AWS Cloud Security Project - IAM Implementation

Project Overview

This project demonstrates the implementation of AWS Identity and Access Management (IAM) security best practices. The goal is to set up a secure AWS environment with proper user access controls, policies, and EC2 instance management.

Table of Contents

Prerequisites

- An active AWS account
- Basic understanding of cloud computing concepts
- Familiarity with AWS console navigation

Architecture

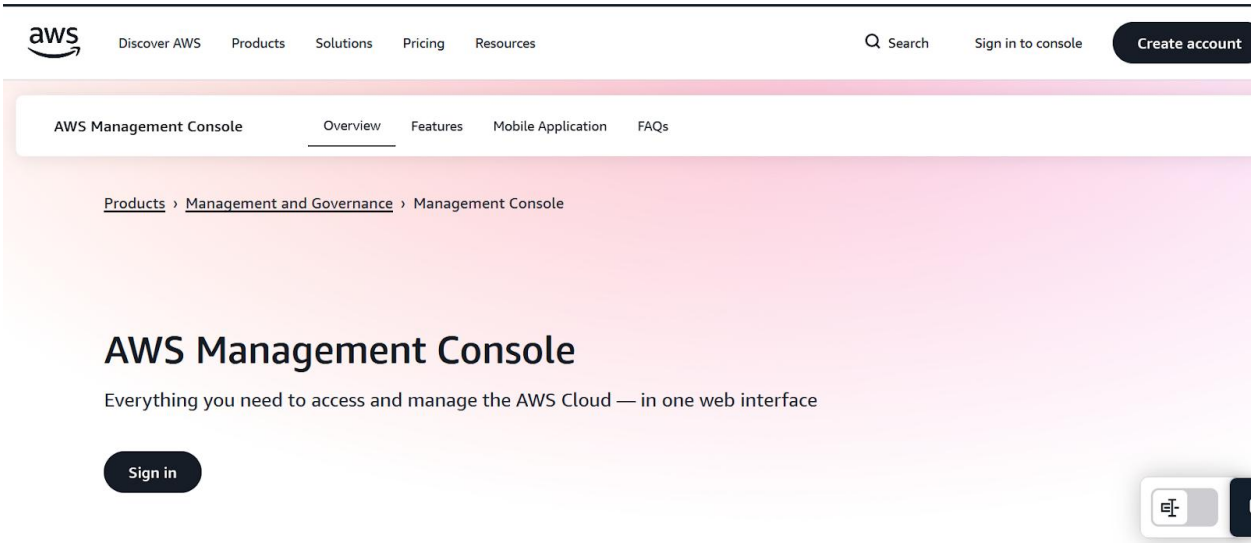
This project implements a basic IAM security structure with:

- AWS Management Console access
- EC2 instances for testing
- Custom IAM policies
- AWS CLI aliases for quick access
- IAM groups and users with specific permissions

Step-by-Step Implementation

Step 1: Setup an AWS Management Console

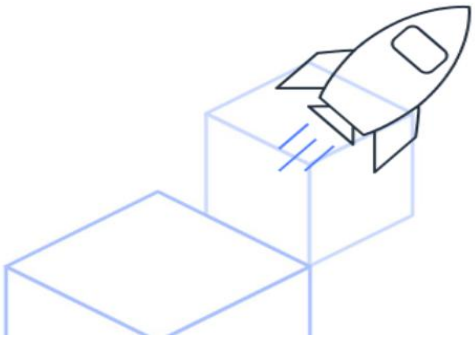
1. Navigate to AWS Console
2. Sign in with your root account credentials
3. Enable Multi-Factor Authentication (MFA) on your root account for security



aws

Try AWS at no cost for up to 6 months

Start with USD $100 in AWS credits, plus earn up to USD $100 by completing various activities.



Sign up for AWS

Root user email address

Used for account recovery and as described in the **AWS Privacy Notice**

AWS account name

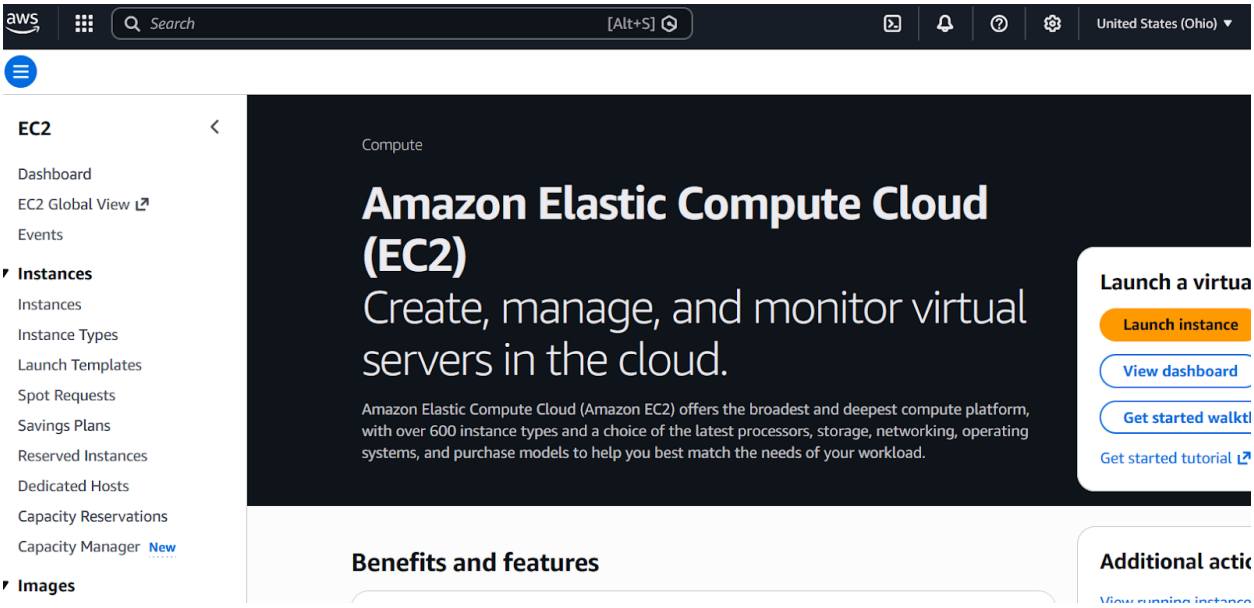
Choose a name for your account. You can change this name in your account settings after you sign up.

