

WORKING WITH IDENTITY & ACCESS MANAGEMENT (IAM)

Identity & Access Management (IAM) Overview

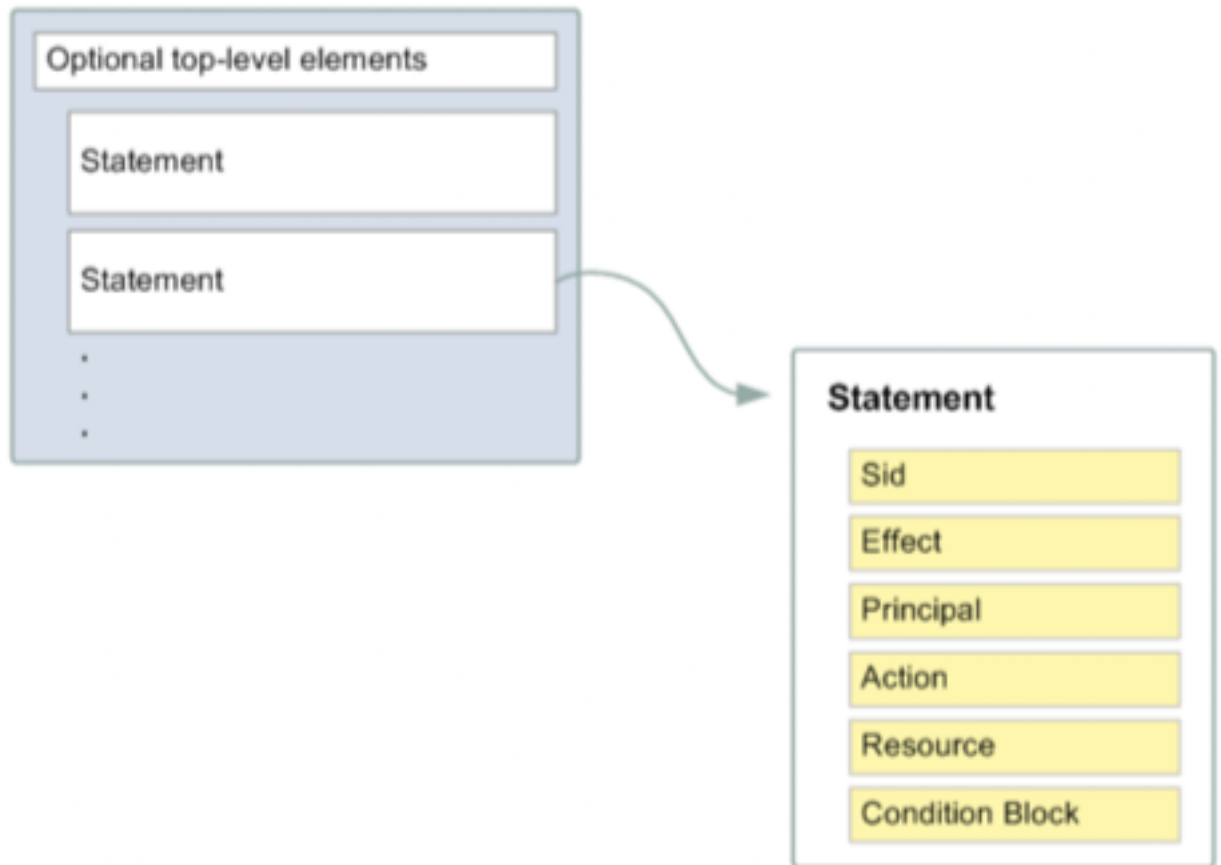
AWS Identity and Access Management (IAM) is a free service that enables you to manage access to AWS services and resources securely. Using IAM, you can create and manage AWS users and groups, and use permissions to allow and deny their access to AWS resources.

Part 1: Create AWS IAM Identity Policies

You manage access in AWS by creating policies and attaching them to IAM entities (users, groups of users, or roles) or AWS resources. A policy is an object in AWS that, when associated with an identity or resource, defines their permissions. AWS evaluates these policies when a Principal entity (user or role) makes a request. Identity-based policies are [JSON](#) (Java Script Object Notation) permissions policy documents that you can attach to an identity (user, group of users, or role) to manage access. A JSON policy document includes these elements:

- Optional policy-wide information at the top of the document
- One or more individual statements Each statement in a policy includes information about a single permission. If a policy includes multiple statements,

AWS applies a logical OR across the statements when evaluating them.



The information in a statement is contained within a series of elements.

1. Version – Specify the version of the policy language that you want to use.
2. Statement – Container for the following elements:
 - **Sid** – Include an optional statement ID to differentiate between your statements.
 - **Effect** – Use Allow or Deny indicating whether the policy allows or denies access.
 - **Principal** – We will NOT use this element in this lab. The Principal element is used in resource policy statements to identify the Principal (account, user, role, or federated user) to which you would like to allow or deny access. This element is NOT used when creating IAM identity policies. In IAM identity policies the Principal is implied from the user or role that the policy is attached to.
 - **Action** – Include a list of actions that the policy allows or denies.
 - **Resource** – Specify a list of resources to which the actions apply.
 - **Condition (Optional)** – Specify the circumstances under which the policy grants permission.

