
    --block-device-mappings \
    DeviceName=/dev/xvda,Ebs={SnapshotId=snap-1234567890abcdef0}
```

Managing AMIs

Viewing Your AMIs

AWS Console:

- Navigate to **EC2 → AMIs** (under Images)
- Filter by ownership: Owned by me, Private images, Public images

AWS CLI:

```
bash
```

```
# List your AMIs
aws ec2 describe-images --owners self

# List specific AMI
aws ec2 describe-images --image-ids ami-1234567890abcdef0
```

Copying AMIs Across Regions

AWS Console:

1. Select the AMI
2. Click **Actions → Copy AMI**
3. Select destination region
4. Optionally encrypt the AMI
5. Click **Copy AMI**

AWS CLI:

```
bash

aws ec2 copy-image \
  --source-region us-east-1 \
  --source-image-id ami-1234567890abcdef0 \
  --region us-west-2 \
  --name "MyAMI-Copy"
```

Sharing AMIs

AWS Console:

1. Select the AMI
2. Click **Actions → Edit AMI permissions**
3. Choose sharing option:
   - **Private**: Only your account
   - **Specific AWS accounts**: Enter account IDs
   - **Public**: Available to all (use with caution)
4. Click **Save changes**

AWS CLI:

```
bash
```

```
# Share with specific account
aws ec2 modify-image-attribute \
    --image-id ami-1234567890abcdef0 \
    --launch-permission "Add=[{UserId=123456789012}]"

# Make public
aws ec2 modify-image-attribute \
    --image-id ami-1234567890abcdef0 \
    --launch-permission "Add=[{Group=all}]"
```

Deregistering (Deleting) AMIs

Important: Deregistering an AMI doesn't delete the associated snapshots. You must delete them separately.

AWS Console:

1. Select the AMI
2. Click **Actions → Deregister AMI**
3. Confirm the action
4. Go to **Snapshots** and delete associated snapshots

AWS CLI:

```
bash

# Deregister AMI
aws ec2 deregister-image --image-id ami-1234567890abcdef0

# Delete associated snapshot
aws ec2 delete-snapshot --snapshot-id snap-1234567890abcdef0
```

Best Practices

Naming Conventions

- Use descriptive names with version numbers or dates
- Example: `webapp-prod-v2.1.3-20240106`
- Include environment, application, and timestamp

Security

- **Encrypt AMIs** containing sensitive data
- **Regular updates**: Create new AMIs with latest security patches

- **Limit sharing**: Only share AMIs with trusted accounts
- **Review permissions**: Regularly audit who has access

Organization

- **Use tags** for categorization:
  - Environment (Production, Staging, Development)
  - Application name
  - Owner/Team
  - Cost center
- **Document**: Keep records of what's installed on each AMI
- **Version control**: Maintain AMI versions for rollback capability

Cost Management

- **Delete unused AMIs**: Regular cleanup of old AMIs
- **Monitor snapshot storage**: Snapshots incur storage costs
- **Consolidate**: Avoid duplicate AMIs across regions unless necessary
- **Lifecycle policies**: Implement automated cleanup

Backup Strategy

- Create AMIs on a regular schedule for critical instances
- Keep AMIs in multiple regions for disaster recovery
- Test AMI restoration periodically
- Document recovery procedures

Common Use Cases

Golden Images

Create standardized AMIs with:

- Hardened OS configurations
- Required security agents
- Monitoring tools
- Company-standard software

Auto Scaling

- Use AMIs with Auto Scaling groups
- Pre-bake applications into AMIs for faster scaling
- Keep AMIs updated for new instances

Disaster Recovery

- Maintain AMIs in multiple regions
- Regular backups of production instances
- Quick instance recovery from AMIs

Development/Testing

- Create baseline AMIs for development environments
- Share AMIs across development teams
- Test environment consistency

Monitoring and Maintenance

Regular Tasks

- **Monthly**: Review and clean up unused AMIs
- **Quarterly**: Update base AMIs with security patches
- **As needed**: Create new AMIs after significant configuration changes

Automation with AWS Lambda

```
python
```

```python
# Example: Automated AMI cleanup (Python)
import boto3
from datetime import datetime, timedelta

ec2 = boto3.client('ec2')

def cleanup_old_amis():
    cutoff_date = datetime.now() - timedelta(days=90)

    images = ec2.describe_images(Owners=['self'])['Images']

    for image in images:
        creation_date = datetime.strptime(
            image['CreationDate'],
            '%Y-%m-%dT%H:%M:%S.%fZ'
        )

        if creation_date < cutoff_date:
            print(f"Deregistering old AMI: {image['ImageId']}")
            ec2.deregister_image(ImageId=image['ImageId'])
```

Troubleshooting

AMI Creation Takes Too Long

- Check instance size and number of volumes
- Ensure adequate IOPS for snapshot creation
- Consider using instance store instead of EBS for temporary data

AMI Won't Launch

- Verify AMI is in the same region as launch attempt
- Check AMI permissions and sharing settings
- Ensure instance type compatibility with AMI architecture

Snapshot Costs Growing

- Identify and delete orphaned snapshots
- Implement retention policies
- Use lifecycle management tools

Additional Resources

- AWS Documentation: https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/AMIs.html
- AWS CLI Reference: https://docs.aws.amazon.com/cli/latest/reference/ec2/
- AWS SDK Documentation: https://aws.amazon.com/tools/