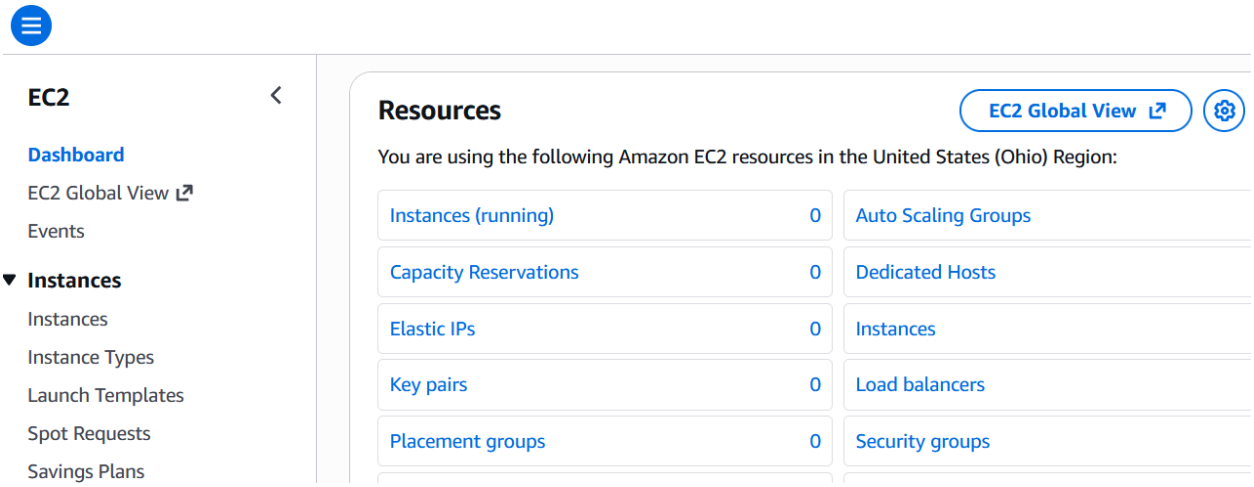
Verify email address

OR

Sign in to an existing AWS account

This site uses essential cookies. See our Cookie Notice for more information.