If you want to define more than one permission for an entity (user, group, or role), you can use multiple statements in a single policy. You can also attach multiple policies to an IAM entity to manage access. For example, below is an example has two permission statements included in the policy that enable all DynamoDB actions against two tables located us-east-1 and us-west-2 regions in the AWS Account 123456789012.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "dynamodb:*",
      "Resource": [ "arn:aws:dynamodb:us-east-1:123456789012:table/MyTestApp_DDB_Table" ]
    },
    {
      "Effect": "Allow",
      "Action": "dynamodb:*",
      "Resource": [ "arn:aws:dynamodb:us-west-2:123456789012:table/MyTestApp_DDB_Table" ]
    }
  ]
}
```

We will now create an AWS IAM identity policies using the AWS console.

1. Log into the AWS account you plan to use for this lab. Ensure that you authenticate using an identity which has been granted Administrative access in the AWS account. It is recommended that you use an identity that has the AdministratorAccess policy (arn:aws:iam::aws:policy/AdministratorAccess) attached.
2. Access the AWS [IAM console](#) and select Policies from the sidebar. Click **Create Policy**. On the "Create Policy" screen, select the JSON tab and paste the policy contents from below into the JSON text editing panel in the AWS console.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowDepartmentEC2Management",
      "Effect": "Allow",
      "Action": "ec2:*",
```

```

        "Resource": "*",
        "Condition": {
            "StringEquals": {
                "ec2:ResourceTag/department":
"${aws:PrincipalTag/department}"
            }
        }
    },
    {
        "Sid": "AllowEC2DescribeAll",
        "Effect": "Allow",
        "Action": "ec2:Describe*",
        "Resource": "*"
    },
    {
        "Sid": "DenyTagManagement",
        "Effect": "Deny",
        "Action": [
            "iam:UntagUser",
            "iam:UntagRole",
            "ec2:DeleteTags",
            "ec2:CreateTags",
            "iam:TagRole",
            "iam:TagUser"
        ],
        "Resource": "*"
    }
]
}

```

The policy has 3 statements. When attached to an IAM entity these statements:

- Allow the Principal (e.g. User) to execute all Elastic Compute Cloud (EC2) actions against EC2 instances which have the same department tag as the Principal attempting to perform the action.
- Allow the Principal (e.g. User) to describe all EC2 instances within the AWS account.
- Explicitly deny the Principal (e.g. User) access to manipulate IAM or EC2 tags within the AWS account.

`aws:PrincipalTag` is an AWS Global Condition Context Key. AWS provides context keys which can be used in IAM policies to restrict access. In this IAM policy, we use it to check that the tag attached to the IAM Principal (e.g. User) making the request. This policy uses the `StringEquals` operator to compare the value of the department key attached to the IAM Principal with the department key of the EC2 resource they are attempting to access using the Amazon EC2 `ec2:ResourceTag` condition key.

3. After pasting the policy from above into the JSON editor in the AWS Console. Click **Review policy**.

Create policy

1 2

A policy defines the AWS permissions that you can assign to a user, group, or role. You can create and edit a policy in the visual editor and using JSON. [Learn more](#)

Visual editor

JSON

[Import managed policy](#)

```
1 {
2   "Version": "2012-10-17",
3   "Statement": [
4     {
5       "Sid": "AllowDepartmentEC2Management",
6       "Effect": "Allow",
7       "Action": "ec2:*",
8       "Resource": "*",
9       "Condition": {
10        "StringEquals": {
11          "ec2:ResourceTag/department": "${aws:PrincipalTag/department}"
12        }
13      }
14    },
15    {
16      "Sid": "AllowEC2DescribeAll",
17      "Effect": "Allow",
18      "Action": "ec2:Describe*",
19      "Resource": "*"
20    },
21    {
22      "Sid": "DenyTagManagement"
```

Character count: 470 of 6,144.

Cancel

Review policy

4. Name the policy `departmental-ec2-access` and optionally, add a description. Click **Create policy**.

Create policy

1

2

Review policy

Name*

Use alphanumeric and '+-=,@_.' characters. Maximum 128 characters.

Description

Maximum 1000 characters. Use alphanumeric and '+-=,@_.' characters.

Summary

Q Filter

Service	Access level	Resource	Request condition
Explicit deny (2 of 235 services)			
EC2	Full: Tagging	All resources	None
IAM	Full: Tagging	All resources	None
Allow (1 of 235 services) Show remaining 234			
EC2	Full: List, Read, Write, Permissions management	All resources	ec2:ResourceTag/department = \${aws:PrincipalTag/department}

* Required

[Cancel](#)

[Previous](#)

[Create policy](#)

