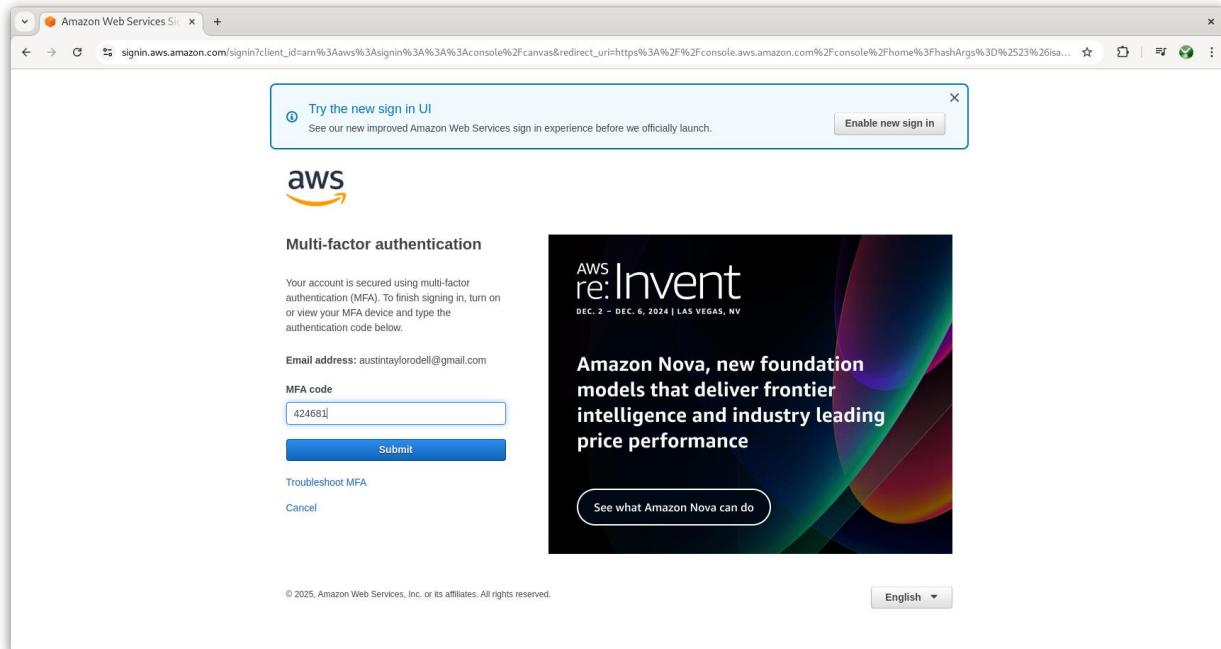
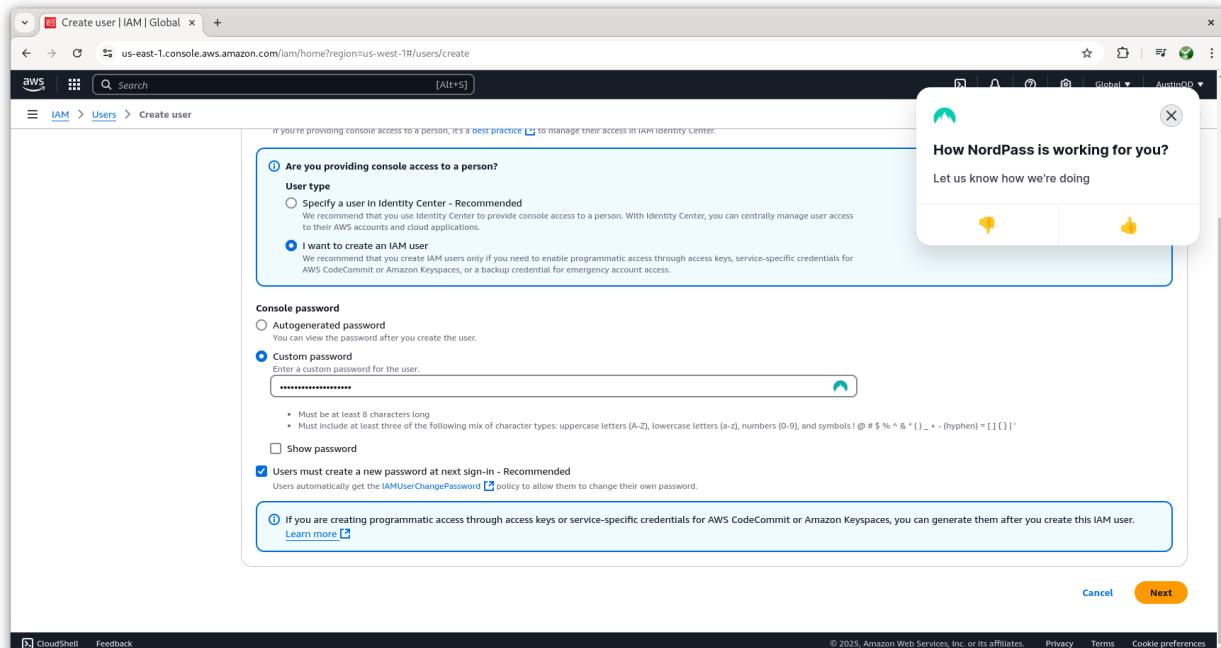


# Security with AWS: Identity and Access Management

By Austin O'Dell



Login to root user using MFA(Never allow root logins without MFA)



Go to iAM>Users>Create User and create a user: SecurityTeamAdmin I chose to use a custom password generated by my password manager Nordpass for ease of use.

**Permissions policies (1/1318)**

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

Policy name	Type	Attached entities
<input type="checkbox"/> AccessAnalyzerServiceRolePolicy	AWS managed	0
<input checked="" type="checkbox"/> AdministratorAccess	AWS managed - Job function	1
<input type="checkbox"/> AdministratorAccess-Amplify	AWS managed	0
<input type="checkbox"/> AdministratorAccess-AWSElasticBeanstalk	AWS managed	0
<input type="checkbox"/> AIOpsAssistantPolicy	AWS managed	0
<input type="checkbox"/> AIOpsConsoleAdminPolicy	AWS managed	0
<input type="checkbox"/> AIOpsOperatorAccess	AWS managed	0
<input type="checkbox"/> AIOpsReadOnlyAccess	AWS managed	0
<input type="checkbox"/> AlexaForBusinessDeviceSetup	AWS managed	0
<input type="checkbox"/> AlexaForBusinessFullAccess	AWS managed	0
<input type="checkbox"/> AlexaForBusinessGatewayExecution	AWS managed	0

Provide the user with necessary permissions. This user will act as an administrator for the security team we are creating in this project.

**Set permissions**

Add user to an existing group or create a new one. Using groups is a best-practice way to manage user's permissions by job functions. [Learn more](#)

**Permissions options**

- Add user to group: Add user to an existing group, or create a new group. We recommend using groups to manage user permissions by job function.
- Copy permissions: Copy all group memberships, attached managed policies, and inline policies from an existing user.
- Attach policies directly: Attach a managed policy directly to a user. As a best practice, we recommend attaching policies to a group instead. Then, add the user to the appropriate group.