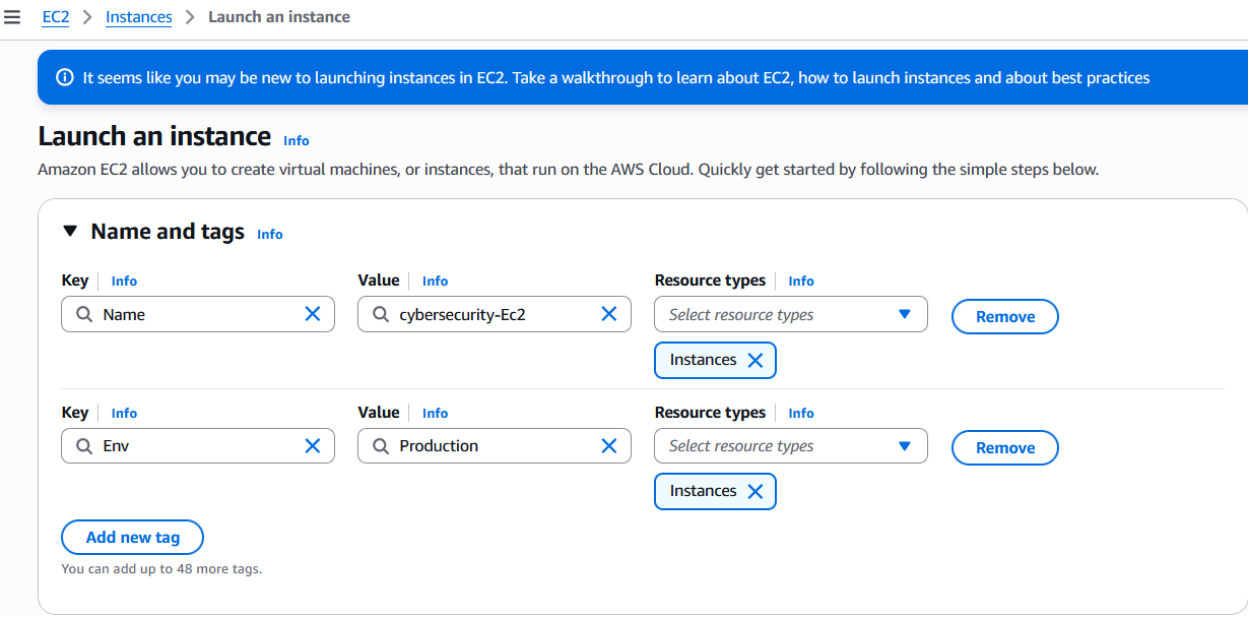
- Capture the MFA configuration screen

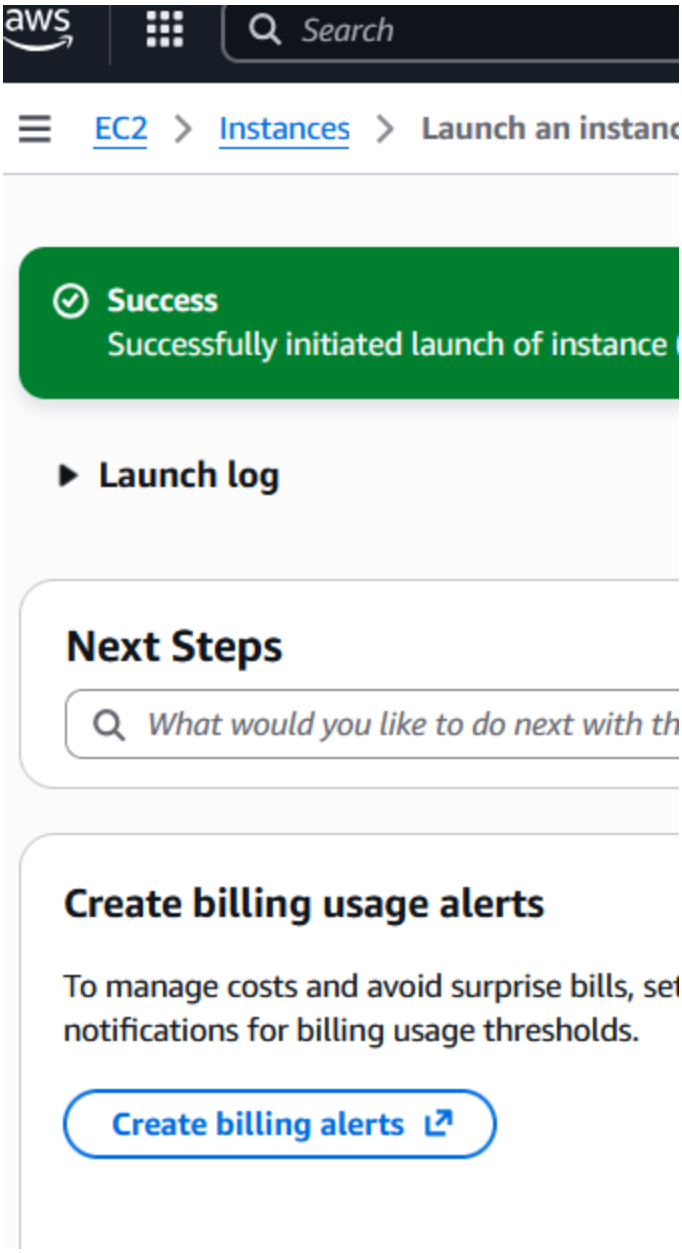
Key Points:

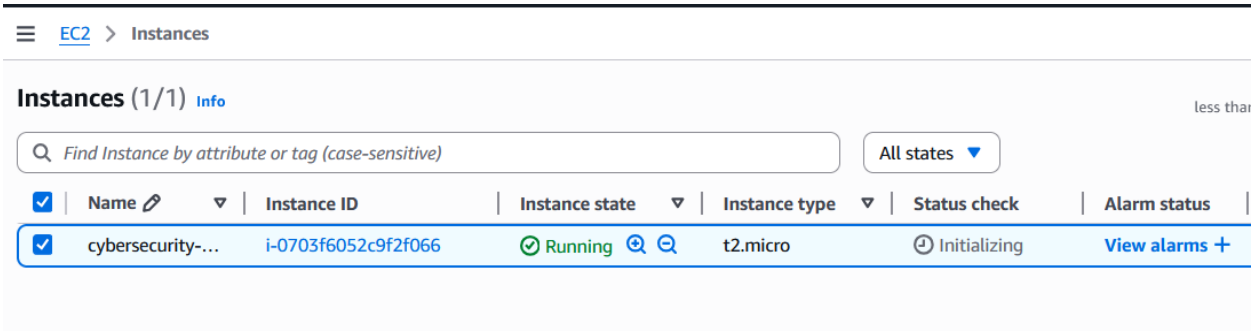
- Never use root account for daily operations
- Always enable MFA on root account
- Note your AWS Account ID for future reference