5. Policy creation is confirmed as you are returned the IAM console policy dashboard.
6. Repeat steps 1 through 5 to create a second policy. Name the policy `contractorsroleassumptionpolicy`. On the create policy screen, use the policy provided below. This policy allows the assumption of an IAM Role if the condition is met that the role has a tag with a key of `contractorsassumerole` and a value of `true`.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "sts:AssumeRole",
    "Resource": "*",
    "Condition": { "StringLike": { "iam:ResourceTag/contractorsassumerole":
"true"}}
  }
}
```

Congratulations! You have created two IAM Identity policies. These policies are customer-managed policies. Customer managed policies are standalone identity-based policies that you create and which you can attach to multiple IAM users, groups, or roles in your AWS account.

Part 2: Create IAM Users

1. Access the AWS [IAM console](#) and select Users from the sidebar. Click **Add User**.
2. On the **Add user** screen, under “Set User Details” enter “jdoe” in the User name field
3. On the “Add user” screen, under “Select AWS access type”, check the checkbox to enable AWS Management Console access. Do not enable programmatic access. Under “Console password” select the Custom password radio button and enter a password in the text box – make a note of this password since it will be needed to later to test the security configuration. Ensure the “User must create a new password at next sign-in” checkbox is unchecked. Click **Next: Permissions**.

Add user



Set user details

You can add multiple users at once with the same access type and permissions. [Learn more](#)

User name*

[+ Add another user](#)

Select AWS access type

Select how these users will access AWS. Access keys and autogenerated passwords are provided in the last step. [Learn more](#)

Access type* ☐ **Programmatic access**
Enables an **access key ID** and **secret access key** for the AWS API, CLI, SDK, and other development tools.

☒ **AWS Management Console access**
Enables a **password** that allows users to sign-in to the AWS Management Console.

Console password* ☐ Autogenerated password
☒ Custom password

☐ Show password

Require password reset ☐ User must create a new password at next sign-in
Users automatically get the [IAMUserChangePassword](#) policy to allow them to change their own password.

* Required

[Cancel](#)

[Next: Permissions](#)


4. On the **Set permissions** screen, select the “Attach existing policies directly” option and use the filter search to locate the Identity policy created in part one


of this lab. Select the checkbox next to the policy to select it. Click **Next: Tags**.


Add user

1 2 3 4 5

Set permissions

 Add user to group

 Copy permissions from existing user

 Attach existing policies directly

Create policy

Filter policies Showing 1 result

	Policy name	Type	Used as
<input checked="" type="checkbox"/>	departmental-ec2-access	Customer managed	None

5. On the “Add Tags” screen, add a tag with the Key **department** and a Value of **hr**.

Add user

1 2 3 4 5

Add tags (optional)

IAM tags are key-value pairs you can add to your user. Tags can include user information, such as an email address, or can be descriptive, such as a job title. You can use the tags to organize, track, or control access for this user. [Learn more](#)

Key	Value (optional)	Remove
<input type="text" value="department"/>	<input type="text" value="hr"/>	
<input type="text" value="Add new key"/>	<input type="text"/>	

You can add 49 more tags.

6. On the “Review” screen, review your choices. Click **Create User**
7. On the final “Add user” screen, verify that user was successfully created. Click **Close**.
8. Repeat Lab Part Two steps 1 through 7 to create the user “sally”. On step 2 enter “sally” in the User name field. On step 3, provide sally with the same permissions as jdoe. On step 4, when adding a tag to the user sally, the tag

should have the key “department” and a value of “finance” as seen below.

Add user



Add tags (optional)

IAM tags are key-value pairs you can add to your user. Tags can include user information, such as an email address, or can be descriptive, such as a job title. You can use the tags to organize, track, or control access for this user. [Learn more](#)

Key	Value (optional)	Remove
<input type="text" value="department"/>	<input type="text" value="finance"/>	<input type="button" value="✕"/>
<input type="text" value="Add new key"/>	<input type="text"/>	

You can add 49 more tags.

9. Repeat Lab Part Two steps 1 through 7 to create the user “Anne”. On step 2 enter “anne” in the User name field. On step 3, DO NOT provide any permissions. On step 4, do not add any tags to the user anne.
10. Repeat Lab Part Two steps 1 through 7 to create the user “Bob”. On step 2 enter “Bob” in the User name field. On step 3, DO NOT provide any permissions. On step 4, do not add any tags to the user Bob.

Congratulations! You have created four IAM users. The users are named jdoe, Sally, Anne and Bob.

- jdoe has the permission policy “department-ec2-access” attached to her identity. jdoe has a tag with a key of “department” and a value of “hr” attached to her identity.