**Permissions policies (2/1318)**

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

Policy name	Type	Attached entities
<input type="checkbox"/> AmazonVPCCrossAccountNetworkInterfaceOpe...	AWS managed	0
<input type="checkbox"/> AWS-SSM-DiagnosisAutomation-OperationalAc...	AWS managed	0
<input type="checkbox"/> AWS-SSM-RemediationAutomation-Operational...	AWS managed	0
<input type="checkbox"/> AWSAccountActivityAccess	AWS managed	0
<input checked="" type="checkbox"/> AWSAccountManagementFullAccess	AWS managed	0
<input type="checkbox"/> AWSAccountManagementReadOnlyAccess	AWS managed	0

Next we assign permissions, however since we are creating this user prior to the creation of a group we will attach permissions directly. **AWSAccountManagementFullAccess** policy allows full management of AWS accounts including users and groups.

User created successfully

You can view and download the user's password and email instructions for signing in to the AWS Management Console.

Step 1 Specify user details  
Step 2 Set permissions  
Step 3 Review and create  
Step 4 Retrieve password

**Retrieve password**

You can view and download the user's password below or email users instructions for signing in to the AWS Management Console. This is the only time you can view and download this password.

**Console sign-in details**

Console sign-in URL  
https://816069122057.signin.aws.amazon.com/console

User name  
SecurityTeamAdmin

Console password  
\*\*\*\*\* Show

Email sign-in instructions

Cancel Download .csv file Return to users list

The SecurityTeamAdmin user is created and we're given the sign-in console URL.

You are currently using the improved sign in UI experience. The [improved sign in](#) experience will launch soon. During this time, you can still change back to legacy sign in using the dropdown in the upper right corner.

**IAM user sign in**

Account ID (12 digits) or account alias  
816069122057

IAM username  
SecurityTeamAdmin

Password  
\*\*\*\*\*

Show Password Having trouble?

Sign in

Create a new AWS account

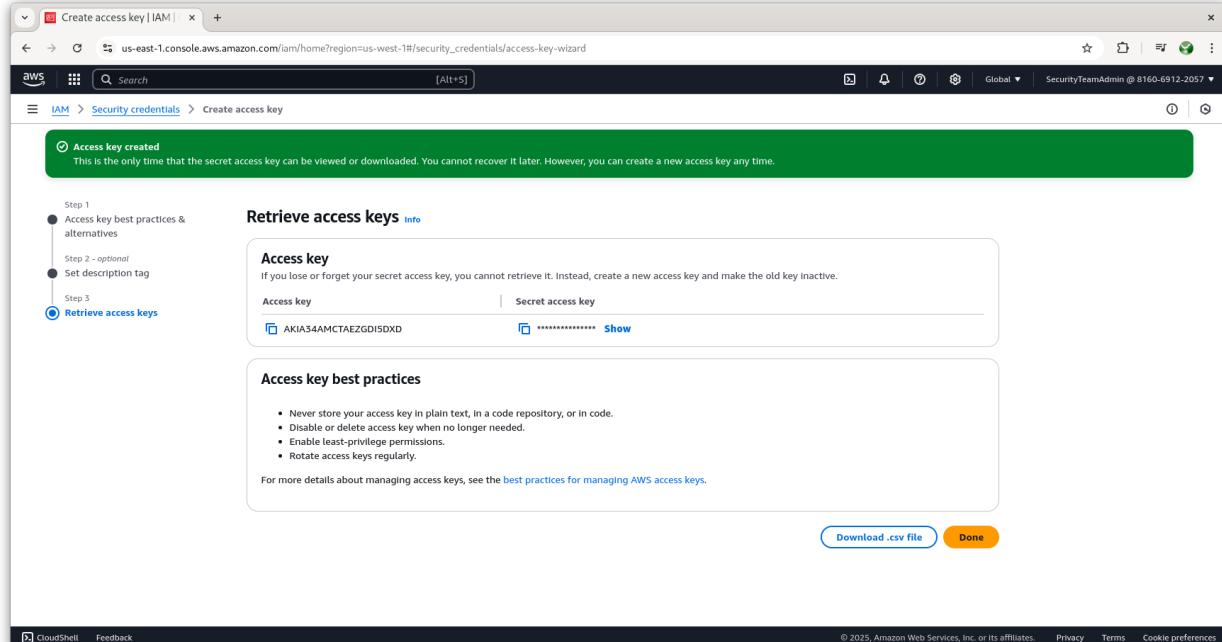
Remember this account

AWS re:Invent DEC. 2 – DEC. 6, 2024 | LAS VEGAS, NV

Amazon Nova, new foundation models that deliver frontier intelligence and industry leading price performance

See what Amazon Nova can do

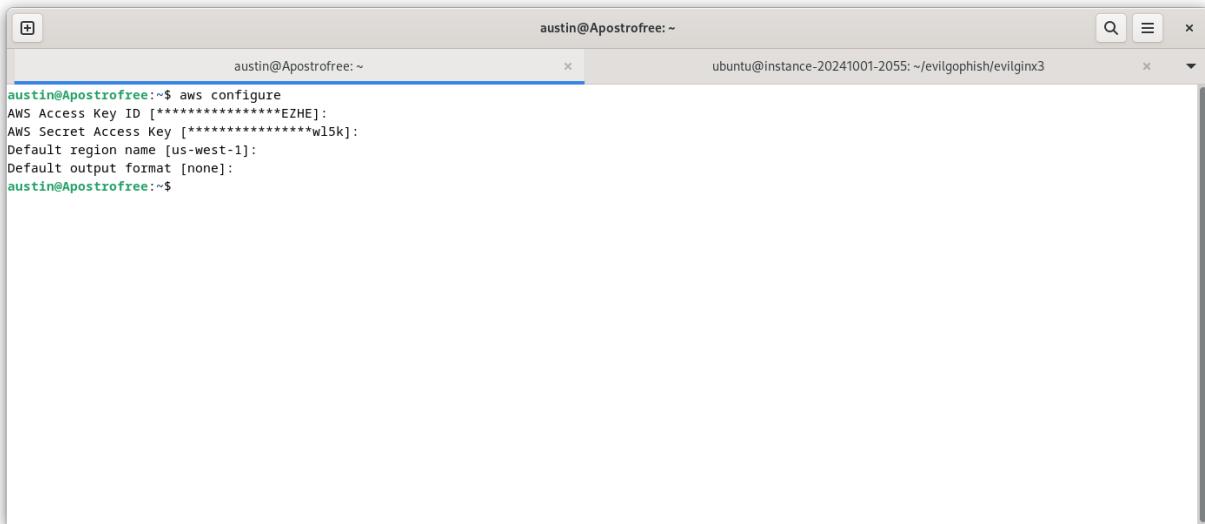
Opening the URL we're presented with the login page for the newly created user.



Next we go to IAM > Security Credentials > Create access key and we will generate an access key to use with the AWS CLI so that we can manage resources from our terminal as the newly created SecurityTeamAdmin user.