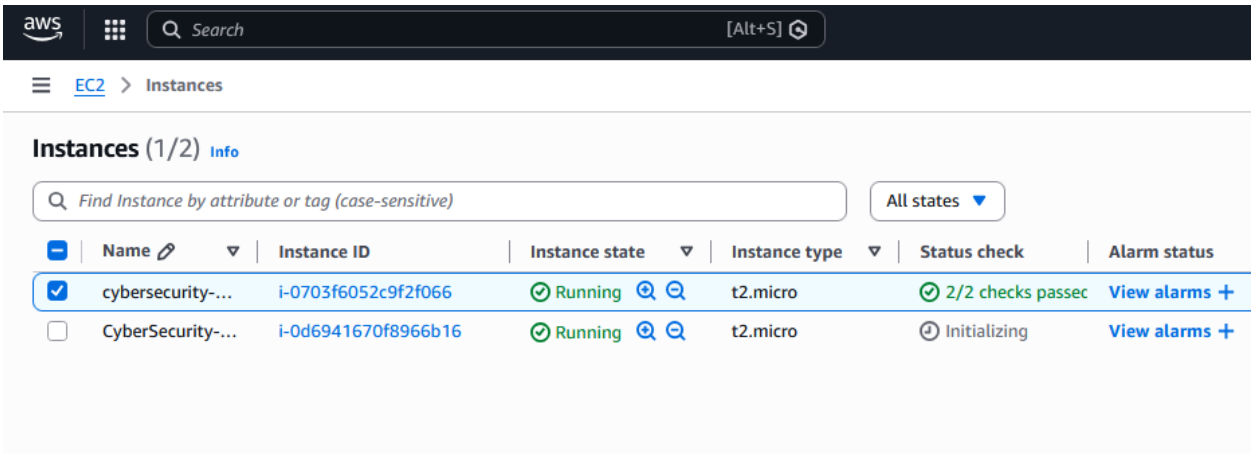
Step 2: Launch 2 EC2 Instances

1. In the AWS Console, navigate to **EC2 Dashboard**
2. Click **Launch Instance**
3. Configure First Instance:
    - **Name:** IAM-Test-Instance-1
    - **AMI:** Amazon Linux 2023 (or latest)
    - **Instance Type:** t2.micro (Free tier eligible)
    - **Key Pair:** Create new key pair or select existing
    - **Network Settings:** Default VPC
    - **Security Group:** Allow SSH (Port 22) from your IP
    - Click **Launch Instance**
4. Repeat for Second Instance:
    - **Name:** IAM-Test-Instance-2
    - Use same configuration as Instance 1




aws
Search
EC2 > Instances > Launch an instanc
Success
Successfully initiated launch of instance
Launch log
Next Steps
What would you like to do next with th
Create billing usage alerts
To manage costs and avoid surprise bills, set
notifications for billing usage thresholds.
Create billing alerts


EC2 > Instances
Instances (1/1) Info
less thar
Find Instance by attribute or tag (case-sensitive)
All states
Name
Instance ID
Instance state
Instance type
Status check
Alarm status
cybersecurity-...
i-0703f6052c9f2f066
Running
t2.micro
Initializing
View alarms +



- Capture the EC2 dashboard showing both instances in "running" state

Important Notes:

- Save your key pair (.pem file) securely
- Note the Instance IDs for both instances
- Verify both instances are in "running" state

Step 3: Create an IAM Policy

1. Navigate to **IAM Dashboard**
2. Click **Policies** in the left sidebar
3. Click **Create Policy**
4. Select **JSON** tab
5. Enter the following policy (example - EC2 read-only access):