- Sally has the permission policy “department-ec2-access” attached to her identity. Sally has a tag with a key of “department” and a value of “finance” attached to her identity.
- Anne has no permission policy attached to his identity. Anne has no tags attached to his identity.
- Bob has no permission policy attached to his identity. Bob has no tags attached to his identity.

Part 3: Create an IAM Group

An IAM group is a collection of users. Groups are often based on job function and can be used to simplify provisioning common user access requirements. They allow you to manage permissions by applying policies to groups of users, rather than applying policies to each individual user.

1. Access the AWS [IAM console](#) and select Groups from the sidebar. Click Create New Group.
2. Type in **Contractors** as the Group Name. Go to the Next Step.

3. On the Attach Policy screen, enter the search string “S3” in the search bar and check the checkbox next to the AWS managed policy titled `AmazonS3ReadOnlyAccess`.
4. On the same Attach Policy screen execute a second search. Enter the search string “contractors” for a and add the customer-managed policy named `contractorsroleassumptionpolicy`. Policies are what give IAM entities permissions. AWS provides managed policies for many common access needs. We will use the `AmazonS3ReadOnlyAccess` which will provide read only access to the Amazon Simple Storage Service (S3) to all members of the Contractors group. Click Next Step.

Create New Group Wizard

Step 1 : Group Name






Step 2 : Attach Policy

Step 3 : Review

Attach Policy

Select one or more policies to attach. Each group can have up to 10 policies attached.

Filter: Policy Type


		Policy Name 
<input type="checkbox"/>		AmazonS3FullAccess
<input checked="" type="checkbox"/>		AmazonS3ReadOnlyAccess
<input type="checkbox"/>		AWSQuickSightS3Policy
<input type="checkbox"/>		QuickSightAccessForS3StorageManagementAnalyticsReadOnly
<input type="checkbox"/>		AmazonDMSRedshiftS3Role

5. On the review screen, click **Create Group**.
6. The group is created, and you are returned to the Group creation IAM console. Select the Contractors group and chose “Add Users to Group” from the Group Actions Menu Dropdown.
7. Add the users named **Anne** and **Bob** to the Contractors group. Check the checkbox next to their names and Click “Add Users”. The IAM groups

dashboard will now show the Contractors group has 2 users.



[IAM](#) > [Groups](#) > **Contractors**

▼ Summary

Group ARN:	arn:aws:iam::[REDACTED]:group/Contractors 
Users (in this group):	2
Path:	/
Creation Time:	2020-07-30 23:28 UTC+1000

Users Permissions Access Advisor

This view shows all users in this group: **2 Users**

User	Actions
 anne	Remove User from Group
 bob	Remove User from Group

Congratulations! You have created an IAM Group and attached the AWS managed policy `AmazonS3ReadOnlyAccess` which provides read only access to Amazon Simple Storage Service (S3) to members of this group. You added the customer managed policy named `contractorsroleassumptionpolicy` which will allow members of this group to assume roles which have been tagged with a Key of `contractorsassumerole` and a Value of `true`. You added the IAM users Anne and Bob into the Contractors group.

Part 4: Create IAM Roles

IAM Roles can be assumed by AWS services, IAM users, or applications. They are assigned temporary rather than permanent credentials whenever assumed. Using roles for privileged permissions sets can help improve your security posture since credential exposure is minimized.


1. Access the AWS [IAM console](#) and select Roles from the sidebar.
2. On the **Select type of trusted identity** page, you decide who or what will be able to assume this role. For this lab, we will create a role that allows an EC2


instance to read files in S3. Therefore, we will stay on the **AWS service** tab and select EC2. Go to **Next: Permissions**.


Create role


1 2 3 4

Select type of trusted entity

**AWS service**
EC2, Lambda and others

**Another AWS account**
Belonging to you or 3rd party

**Web identity**
Cognito or any OpenID provider

**SAML 2.0 federation**
Your corporate directory

Allows AWS services to perform actions on your behalf. [Learn more](#)

Choose a use case

Common use cases

EC2
Allows EC2 instances to call AWS services on your behalf.

Lambda
Allows Lambda functions to call AWS services on your behalf.

Or select a service to view its use cases

API Gateway	CodeGuru	ElastiCache	Kinesis	RoboMaker
AWS Backup	CodeStar Notifications	Elastic Beanstalk	Lake Formation	S3
AWS Chatbot	Comprehend	Elastic Container Service	Lambda	SMS
AWS Support	Config	Elastic Transcoder	Lex	SNS
Amplify	Connect	ElasticLoadBalancing	License Manager	SWF
AppStream 2.0	DMS	Forecast	Machine Learning	SageMaker
AppSync	Data Lifecycle Manager	Gamelift	Media Services	Security Hub

* Required

Cancel

Next: Permissions

3. Attach a managed policy with S3 Read Only access to the role by typing s3 into the search bar, and then selecting the **AmazonS3ReadOnlyAccess** policy. Go to the **Next: Review**.

Create role

1 2 3 4

▼ Attach permissions policies