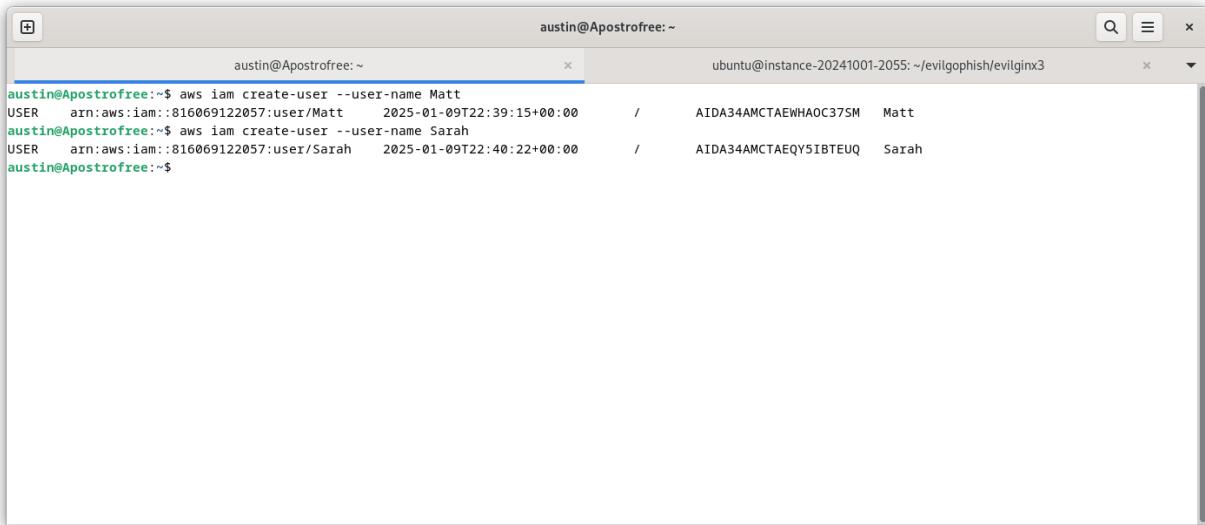
```
austin@aPostrofree:~$ aws configure
AWS Access Key ID [*****5DXD]:
AWS Secret Access Key [*****HZUd]:
Default region name [us-west-1]:
Default output format [None]:
austin@aPostrofree:$
```

Run `aws configure` and input the access key and secret access key we created above for the SecurityTeamAdmin user. The results of running `aws configure` above show that the keys have already been configured and any further aws commands will be run as the SecurityTeamAdmin.



```
austin@Apostrofree:~$ aws configure
AWS Access Key ID [*****EZHE*]:
AWS Secret Access Key [*****w15k*]:
Default region name [us-west-1]:
Default output format [none]:
austin@Apostrofree:~$
```

These keys can be revoked and regenerated, and new keys can be configured for the same user as shown above.



```
austin@Apostrofree:~$ aws iam create-user --user-name Matt
USER arn:aws:iam::816069122057:user/Matt 2025-01-09T22:39:15+00:00 / AIDA34AMCTAEWHAOC375M Matt
austin@Apostrofree:~$ aws iam create-user --user-name Sarah
USER arn:aws:iam::816069122057:user/Sarah 2025-01-09T22:40:22+00:00 / AIDA34AMCTAEQY5IBTEUQ Sarah
austin@Apostrofree:~$
```

Next we use `aws create-user --user-name <user>` as SecurityTeamAdmin to create three users Matt, Sarah, and Deborah.

The screenshot shows the AWS IAM 'Users' page. The left sidebar has 'Identity and Access Management (IAM)' selected. Under 'Access management', 'Users' is also selected. The main area displays a table titled 'Users (6)'. The columns include User name, Path, Group, Last activity, MFA, Password age, Console last sign-in, and Access key ID. The users listed are Deborah, Matt, Sarah, SecurityTeamAdmin, Terraform, and tf-austin. SecurityTeamAdmin is highlighted in blue.

User name	Path	Group	Last activity	MFA	Password age	Console last sign-in	Access key ID
Deborah	/	0	-	-	-	-	-
Matt	/	0	-	-	-	-	-
Sarah	/	0	-	-	-	-	-
SecurityTeamAdmin	/	0	5 minutes ago	Virtual	32 minutes	January 09, 2025, 17:0...	Active - AKIA34AMCTA...
Terraform	/	0	-	-	-	-	-
tf-austin	/	0	Yesterday	-	-	-	Active - AKIA34AMCTA...

The created users are shown graphically in the aws console above. We can also delete users with `aws iam delete-user --user-name Deborah` for instance to delete the user Deborah.

The screenshot shows a terminal window with two tabs. The left tab is for user 'austin@Apostrofree' and the right tab is for user 'ubuntu@instance-20241001-2055'. The command run in the Austin tab is:

```
austin@austin-Apostrofree:~$ aws iam create-group --group-name CloudSecurityTeam
GROUP    arn:aws:iam::816069122057:group/CloudSecurityTeam      2025-01-13T16:52:21+00:00      AGPA34AMCTAETG3PSVC7P  CloudSecurityTeam      /
```

Next we run `aws iam create-group --group-name CloudSecurityTeam` to create a group named CloudSecurityTeam

```
austin@Apostrofree:~$ aws iam add-user-to-group --group-name CloudSecurityTeam --user-name Matt
```

We add Matt and Sarah to the CloudSecurityTeam group with:

*aws iam add-user-to-group --group-name CloudSecurityTeam --user-name <user name>*

The screenshot shows the AWS IAM User Groups page for the 'CloudSecurityTeam' group. The left sidebar shows the navigation path: IAM > User groups > CloudSecurityTeam. The main content area displays the 'Summary' of the group, including the user group name 'CloudSecurityTeam', creation time 'January 13, 2025, 11:52 (UTC-05:00)', and ARN 'arn:aws:iam:816069122057:group/CloudSecurityTeam'. Below this, the 'Users in this group (2)' section lists two users: Matt and Sarah, both of whom are associated with 1 group and have last activity 3 days ago.

User	Groups	Last activity	Creation time
Matt	1	None	3 days ago
Sarah	1	None	3 days ago

Observe that both users have been added to the group at IAM>User Groups>CloudSecurityTeam in the AWS console.

```
austin@Apostrofree:~$ aws iam attach-group-policy --group-name CloudSecurityTeam --policy-arn arn:aws:iam::aws:policy/AmazonS3FullAccess
```

We can attach the AmazonS3FullAccess managed policy to the group by using

*aws iam attach-group-policy --group-name CloudSecurityTeam --policy-arn arn:aws:iam::aws:policy/AmazonS3FullAccess or we can add the policy to the group in the AWS console.*

The screenshot shows the AWS IAM User Groups page for the 'CloudSecurityTeam' group. The 'Permissions' tab is selected. Under 'Permissions policies', there is one policy listed: 'AmazonS3FullAccess'. The ARN of the group is also visible on the right side of the screen.

We see that that the AmazonS3FullAccess permissions policy is attached to the CloudSecurityTeam group on the AWS console above.

The screenshot shows the AWS IAM Policies page. A green banner at the top indicates that the policy 'IAMReadPolicy' has been created. The main table lists one policy: 'IAMReadPolicy' (Type: Customer managed, Used as: None). The left sidebar shows navigation options like Dashboard, Access management, and Access reports.

Next we go to IAM>Policies> Create Policy and create a customer managed policy to allow read only access to all IAM objects and components.

The screenshot shows the 'Attach as a permissions policy' screen. It lists IAM entities: CloudSecurityTeam (selected), Deborah, Matt, Sarah, SecurityTeamAdmin, Terraform, and tf-austin. The 'Attach policy' button is visible at the bottom right.

Attach this policy to the CloudSecurityTeam group.

Policies (1317) [Info](#)

A policy is an object in AWS that defines permissions.