json

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ec2:Describe*",
        "ec2:Get*"
      ],
      "Resource": "*"
    }
  ]
}
```

6. Click **Next: Tags** (optional)
7. Click **Next: Review**
8. **Policy Name:** EC2-ReadOnly-Policy
9. **Description:** "Allows read-only access to EC2 resources"
10. Click **Create Policy**

IAM > Policies > Create policy

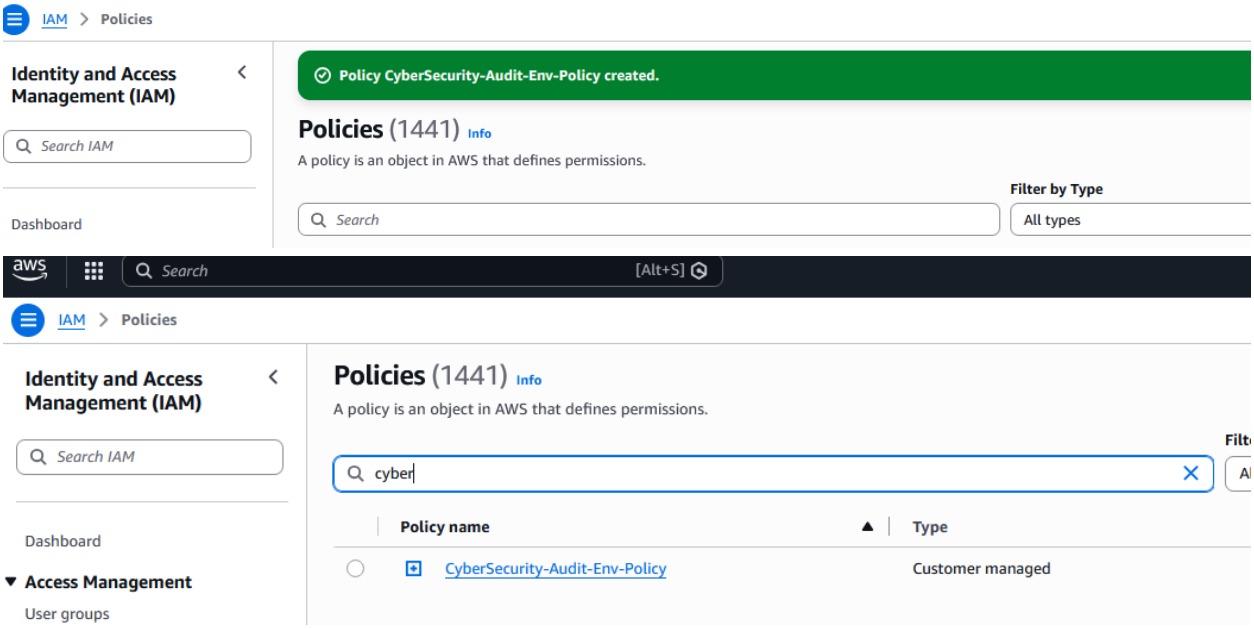
Policy editor

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "ec2:*",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "ec2:ResourceTag/Env": "Audit"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "ec2:Describe*",
            "Resource": "*"
        },
        {
            "Effect": "Deny",
            "Action": [
                "ec2:DeleteTags",
                "ec2:CreateTags"
            ],
            "Resource": "*"
        }
    ]
}
```

+ Add new statement

JSON Ln 29, Col 0

Security: 0 Errors: 0 Warnings: 0 Suggestions: 0



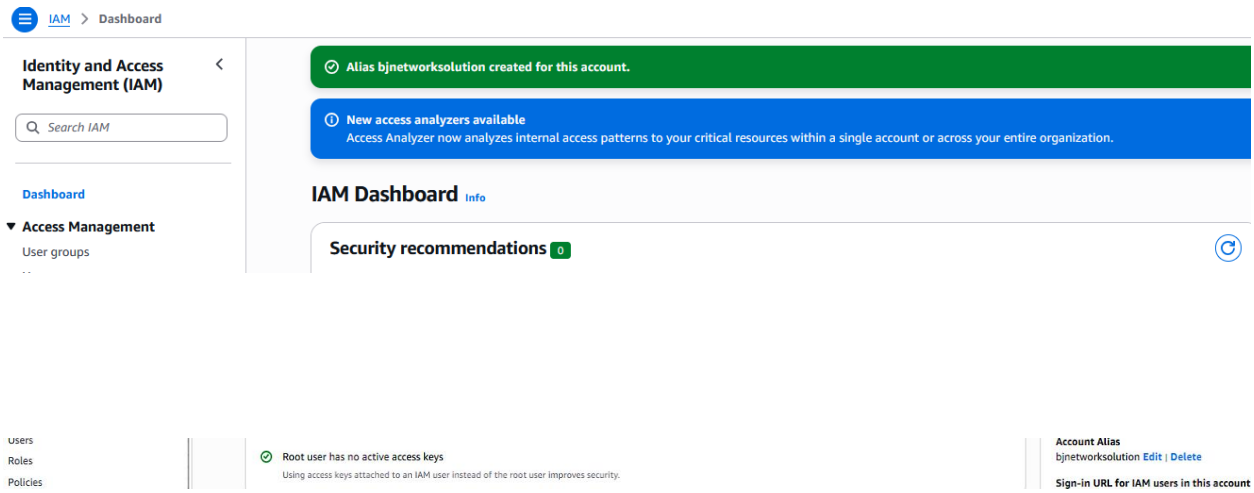
- Capture the success message after policy creation

Policy Explanation:

- ec2:Describe* - Allows listing all EC2 resources
- ec2:Get* - Allows retrieving EC2 resource details
- Resource: "*" - Applies to all EC2 resources

Step 4: Create an AWS Alias

1. In the IAM Dashboard, look for **AWS Account** section (top right)
2. Click on your **Account ID**
3. Click **Create** next to Account Alias
4. Enter a unique alias (e.g., mycompany-aws-prod)
5. Click **Create Alias**



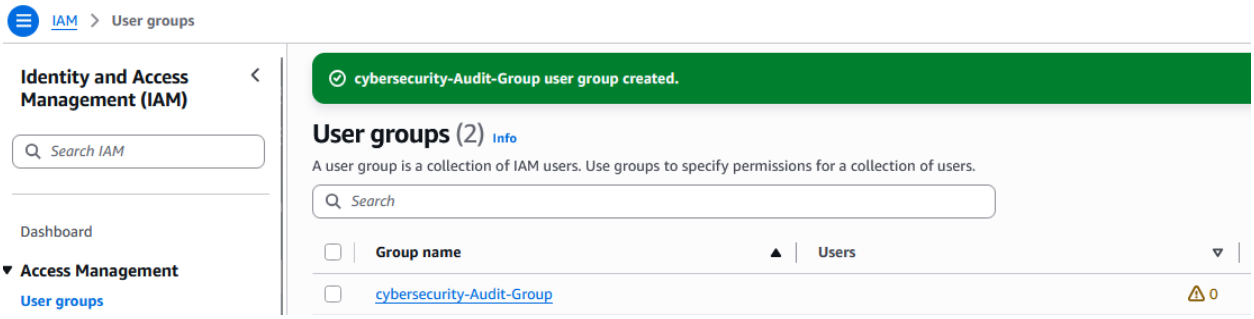
Benefits of Account Alias:

- Easier sign-in URL:https://bjnet.signin.aws.amazon.com/console
- More user-friendly than using account ID
- Professional appearance for team members

Step 5: Create IAM Group and User

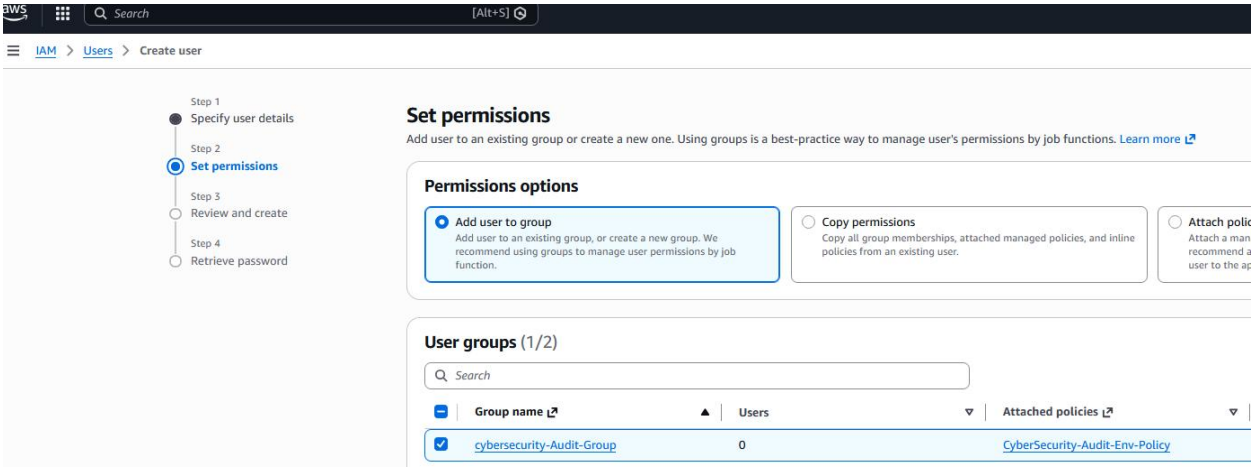
Create IAM Group

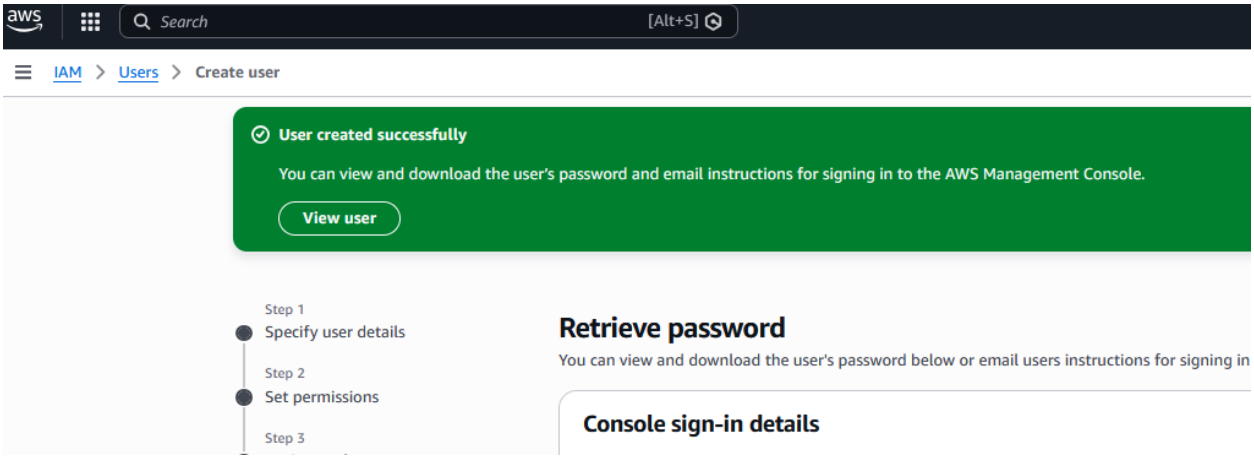
1. In IAM Dashboard, click **User groups**
2. Click **Create Group**
3. **Group Name:** EC2-Administrators
4. Attach the policy created in Step 3: EC2-ReadOnly-Policy
5. Click **Create Group**