Choose one or more policies to attach to your new role.

Create policy

Filter policies Showing 5 results

	Policy name	Used as
<input type="checkbox"/>	AmazonDMSRedshiftS3Role	None
<input type="checkbox"/>	AmazonS3FullAccess	Permissions policy (5)
<input checked="" type="checkbox"/>	AmazonS3ReadOnlyAccess	Permissions policy (3)
<input type="checkbox"/>	AWSQuickSightS3Policy	Permissions policy (1)
<input type="checkbox"/>	QuickSightAccessForS3StorageManagementAnalyticsReadOnly	Permissions policy (1)

- Give your role a descriptive name, such as **EC2S3ReadOnly** and edit the role description to be a helpful summary of what this role is. When you're done, **Create Role**.
- You are now back on the **Roles** page. Enter the name of the role you just created into the search bar and click on the role name.

Identity and Access Management (IAM)

- Dashboard
- Access management
 - Groups
 - Users
 - Roles**
 - Policies

✓ The role **EC2S3ReadOnly** has been created.

Create role Delete role

Q EC2S3

Role name ▾	Trusted entities	Last activity ▾
<input type="checkbox"/> EC2S3ReadOnly	AWS service: ec2	None

- You are now on the Summary page of the role you just created. Here you can view and edit attributes of the role, such as how long the role's temporary credentials last. The default value as you can see below is 1 hour but can be up to 12 hours. Click on the Trust relationships tab and you will see that `ec2.amazonaws.com` is listed as a trusted entity that can assume this role.

[Roles](#) > [EC2S3ReadOnly](#)

Summary

Role ARN	arn:aws:iam::[redacted]:role/EC2S3ReadOnly
Role description	Allows EC2 instances to call AWS services on your behalf. Edit
Instance Profile ARNs	arn:aws:iam::[redacted]:instance-profile/EC2S3ReadOnly
Path	/
Creation time	2020-07-30 16:43 UTC+1000
Last activity	Not accessed in the tracking period
Maximum session duration	1 hour Edit

Permissions Trust relationships Tags Access Advisor Revoke sessions

▾ Permissions policies (1 policy applied)

Attach policies

Policy name ▾	Policy type ▾
▶ AmazonS3ReadOnlyAccess	AWS managed policy

Congratulations! You've just created an IAM role which will allow EC2 instances in your account to assume this role and read objects in S3. We will now create a role which can be assumed by an IAM user in your account.

7. Access the AWS [IAM console](#) and select Roles from the sidebar and then Create role.
8. On the **Select type of trusted identity** page, select “Another AWS Account”. Cross account role assumption is a method to provide privileged access to identities across multiple AWS accounts. For the purposes of this lab we will demonstrate role assumption within one AWS account. We will create a role that we will allow IAM users to switch to this role from within the same account. In the Account ID text box, enter the Account ID of the AWS account that you are currently logged into. For the purposes of this lab, leave both options (require external ID, require MFA) unchecked. Go to **Next: Permissions**.

You can find the AWS account ID, by navigating to the IAM Console and clicking dashboard. Make a note of the account ID from IAM users sign-in link.

9. Under the Attach permissions policies screen, use the text box to perform a filtered search for the `AmazonEC2FullAccess` policy. Check the check box to select this policy and attach it to the role being created. Click **Next: Tags**.


Create role 1 2 3 4

▼ Attach permissions policies

Choose one or more policies to attach to your new role.

Create policy ↺

Filter policies ▼ Showing 1 result

	Policy name ▼	Used as	Description
<input checked="" type="checkbox"/>	 AmazonEC2FullAccess	None	Provides full access to Amazon EC2 via t...

10. On the **Add Tags** screen, add a tag with the Key “contractorsassumerole” and a Value of “true”. Click “Next Review”. On the review screen name the role

"ec2poweruser". Click **create role**.

Create role

Review

Provide the required information below and review this role before you create it.

Role name* ec2poweruser

Use alphanumeric and '+=, @-_' characters. Maximum 64 characters.

Role description ec2poweruser

Maximum 1000 characters. Use alphanumeric and '+=, @-_' characters.

Trusted entities The account 5672

Policies  AmazonEC2FullAccess [↗](#)

Permissions boundary Permissions boundary is not set

No tags were added.

11. The role is created successfully. Access the ec2poweruser role details page by searching for role by name from the Roles dashboard.
12. Confirm that **AmazonEC2FullAccess** is listed under attached policies and copy the switch roles console link from your IAM dashboard (example link location underlined in red in the image below). Copy into a text editor for safekeeping. You will need this URL in Part Five of this lab to test access.

[Roles](#) > [ec2poweruser](#)

Summary

Role ARN [arn:aws:iam::5672:role/ec2poweruser](#) [↗](#)

Role description [ec2poweruser](#) [Edit](#)

Instance Profile ARNs [↗](#)

Path /

Creation time 2019-07-02 11:03 MDT

Maximum CLI/API session duration 1 hour [Edit](#)

Give this link to users who can switch roles in the console <https://signin.aws.amazon.com/switchrole?roleName=ec2poweruser&account=iamlabexercise> [↗](#)