Policy name	Type	Used as	Description
AccessAnalyzerServiceRolePolicy	AWS managed	None	Allow Access Analyzer to analyze resou...
AdministratorAccess	AWS managed - Job function	Permissions policy (2)	Provides full access to AWS services an...
AdministratorAccess-Amplify	AWS managed	None	Grants account administrative permis...
AdministratorAccess-AWSElasticBea...	AWS managed	None	Grants account administrative permis...
AIOpsAssistantPolicy	AWS managed	None	Provides ReadOnly permissions requir...
AIOpsConsoleAdminPolicy	AWS managed	None	Grants full access to Amazon AI Opera...
AIOpsOperatorAccess	AWS managed	None	Grants access to the Amazon AI Opera...
AIOpsReadOnlyAccess	AWS managed	None	Grants ReadOnly permissions to the A...
AlexaForBusinessDeviceSetup	AWS managed	None	Provide device setup access to AlexaFo...
AlexaForBusinessFullAccess	AWS managed	None	Grants full access to AlexaForBusiness ...
AlexaForBusinessGatewayExecution	AWS managed	None	Provide gateway execution access to A...
AlexaForBusinessLifesizeDelegatedA...	AWS managed	None	Provide access to Lifesize AVS devices

To view this resource you are now in the N. California (us-west-1) region. Previously N. Virginia (us-east-1)

financial-data-secure-company325 [Info](#)

Objects (2) [Info](#)

Objects are the fundamental entities stored in Amazon S3. You can use [Amazon S3 inventory](#) to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. [Learn more](#)

Name	Type	Last modified	Size	Storage class
house_sales_prices.csv	csv	January 13, 2025, 17:14:24 (UTC-05:00)	85.9 KB	Standard
sample_financial_data.csv	csv	January 13, 2025, 17:14:51 (UTC-05:00)	128.4 KB	Standard

Next we upload and rename the sample confidential financial data that we will be working with.

**Add permissions**

Permissions policies (1/1021)

Policy name	Type	Description
<input checked="" type="checkbox"/> <a href="#">AmazonS3FullAccess</a>	AWS managed	Provides full access to all buckets via t...
<input type="checkbox"/> <a href="#">AmazonS3ObjectLambdaExecutionRolePolicy</a>	AWS managed	Provides AWS Lambda functions permil...
<input type="checkbox"/> <a href="#">AmazonS3OutpostsFullAccess</a>	AWS managed	Provides full access to Amazon S3 on ...
<input type="checkbox"/> <a href="#">AmazonS3ReadOnlyAccess</a>	AWS managed	Provides read only access to Amazon S...
<input type="checkbox"/> <a href="#">AmazonS3ReadonlyAccess</a>	AWS managed	Provides read only access to all bucket...
<input type="checkbox"/> <a href="#">AmazonS3TablesFullAccess</a>	AWS managed	Provides full access to all S3 table bu...
<input type="checkbox"/> <a href="#">AmazonS3TablesReadOnlyAccess</a>	AWS managed	Provides read only access to all S3 tabl...
<input type="checkbox"/> <a href="#">AmazonSageMakerAdmin-ServiceCatalogProductsServiceRo...</a>	AWS managed	Service role policy used by the AWS Se...
<input type="checkbox"/> <a href="#">AmazonSageMakerCanvasAISevicesAccess</a>	AWS managed	Provides permissions for Amazon Sage...
<input type="checkbox"/> <a href="#">AmazonSageMakerCanvasBedrockAccess</a>	AWS managed	This policy grants permissions to use A...
<input type="checkbox"/> <a href="#">AmazonSageMakerCanvasDataPrepFullAccess</a>	AWS managed	Provides full access to Amazon SageM...
<input type="checkbox"/> <a href="#">AmazonSageMakerCanvasDirectDeployAccess</a>	AWS managed	Allows Amazon SageMaker Canvas to c...
<input type="checkbox"/> <a href="#">AmazonSageMakerCanvasEMRServerlessExecutionRolePolicy</a>	AWS managed	This policy grants permissions to Ama...
<input type="checkbox"/> <a href="#">AmazonSageMakerCanvasForecastAccess</a>	AWS managed	This policy grants permissions commo...
<input type="checkbox"/> <a href="#">AmazonSageMakerCanvasFullAccess</a>	AWS managed	Provides full access to Amazon SageM...

**Identity and Access Management (IAM)**

**Roles (3)**

Role EC2toS3Role created.

Role name	Trusted entities	Last activity
<a href="#">AWSServiceRoleForSupport</a>	AWS Service: support (Service-Linker)	-
<a href="#">AWSServiceRoleForTrustedAdvisor</a>	AWS Service: trustedadvisor (Service)	-
<a href="#">EC2toS3Role</a>	AWS Service: ec2	-

**Roles Anywhere**

Authenticate your non AWS workloads and securely provide access to AWS services.

**Access AWS from your non AWS workloads**

Operate your non AWS workloads using the same authentication and authorization strategy that you use within AWS.

**X.509 Standard**

Use your own existing PKI infrastructure or use [AWS Certificate Manager Private Certificate Authority](#) to authenticate identities.

**Temporary credentials**

Use temporary credentials with ease and benefit from the enhanced security they provide.

Next we create an IAM role to allow an EC2 instance to access S3 data named EC2toS3 role.

The screenshot shows the AWS EC2 Instances "Launch an instance" page. At the top, a green success message states "Successfully initiated launch of instance i-0ad668db4c31b7ef2". Below this, there's a "Launch log" section with a link to "Launch log". Under "Next Steps", there are several options: "Create billing and free tier usage alerts", "Connect to your instance", "Connect an RDS database", "Create EBS snapshot policy", "Manage detailed monitoring", "Create Load Balancer", "Create AWS budget", and "Manage CloudWatch alarms". Each option has a corresponding button like "Create billing alerts", "Connect to instance", "Create an RDS database", etc.

The screenshot shows the AWS EC2 Instances "Modify IAM role" page. It displays the "Instance ID" as i-0ad668db4c31b7ef2 (List 53 Buckets). Under the "IAM role" section, it says "Select an IAM role to attach to your instance or create a new role if you haven't created any. The role you select replaces any roles that are currently attached to your instance." A dropdown menu shows "EC2toS3Role" selected. There are buttons for "Create new IAM role" and "Update IAM role".

Next we create an EC2 instance and attach the role we previously created to it.

```

Amazon Linux 2023
https://aws.amazon.com/linux/amazon-linux-2023

[ec2-user@ip-172-31-4-225 ~]$ aws s3 ls
2024-12-27 14:53:44 detectionlabimagesimport
2025-01-13 22:07:35 financial-data-secure-company325
[ec2-user@ip-172-31-4-225 ~]$

```

i-0ad668db4c31b7ef2 (List S3 Buckets)

Here we see that the EC2 instance can access the S3 data by running `aws s3 ls` from the instance.

```

ami-id
ami-launch-index
ami-manifest-path
block-device-mapping/
events/
hostname
iam/
identity-credentials/
instance-action
instance-id
instance-lifecycle
instance-type
local-hostname
local-ipv4
mac
managed-ssh-keys/
metrics/
network/
placement/
profile
public-hostname
public-ipv4
public-keys/
reservation-id
security-groups
services/
system[ec2-user@ip-172-31-4-225 curl http://169.254.169.254/latest/meta-data/public-hostname
ec2-13-52-248-8.us-west-1.comcurl http://169.254.169.254/latest/meta-data/instance-type
t2.micro[ec2-user@ip-172-31-4-2 curl http://169.254.169.254/latest/meta-data/iam/info
{
  "Code" : "Success",
  "LastUpdated" : "2025-01-13T22:23:45Z",
  "InstanceProfileArn" : "arn:aws:iam::1816060122057:instance-profile/EC2toSSRole",
  "InstanceProfileId" : "ATPA34AMCTAE2QF7NSBIV"
}[ec2-user@ip-172-31-4-225 ~]$ curl http://169.254.169.254/latest/meta-data/iam/securitycredentials/
<?xml version="1.0" encoding="iso-8859-1"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"