Create IAM User

1. Click **Users** in IAM Dashboard
2. Click **Add Users**
3. **User Name:** test-user-1
4. **Access Type:** Select both:
    - ✅ AWS Management Console access
    - ✅ Programmatic access (for API/CLI)
5. **Console Password:** Choose "Autogenerated" or "Custom"
6. **Require Password Reset:** ✅ (Recommended)
7. Click **Next: Permissions**
8. Select **Add user to group**
9. Select EC2-Administrators group
10. Click **Next: Tags** (optional)
11. Click **Next: Review**
12. Click **Create User**
13. **IMPORTANT:** Download the credentials CSV file





Security Notes:

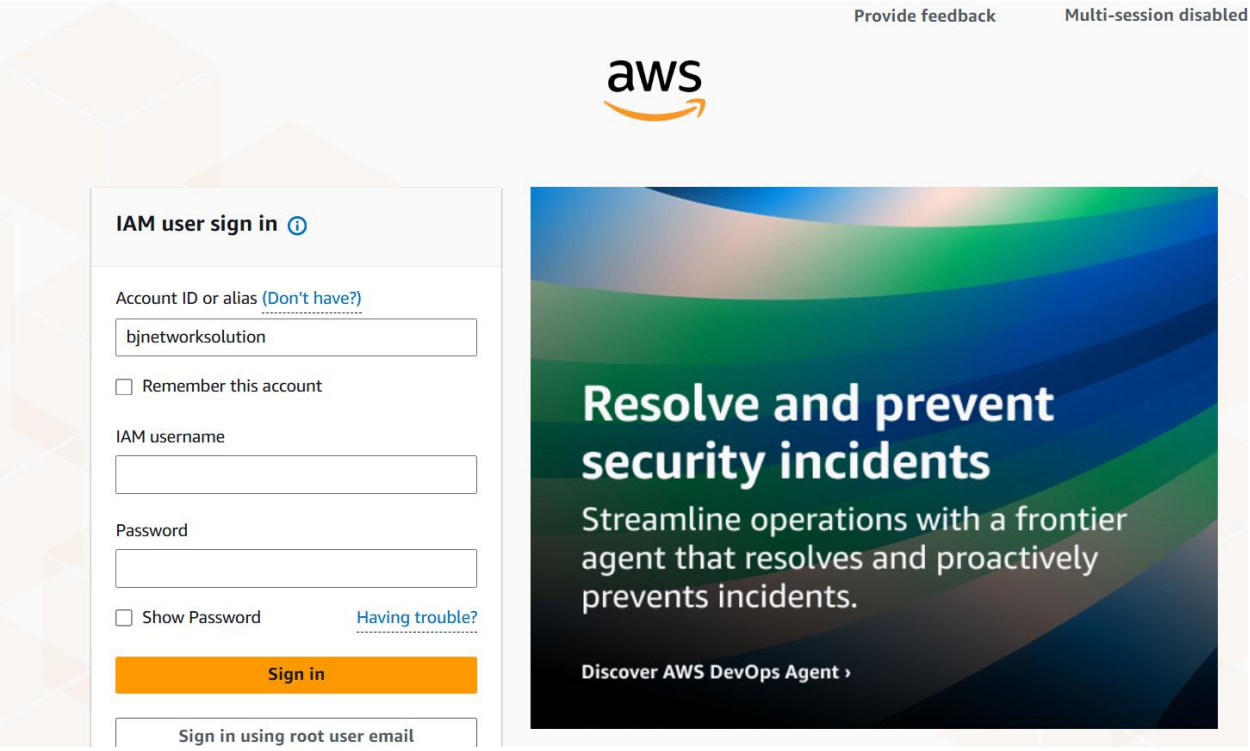
- Store credentials securely
- Never commit credentials to Git
- Enable MFA for all IAM users in production