Permissions

Trust relationships

Tags (1)

Access Advisor

Revoke sessions

▼ Permissions policies (1 policy applied)

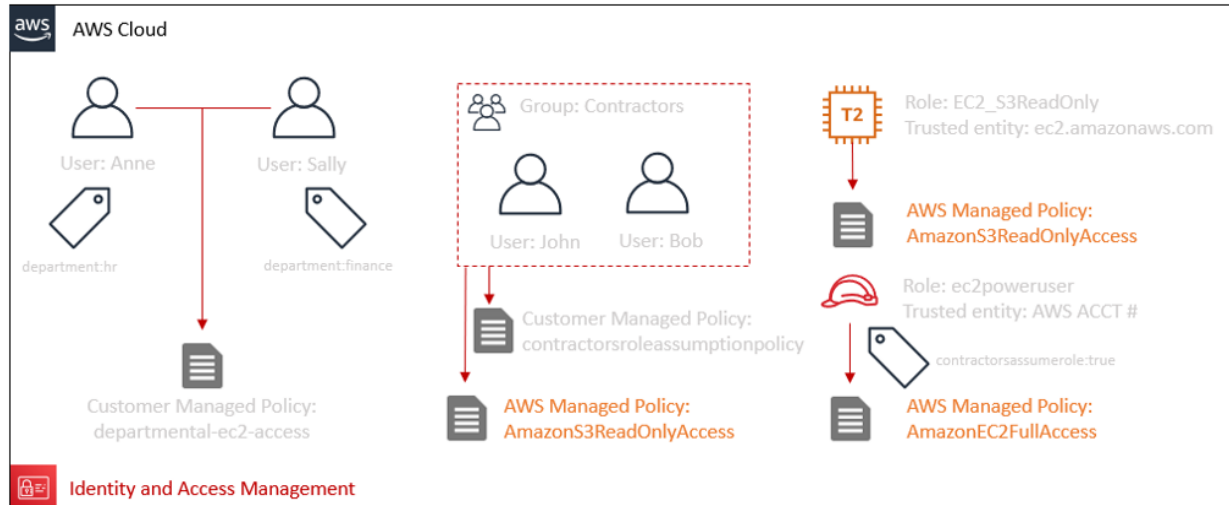
[Attach policies](#)

Policy name ▼

▶  AmazonEC2FullAccess

▶ Permissions boundary (not set)

Part 5: Test Access



We can summarize the existing configurations we have completed in this lab as follows:

- A customer managed IAM identity policy named departmental-ec2-access.
- A customer managed IAM identity policy named contractorsroleassumptionpolicy.
- A user named jdoo, who has a tag with the key of department and the value of hr. jdoo has the departmental-ec2-access policy attached to his identity.
- A user named Sally, who has a tag with the key of department and the value of finance. Sally has the departmental-ec2-access policy attached to her identity.
- A user named Anne.
- A user named Bob.
- An IAM Group named Contractors which has the AWS Managed policy named `AmazonS3ReadOnlyAccess` attached to the group. John and Bob are members of the group.
- A role named `EC2S3ReadOnly` (which has a trust policy making the role assumable by an EC2 instance) that has the Amazon managed policy named `AmazonS3ReadOnlyAccess` attached to the role.
- A role named `ec2poweruser` (which has a trust policy making the role assumable by IAM entities within this AWS account) that has the Amazon managed policy named `AmazonEC2FullAccess` attached to the role. The role has a tag with the Key of `contractorsassumerole` and the Value of `true`

We have used both AWS and Customer managed policies. We have attached policies to IAM users, IAM Roles and Groups of IAM users. We have tagged IAM Users jdoo and Sally. We have tagged the `ec2poweruser` IAM Role. It Is not possible to tag a Group of users. In the next part of this lab, we will test the access provided.

1. Ensure that you are still logged into your AWS account using an identity which has been granted Administrative access in the AWS account. It is recommended

that you use an identity that has the AdministratorAccess policy (arn:aws:iam::aws:policy/AdministratorAccess) attached. Access the [EC2 Console](#). Click the Launch instances button. The EC2 Launch wizard screen loads. On “Step 1: Choose an Amazon Machine Image (AMI)” page, select an Amazon Linux 2 AMI

1. Choose AMI 2. Choose Instance Type 3. Configure Instance 4. Add Storage 5. Add Tags 6. Configure Security Group 7. Review

Step 1: Choose an Amazon Machine Image (AMI) Cancel and Exit

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. You can select an AMI provided by AWS, our user community, or the AWS Marketplace; or you can select one of your own AMIs.

Search for an AMI by entering a search term e.g. "Windows" X

Search by Systems Manager parameter

Quick Start (3) 1 to 3 of 3 AMIs

My AMIs (0)
AWS Marketplace (179)
Community AMIs (7669)
☒ Free tier only ⓘ

Amazon Linux 2 AMI (HVM), SSD Volume Type - ami-08f3d892de259504d (64-bit x86) / ami-0ba960472fc891755 (64-bit Arm)

Amazon Linux 2
Free tier eligible
Amazon Linux 2 comes with five years support. It provides Linux kernel 4.14 tuned for optimal performance on Amazon EC2, systemd 219, GCC 7.3, Glibc 2.26, Binutils 2.29.1, and the latest software packages through extras.
Root device type: ebs Virtualization type: hvm ENA Enabled: Yes

Select

Amazon Linux 2 with .NET Core, PowerShell, Mono, and MATE Desktop Environment - ami-0652d6c7a2e2d090a

Amazon Linux 2 with .NET Core, PowerShell, Mono, and MATE Desktop Environment
Free tier eligible
.NET Core 3.1, Mono 6.8, PowerShell 6.2, and MATE DE pre-installed to run your .NET applications on Amazon Linux 2 with Long Term Support (LTS).
Root device type: ebs Virtualization type: hvm ENA Enabled: Yes

Select

Amazon Linux AMI 2018.03.0 (HVM), SSD Volume Type - ami-09d8b522f2b93bf0

Amazon Linux AMI 2018.03.0 (HVM), SSD Volume Type
Free tier eligible
The Amazon Linux AMI is an EBS-backed, AWS-supported image. The default image includes AWS command line tools, Python, Ruby, Perl, and Java. The repositories include Docker, PHP, MySQL, PostgreSQL, and other packages.
Root device type: ebs Virtualization type: hvm ENA Enabled: Yes

Select

- On “Step 2: Choose an Instance Type”, select a **t2.micro** EC2 instance type. Click “Next: Configure Instance Details”.

1. Choose AMI 2. Choose Instance Type 3. Configure Instance 4. Add Storage 5. Add Tags 6. Configure Security Group 7. Review

Step 2: Choose an Instance Type

Amazon EC2 provides a wide selection of instance types optimized to fit different use cases. Instances are virtual servers that can run applications. They have varying combinations of CPU, memory, storage, and networking capacity, and give you the flexibility to choose the appropriate mix of resources for your applications. [Learn more](#) about instance types and how they can meet your computing needs.

Filter by: All instance types Current generation Show/Hide Columns

Currently selected: t2.micro (Variable ECUs, 1 vCPUs, 2.5 GHz, Intel Xeon Family, 1 GiB memory, EBS only)

	Family	Type	vCPUs ⓘ	Memory (GiB)	Instance Storage (GiB) ⓘ	EBS-Optimized Available ⓘ	Network Performance ⓘ	IPv6 Support ⓘ