```

i-0ad668db4c31b7ef2 (List S3 Buckets)

We can run `curl http://169.254.169.254/latest/meta-data/iam/info` to see the role that the EC2 has assumed.

```

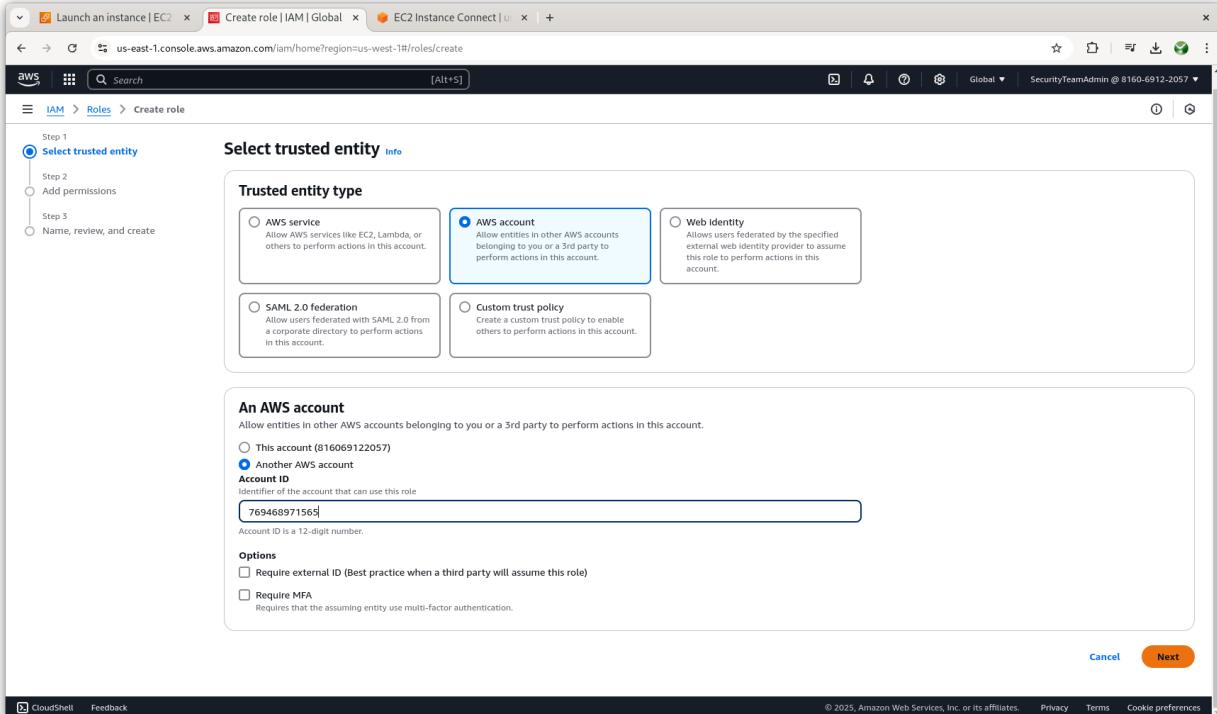
Launch an instance | EC2 | Instances | EC2 | us-west-1 | EC2 Instance Connect | 
us-west-1.console.aws.amazon.com/ec2-instance-connect/ssh/home?addressFamily=ipv4&connType=standard&instanceId=i-0ad668db4c31b7ef2&osUser=ec2-user&region=us-west-1&sshPort=22
aws Search [Alt+S] United States (N. California) SecurityTeamAdmin @ 8160-6912-2057
To view this resource you are now in the N. California (us-west-1) region. Previously N. Virginia (us-east-1)

[ec2-user@ip-172-31-4-225 ~]$ curl http://169.254.169.254/latest/meta-data/iam/security-credentials/EC2toS3Role/
{
    "Code" : "Success",
    "LastUpdated" : "2025-01-13T22:42:02Z",
    "Type" : "AWS-HMAC",
    "AccessKeyId" : "ASIA34AMCTAEYXTA17HQ",
    "SecretAccessKey" : "r-8KmlZD1kNpwvUpSbuZUfTuHNqa7bad/zZ1+v",
    "Token" : "I0oJb3jzzlXvJEA8aCVxLXdC3QMSJ1MEYC1QJCuyRFWNUl3FrRz0nNSTCjnjsiO1UFcje2xtkwIhLBFP7RM075pk2av5k5TpWJgqeQH9v5ykGZ+AZGLRKsqFCPj//////////wEQABoMDE2MDY5MTIyMDU3IgZK1fx5hmeREfuVuiAmUQSUS1zTXN8ayRcvJvbUobckEnwyB1f6knkJK63sHrLoCHrrqNPxpjLN+pY+cwBNLX/lb+LoB7bUntqu95eZC2vUrTMQNsHMe2t3z2EyrnlkssMtjz384vr+q1fxqytYQbPStKK-dkmZohnZ3Tc8HAPaPVLAz2qqDah+M2Bhf4HJTQEPDsa7gAx0x60UUUx0q98/6W0uwf5Q1/10qY102hwZNBJLrcqXYt8f3KImecRvk5q+A631DEWMn07xeGtWT4d1qug402GK9tcs0pVvdzL1+PzmannYAlp7PuzeMBq4GbWSs53vLYNyY96cmH01TmfRtha/uwnnn1hKn0@0tMfsgiv8vTb34j9+R0LPtQDRh6sDxudvbaTUJN6h1ffDVOGwZo6taMX2y7/FUr3dlsrqQwyhltxU4y7yr84xehxRt1T1qg2ByGvcbIW3Gdtk0XtVf4p2n70eHs0zH4ob1ks1tf0u0L+b1ydgkv8E7ryB10Rzr3LRau//es0Jt92IYtL2dsVx2t0tAc2hcCm98LnsHEmxSS1Iwfb3NzDEHXRM3ps3B4xH-E6WEHJHsavnAxy1zd9K7yG3tp1uHv1r1kdiQoIMhVE84fsBnxKCJS+Ixiaw1oroy72edp0tK19yJYyjgwjdjEkD5157LnhH/Jfc1c2jbT/a7Ngaw-NRtJFg+5PoDADZ03100CCuy15pUDl16sA60978/XwCY2h8DFpW66j7bpqtwcy/HpjtVU47MYE1byen4ZzEkiDeJHLwrtQbQAlavkInx4b770oP4yFmpYqp/xwSk8suhkJbhNx1h2J1V/dVCBeuQyJ9HPGKlrwG0tAB7xtXo/Adci5uYJFl11RJf4t7h1uHmUmKwYHOXJ06INT52wCdV//224WhvPvgHtCokxcjCv20pn1g0pAtUAYppnqUsHvn0GaNh8H+eioxdmnefrRq956YSoeV6pPxMs98ubJQsh3kcheavq2jR7mVK9ry/v8he8khf45nbs9skc881IHACjCluimhdeGg5rIaMPVsZgw93y13680uLe73ym6WMkynzSH1ndsa=",
    "Expiration" : "2025-01-14T04:58:45Z"
}[ec2-user@ip-172-31-4-225 ~]$ 

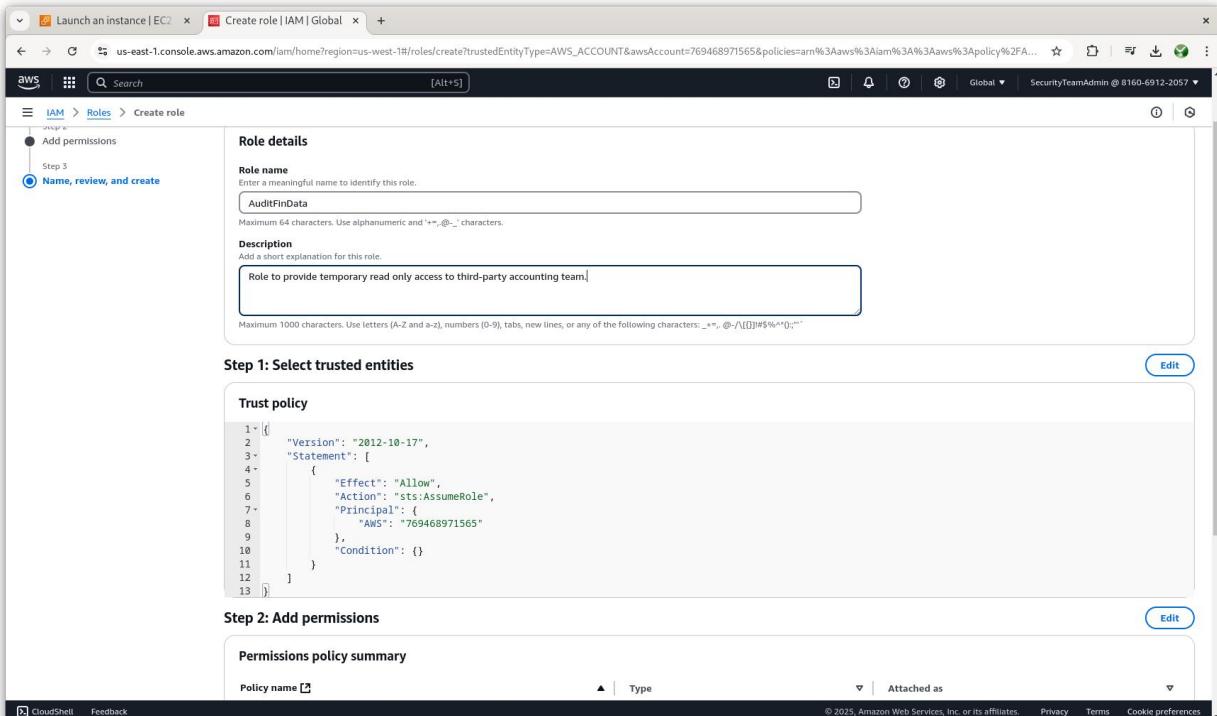
```