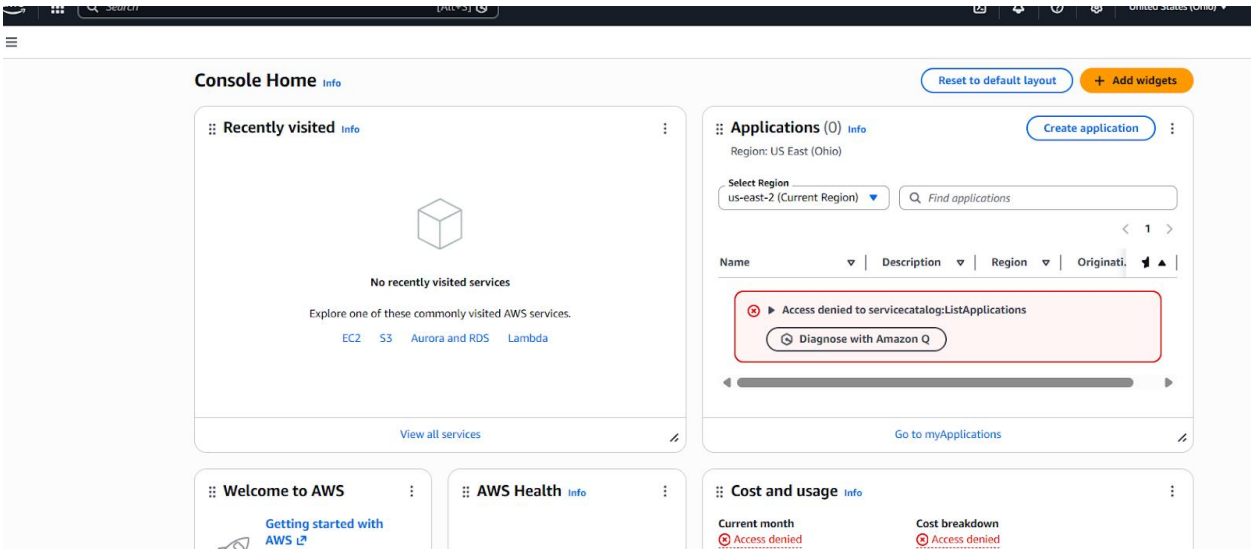
Step 6: Test The IAM User Access

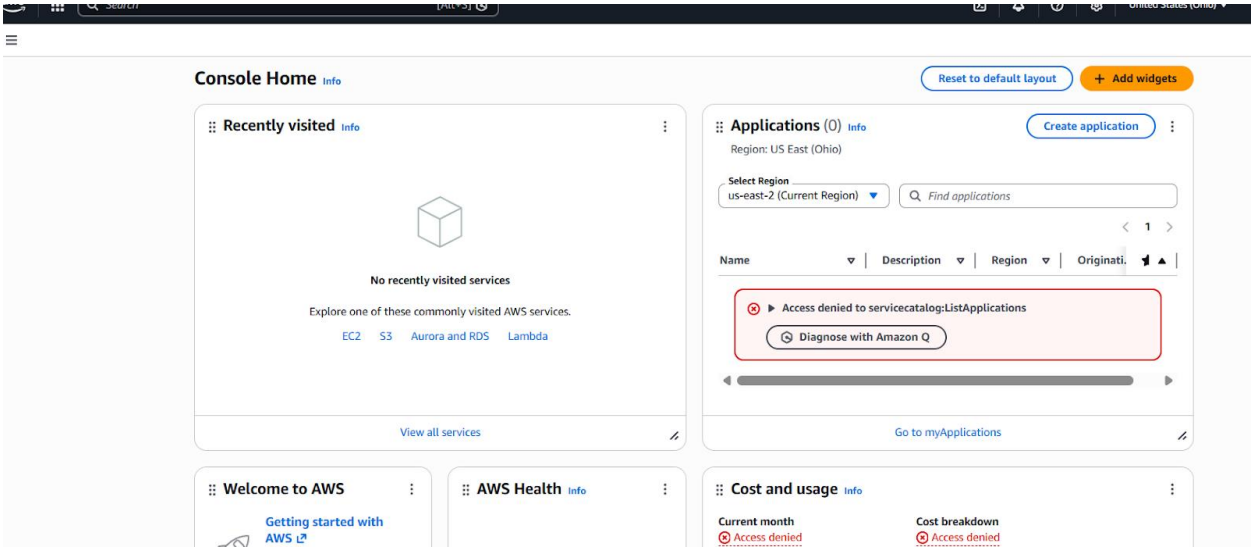
Test Console Access

1. Sign out of the root account
2. Navigate to the sign-in URL with your alias
3. Log in with the IAM user credentials:
    - **Account ID/Alias:** Your account alias
    - **Username:** test-user-1
    - **Password:** The password from Step 5
4. If prompted, change the password

5. Navigate to EC2 Dashboard
6. Verify you can **view** EC2 instances (read-only)
7. Try to **stop** an instance - this should fail with "Access Denied"







Security Best Practices

✅ Implemented in This Project

- Root account MFA enabled
- IAM users instead of root for daily operations
- Least privilege principle (read-only policy)
- IAM groups for permission management
- Password policy enforcement

🔒 Additional Recommendations

1. **Enable CloudTrail:** Log all API calls for auditing
2. **Set up billing alerts:** Monitor unexpected charges
3. **Rotate access keys regularly:** Every 90 days minimum
4. **Use IAM roles for EC2:** Instead of embedding credentials
5. **Implement password policy:** Require complex passwords
6. **Regular access reviews:** Remove unused users/permissions
7. **Enable AWS Organizations:** For multi-account management

Troubleshooting

Issue: Cannot launch EC2 instances

Solution:

- Check if you've reached the instance limit (default is 20 per region)
- Verify you're in a region that supports t2.micro instances
- Ensure your account has EC2 service permissions

Issue: IAM user cannot sign in

Solution:

- Verify the sign-in URL includes your account alias or ID
- Check username and password are correct
- Ensure the user has console access enabled

Issue: Policy not working as expected

Solution:

- Review the JSON syntax for errors
- Check the policy simulator: IAM → Policies → Simulate
- Verify the policy is attached to the correct group/user

Issue: Access denied errors

Solution:

- Verify the user is in the correct IAM group
- Check that the policy is attached to the group
- Wait a few minutes for IAM changes to propagate

Conclusion

This project demonstrates fundamental AWS IAM security concepts:

- ✅ Proper account setup and security
- ✅ EC2 instance management
- ✅ Custom IAM policy creation
- ✅ Group-based permission management
- ✅ User access testing and validation

Author

Your Name -Bolaji Bakare

Acknowledgments

- AWS Documentation Team
- Cloud Security Community