<input type="checkbox"/>	General purpose	t2.nano	1	0.5	EBS only	-	Low to Moderate	Yes
<input checked="" type="checkbox"/>	General purpose	t2.micro Free tier eligible	1	1	EBS only	-	Low to Moderate	Yes
<input type="checkbox"/>	General purpose	t2.small	1	2	EBS only	-	Low to Moderate	Yes
<input type="checkbox"/>	General purpose	t2.medium	2	4	EBS only	-	Low to Moderate	Yes

- On “Step 3: Configure Instance Details”, choose the default vpc and subnet within your account to launch your EC2 instance. Click **Next: Add Storage**.

Neither SSH inbound to, or outbound Internet access from the EC2 instance are required during this lab.

- On “Step 4: Add Storage”, accept the default storage size allocated to your EC2 instance.
- Click “Step 5: Add Tags”. Add two tags. The first tag should have the key **department** and a value of **hr**. The second tag should have the key **Name** and a value of **HR**. Click **Next Configure Security Group**

Values specified for tags values are case sensitive !!

1. Choose AMI 2. Choose Instance Type 3. Configure Instance 4. Add Storage 5. Add Tags 6. Configure Security Group 7. Review

Step 5: Add Tags

A tag consists of a case-sensitive key-value pair. For example, you could define a tag with key = Name and value = Webserver. A copy of a tag can be applied to volumes, instances or both. Tags will be applied to all instances and volumes. [Learn more](#) about tagging your Amazon EC2 resources.

Key (128 characters maximum)	Value (256 characters maximum)	Instances	Volumes	
department	hr	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="button" value="X"/>
Name	HR	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="button" value="X"/>

(Up to 50 tags maximum)

- On "Step 6: Configure Security Group", create a new security group with a name and description of **resource-tagging-lab**. Remove all inbound rules to the EC2 instance that are defined in the security group. Click **Review and Launch**

1. Choose AMI 2. Choose Instance Type 3. Configure Instance 4. Add Storage 5. Add Tags 6. Configure Security Group 7. Review

Step 6: Configure Security Group

A security group is a set of firewall rules that control the traffic for your instance. On this page, you can add rules to allow specific traffic to reach your instance. For example, if you want to set up a web server and allow Internet traffic to reach your instance, add rules that allow unrestricted access to the HTTP and HTTPS ports. You can create a new security group or select from an existing one below. [Learn more](#) about Amazon EC2 security groups.

Assign a security group: ☒ Create a new security group
☐ Select an existing security group

Security group name:

Description:

Type	Protocol	Port Range	Source	Description
This security group has no rules				

Warning
 You will not be able to connect to this instance as the AMI requires port(s) 22 to be open in order to have access. Your current security group doesn't have port(s) 22 open.

- On "Step 7: Review Instance Launch", review/validate your configurations. Click Launch. You will be prompted by the "Select an Existing Key Pair or Create a New Key Pair" prompt. Choose **Proceed without keypair** from the dropdown menu bar in the prompt and tick the checkbox to acknowledge that you will not be able to connect to the provisioned EC2 instance. Operating system access is not required for this lab.
- Repeat steps 1 through 8 to provision a second EC2 instance in your AWS account to provision an EC2 instance for the **FINANCE** department. When completing steps 1 through 8 for a second time, change the tag values provided in step 6. When creating the FINANCE EC2 instance, the first tag should have the key **department** and a value of **finance**. The second tag should have the key **Name** and a value of **FINANCE**. Please treat these values as case sensitive !!
- Test jdoe's access and ability to administer resources for the hr department
 - Return to the AWS [IAM console](#). Make note of the IAM users sign-in link immediately underneath the welcome banner (see example screenshot below). You will use this to sign in using the IAM User identities we have created in this lab (Anne, Sally and John) to validate their access. Copy the IAM users sign-in link into a text editor for safekeeping. It should be

of the format: <https://youraccountID-or-alias.signin.aws.amazon.com/console>

- Log out of the AWS console. Log in as jdoe using the IAM users sign-in link you collected from the IAM Console. Specify the IAM user jdoe as the User and use the password you specified earlier when creating the user jdoe.
11. Continue testing jdoe's access. Access the [EC2 console](#). Select the checkbox for the EC2 instance named FINANCE. Using the Actions menu, attempt to Terminate the EC2 instance named "FINANCE" by selecting STOP under the Instance State menu option. The operation will fail because the department tag value attached to the EC2 instance does not match the department tag value attached to the Principal (jdoe). The condition in the statement titled `AllowDepartmentEC2Management` in the departmental-ec2-access policy is not met and the request fails. Attempt the same operation after deselecting the EC2 instance named "FINANCE" and selecting the instance named "HR". The operation will succeed because the condition in the policy statement is met and the action is allowed. Log out of the AWS Console.