i-0ad668db4c31b7ef2 (List S3 Buckets)

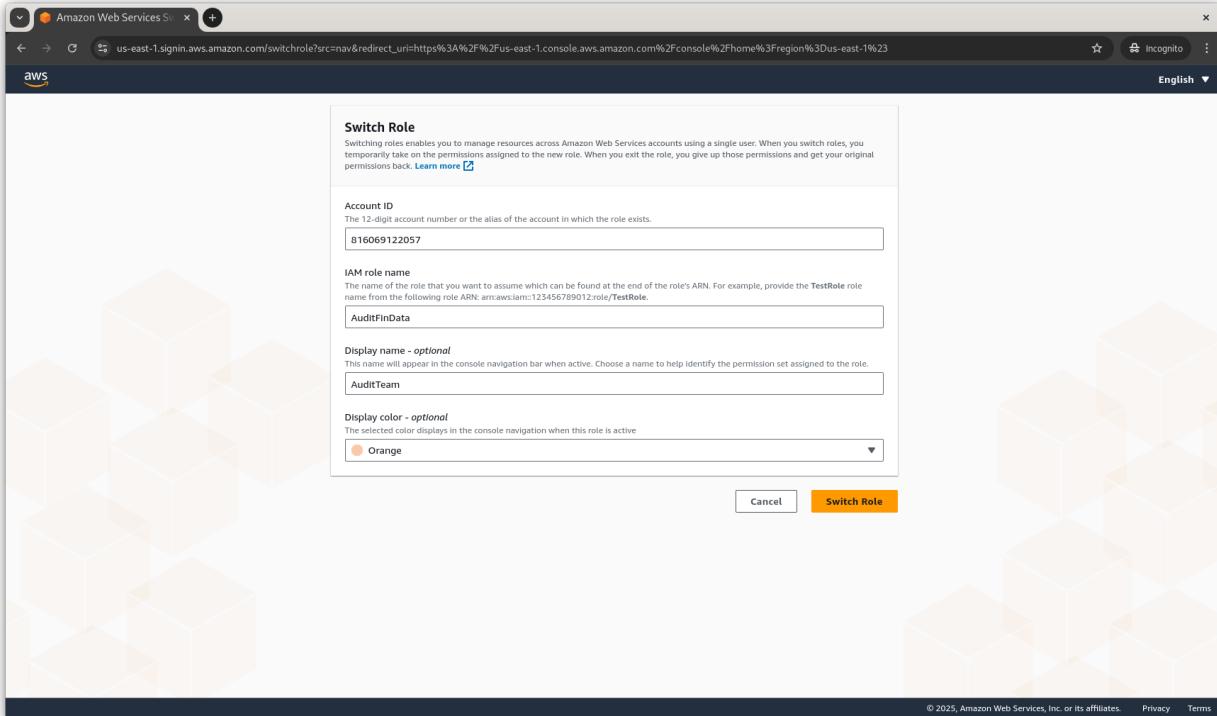
We can run curl `http://169.254.169.254/latest/meta-data/iam/security-credentials/` to see the temporary security token used by the EC2 instance for its assumed role. Using IAM roles is a security best practice because roles are easy to revoke and access tokens are temporary.



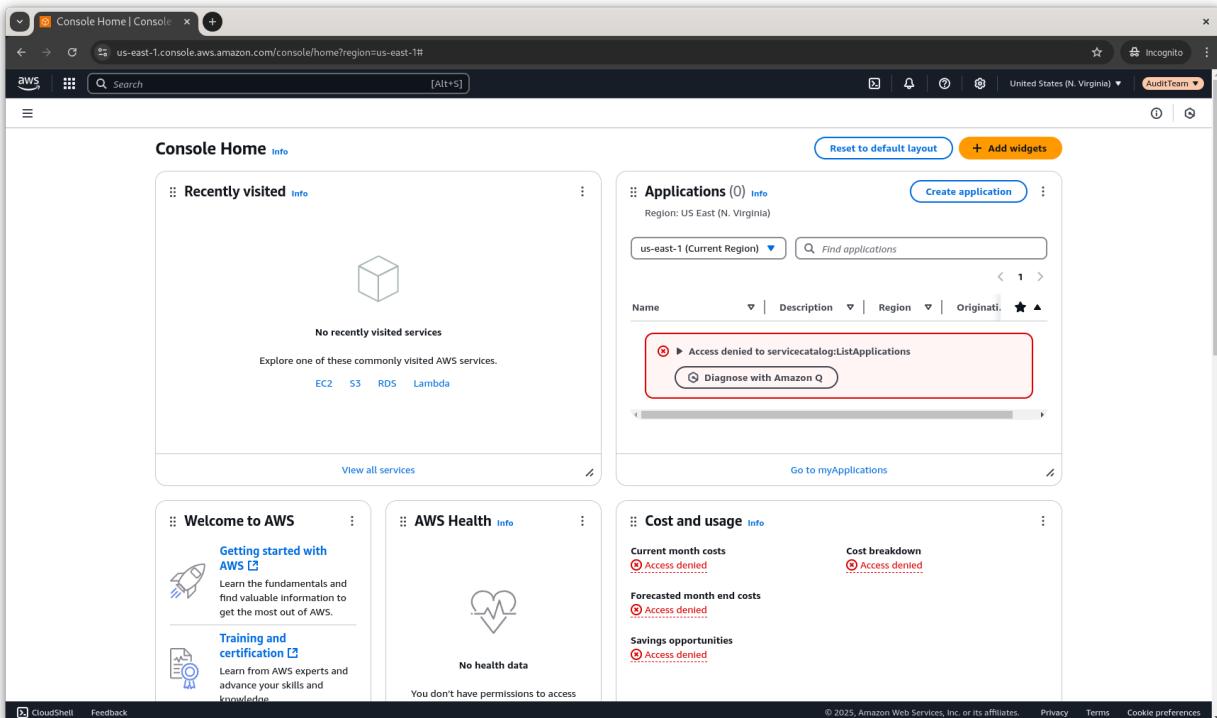
Next we create a role for an external auditor to assume to access the financial data uploaded to our S3 without having to provide access to our AWS account. Under create role > Select trusted entity we select AWS account and select Another AWS account and input the account ID of the external auditor.



We name the role AuditFinData and provide read only access to S3 data.



After logging in to the auditor account in an incognito window we can switch roles to the newly created AuditFinData role.



We see that the console for the auditor is filled with Access Denied warning because we have limited their access to specific resources.

**Amazon S3**

**General purpose buckets**

**General purpose buckets (2)**

Name	AWS Region	IAM Access Analyzer	Creation date
detectionlabimageimport	US East (Ohio) us-east-2	<a href="#">View analyzer for us-east-2</a>	December 27, 2024, 09:53:44 (UTC-05:00)
financial-data-secure-company325	US West (N. California) us-west-1	<a href="#">View analyzer for us-west-1</a>	January 13, 2025, 17:07:34 (UTC-05:00)

However the auditor is able to list files on S3.

**EC2 Global View**

**Resources**

Instances (running)	0	Auto Scaling Groups	① API Error	Capacity Reservations	① API Error
Dedicated Hosts	① API Error	Elastic IPs	① API Error	Instances	① API Error
Key pairs	① API Error	Load balancers	① API Error	Placement groups	① API Error
Security groups	① API Error	Snapshots	① API Error	Volumes	① API Error

**Launch instance**

To get started, launch an Amazon EC2 Instance, which is a virtual server in the cloud.

**Service health**

An error occurred: An error occurred retrieving service health information

**Zones**

An error occurred: An error occurred retrieving service health information

**Account attributes**

An error occurred: An error occurred checking for a default VPC

They are not allowed to modify, access, or launch EC2 instances.

The screenshot shows the AWS S3 console interface. On the left, there's a sidebar with various navigation options like 'General purpose buckets', 'Storage Lens', and 'AWS Marketplace for S3'. The main area is titled 'financial-data-secure-company325' and shows a list of objects. There are two CSV files listed:

Name	Type	Last modified	Size	Storage class
house_sales_prices.csv	csv	January 13, 2025, 17:14:24 (UTC-05:00)	85.9 KB	Standard
sample_financial_data.csv	csv	January 13, 2025, 17:14:51 (UTC-05:00)	128.4 KB	Standard

But they can access the required financial data.

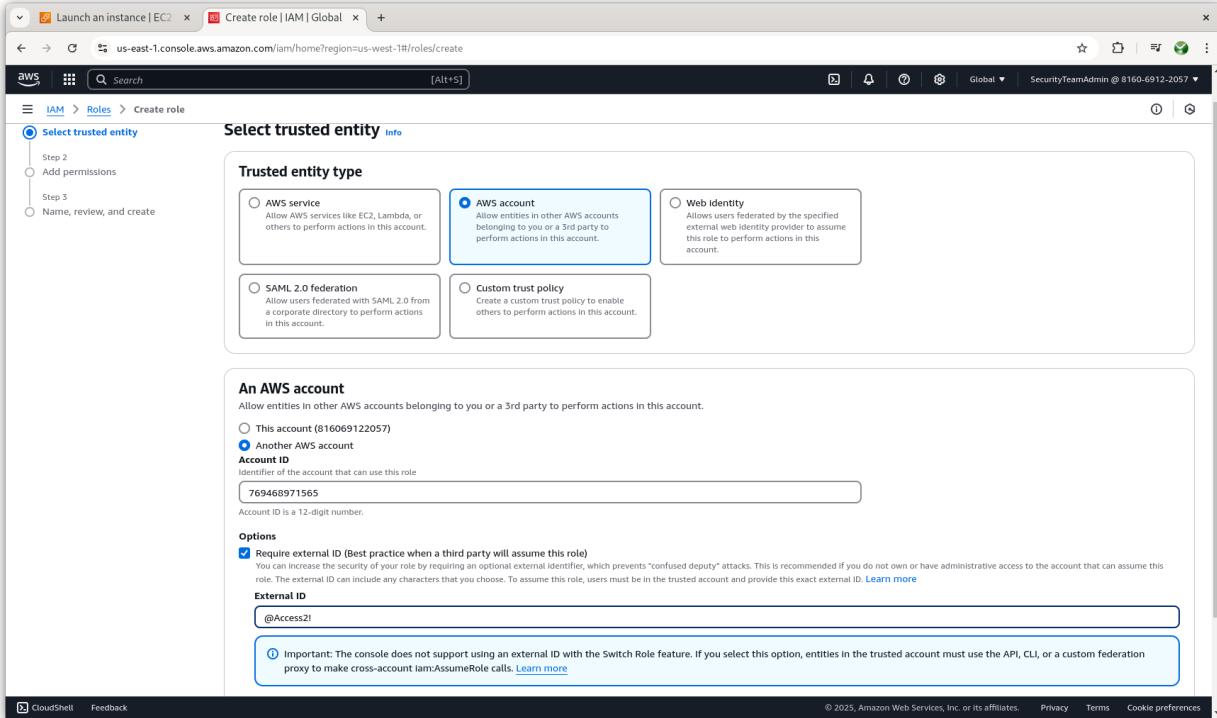
The screenshot shows the 'Upload objects - S3 buck.' page in the AWS S3 console. A red banner at the top says 'Upload failed' with a link to 'Error column in the Files and folders table'. Below it, a summary table shows the upload status:

Destination	Succeeded	Failed
s3://financial-data-secure-company325	0 files, 0 B (0%)	1 file, 4.4 KB (100.00%)

Under the 'Files and folders' tab, a table lists the uploaded file:

Name	Folder	Type	Size	Status	Error
main.tf	-	-	4.4 KB	Failed	Access Denied

However, as shown, the role cannot upload files to S3.



Next we create another role with an External ID set that will need to be provided when assuming the role. This is considered best practice when providing access to a third party.