12. Test Sally's access and ability to administer resources for the finance department. a. Log in as Sally using the IAM users sign-in link you collected from the IAM Console. Specify the IAM user Sally as the User and use the password you specified earlier when creating the user Sally. b. Access the [EC2 console](#). Attempt to Stop the EC2 instance named "FINANCE". The operation will succeed. Do not terminate the EC2 instance named "FINANCE". Log out of the AWS Console.

Congratulations !! You have validated access for John and Sally to departmental EC2 resources based on tag key and tag values.

13. Test Anne's ability to access S3 as a result of her membership to the contractors group.
- Log in as Anne using the IAM users sign-in link you collected from the IAM Console. Specify the IAM user Anne as the User and use the

password you specified earlier when creating the user Anne.



Sign in as IAM user

Account ID (12 digits) or account alias

IAM user name

Password

Sign in

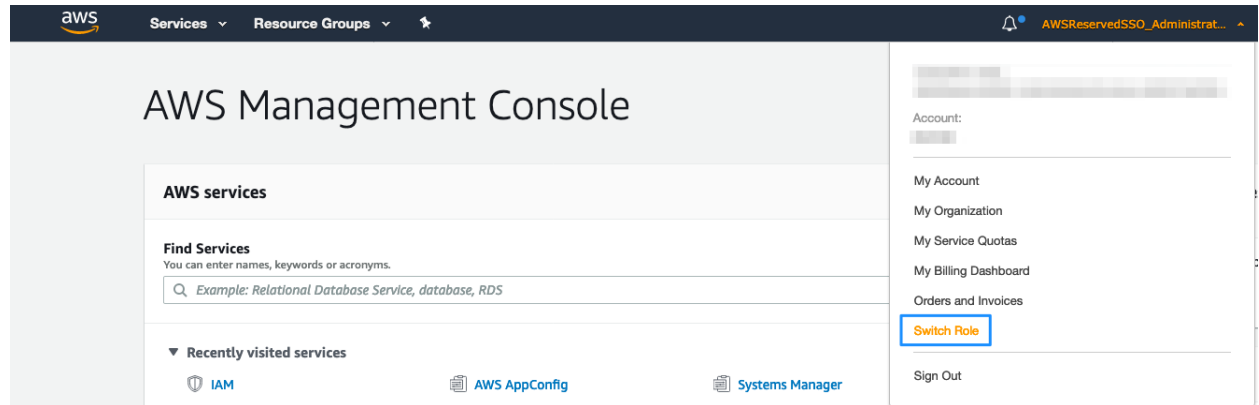
[Sign in using root user email](#)

[Forgot password?](#)

- Access the [S3 console](#). The operation will succeed. Attempt to create a new S3 bucket. The operation will fail. Anne has been provided read only access to S3 via his membership of the Contractors IAM group.
14. Test Anne's ability to assume the `ec2poweruser` role given her **assumerole** tag with a value of "true"
- While still logged in as the IAM user Anne, use the link captured in part four step 12 of this lab to switch roles to ec2poweruser role. Paste the link or type in a browser. It should be of the

format: <https://signin.aws.amazon.com/switchrole?roleName=ec2poweruser&account=youraccountID-or-alias>

Alternatively, use the switch role option under the user menu in the primary AWS console



Make a note of the account ID from the drop down.

Click Switch Role and provide your AccountID or account alias, the role name and a display value and color. Click Switch role.

Switch Role

Allows management of resources across AWS accounts using a single user ID and password. You can switch roles after an AWS administrator has configured a role and given you the account and role details. [Learn more.](#)

Account* ⓘ
Role* ⓘ
Display Name ⓘ
Color a a a a a a

*Required

[Cancel](#)

[Switch Role](#)

Having assumed the ec2poweruser role, proceed to the EC2 console and terminate the remaining EC2 instance named "FINANCE". The operation will succeed. Access the S3 console. The operation will fail. You do not have access to the S3 console. This is because the permissions of your IAM user (e.g. Anne) and any roles that you switch to (e.g. ec2poweruser) are not cumulative. Anne has S3 read access as a result of her membership to the Contractors IAM Group. The ec2poweruser role has access to EC2 but no access to S3. Only one set of permissions is active at a time. When you switch to a role, you temporarily give up your user permissions and work with the permissions that are assigned to the role. When you exit the role, your user permissions are automatically restored.

Congratulations !! You have validated the IAM user Anne has read only access to S3 and validated his ability to assume the ec2poweruser role with its associated permissions. You observed how permissions are not cumulative with role assumption. You used the ec2poweruser role to clean up EC2 resources provisioned to test IAM user access in the lab.

Part 6: Clean up Lab Resources

We will now cleanup the remaining resources provisioned for this lab. Log into the AWS account you used for this lab via the event engine.

1. Access the AWS [IAM console](#) and select Policies from the sidebar. Use the search bar to search for the Policy you created in Part One. Select the radio button next to the Policy you created (named `departmental-ec2-access`) and then select **"Delete"** from the Policy Actions dropdown menu. Accept confirmations to delete the IAM Policy.
2. Access the AWS IAM console and select **Users** from the sidebar. For each user created in this lab (Anne, Sally, John), select the checkmark next to their username and then click **"Delete User"**. Accept confirmations to delete the IAM user identities.
3. Access the AWS IAM console and select **Groups** from the sidebar. Check the checkmark next to the group named Contractors and then select **"Delete Group"** from the Group Actions dropdown menu. Accept confirmations to delete the IAM Group.
4. Access the AWS IAM console and select **Roles** from the sidebar. Use the search bar to search for the Role you created in Part One. Check the checkmark next to the Role named `EC2S3ReadOnly` and then click **"Delete Role"**. Accept confirmations to delete the IAM Role.
5. Access the [EC2 console](#). Validate that the instances named HR and Finance have been terminated successfully.

Additional Resources

[IAM Introduction](#)

[IAM Best Practices](#)

[IAM Policies](#)

[IAM Tutorials](#)