Launch an instance | EC2 | Create role | IAM | Global | Policies | IAM | Global

us-east-1.console.aws.amazon.com/iam/home?region=us-west-1#/roles/create?trustedEntityType=AWS\_ACCOUNT&awsAccount=769468971565&isThirdParty=true&externalId=%40Access2%21

**IAM > Roles > Create role**

SAML 2.0 federation  
Allows users federated with SAML 2.0 from a corporate directory to perform actions in this account.

Custom trust policy  
Create a custom trust policy to enable others to perform actions in this account.

**An AWS account**  
Allow entities in other AWS accounts belonging to you or a 3rd party to perform actions in this account.

This account (816069122057)

Another AWS account

**Account ID**  
Identifier of the account that can use this role  
**769468971565**  
Account ID is a 12-digit number.

**Options**

**Require external ID (Best practice when a third party will assume this role)**  
You can increase the security of your role by requiring an optional external identifier, which prevents "confused deputy" attacks. This is recommended if you do not own or have administrative access to the account that can assume this role. The external ID can include any characters that you choose. To assume this role, users must be in the trusted account and provide this exact external ID. [Learn more](#)

**External ID**  
@Access2!

**Important:** The console does not support using an external ID with the Switch Role feature. If you select this option, entities in the trusted account must use the API, CLI, or a custom federation proxy to make cross-account iam:AssumeRole calls. [Learn more](#)

**Require MFA**  
Requires that the assuming entity use multi-factor authentication.

**Cancel** **Next**

CloudShell Feedback © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Launch an instance | EC2 | Create role | IAM | Global | Policies | IAM | Global

us-east-1.console.aws.amazon.com/iam/home?region=us-west-1#/roles/create?trustedEntityType=AWS\_ACCOUNT&awsAccount=769468971565&isThirdParty=true&externalId=%40Access2%21&policies=arn%3Aa...  
Global SecurityTeamAdmin @ 8160-6912-2057

**IAM > Roles > Create role**

Add permissions

Name, review, and create

**Step 5: Name, review, and create**

**Role details**

**Role name**  
Enter a meaningful name to identify this role.  
**AuditFinDataExtID**  
Maximum 64 characters. Use alphanumeric and '+-,@\_-' characters.

**Description**  
Add a short explanation for this role.  
Allow audit team admin read-only access to financial data with an external ID.

**Step 1: Select trusted entities**

**Trust policy**

```

1- [{
2-   "Version": "2012-10-17",
3-   "Statement": [
4-     {
5-       "Effect": "Allow",
6-       "Action": "sts:AssumeRole",
7-       "Principal": {
8-         "AWS": "769468971565"
9-       },
10-      "Condition": {
11-        "StringEquals": {
12-          "sts:ExternalId": "@Access2!"
13-        }
14-      }
15-    }
16-  ]
17- }

```

**Step 2: Add permissions**

CloudShell Feedback © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

**Add permissions**

Step 1 Select trusted entity  
Step 2 Add permissions  
Step 3 Name, review, and create

**Permissions policies (1/1022)**

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

Filter by Type: Customer managed | 2 matches

Policy name	Type	Description
IAMReadPolicy	Customer managed	Policy to allow read only access to resour...
S3ListAndRead	Customer managed	Policy to allow S3 resources to be listed ...

**Set permissions boundary - optional**

Cancel Previous Next

**Identity and Access Management (IAM)**

Role AuditFinDataExtID created.

**Roles (5)**

An IAM role is an identity you can create that has specific permissions with credentials that are valid for short durations. Roles can be assumed by entities that you trust.

Role name	Trusted entities	Last activity
AuditFinData	Account: 769468971565	14 hours ago
AuditFinDataExtID	Account: 769468971565	-
AWSServiceRoleForSupport	AWS Service: support (Service-Linker)	-
AWSServiceRoleForTrustedAdvisor	AWS Service: trustedadvisor (Service)	-
EC2toS3Role	AWS Service: ec2	14 hours ago

**Access management**

- User groups
- Users
- Roles**
- Policies
- Identity providers
- Account settings
- Root access management

**Access reports**

- Access Analyzer
- External access
- Unused access
- Analyzer settings
- Credential report
- Organization activity
- Service control policies
- Resource control policies

**Access AWS from your non AWS workloads**

Operate your non AWS workloads using the same authentication and authorization strategy that you use within AWS.

**X.509 Standard**

Use your own existing PKI infrastructure or use [AWS Certificate Manager Private Certificate Authority](#) to authenticate identities.

**Temporary credentials**

Use temporary credentials with ease and benefit from the enhanced security they provide.

```

austin@Apostrofree:~ $ aws s3 ls
2024-12-27 09:53:44 detectionlabimagesimport
2025-01-13 17:07:35 financial-data-secure-company325
austin@Apostrofree:~ $ aws configure
AWS Access Key ID [*****EZHE]: AKIA3GJ7MXIW7JTNHKN6
AWS Secret Access Key [*****w15k]: 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
Default region name [us-west-1]:
Default output format [text]:
austin@Apostrofree:~ $ aws sts get-caller-identity
arn:aws:iam::769468971565:user/AuditTeamAdmin AIDA3GJ7MXIW5FXVP4E4P
austin@Apostrofree:~ $ aws s3 ls
austin@Apostrofree:~ $ aws sts assume-role --role-arn arn:aws:iam::816069122057:role/AuditFinDataExtID --role-session-name auditdemo

An error occurred (AccessDenied) when calling the AssumeRole operation: User: arn:aws:iam::769468971565:user/AuditTeamAdmin is not authorized to perform: sts:AssumeRole on resource: arn:aws:iam::816069122057:role/AuditFinDataExtID
austin@Apostrofree:~ $

```

After setting the appropriate access tokens for the auditing team as shown by `aws sts get-caller-identity` in the AWS CLI we first attempt to assume the newly created role without the external ID and are met with an Access Denied error.

```

austin@Apostrofree:~ $ aws sts assume-role --role-arn arn:aws:iam::816069122057:role/AuditFinDataExtID --role-session-name auditdemo --external-id @Access2!
An error occurred (ValidationException) when calling the AssumeRole operation: 1 validation error detected: Value '@Access2!' at 'externalId' failed to satisfy constraint: Member must satisfy regular expression pattern: [w+=,.@:/\-\*]
austin@Apostrofree:~ $ aws sts assume-role --role-arn arn:aws:iam::816069122057:role/AuditFinDataExtID --role-session-name auditdemo --external-id "Access2"!
An error occurred (ValidationException) when calling the AssumeRole operation: 1 validation error detected: Value '@Access2!' at 'externalId' failed to satisfy constraint: Member must satisfy regular expression pattern: [w+=,.@:/\-\*]
austin@Apostrofree:~ $ aws sts assume-role --role-arn arn:aws:iam::816069122057:role/AuditFinDataExtID --role-session-name auditdemo --external-id "Access2\!"
An error occurred (ValidationException) when calling the AssumeRole operation: 1 validation error detected: Value '@Access2\!' at 'externalId' failed to satisfy constraint: Member must satisfy regular expression pattern: [w+=,.@:/\-\*]
austin@Apostrofree:~ $ aws sts assume-role --role-arn arn:aws:iam::816069122057:role/AuditFinDataExtID --role-session-name auditdemo --external-id Access2225
ASSUMEDROLEUSER arn:aws:sts::816069122057:assumed-role/AuditFinDataExtID@auditdemo AR0A34AMCTAEVSPALHSB:auditdemo
CREDENTIALS ASIA34AMCTAE2TGLZ53N 2025-01-14T14:46:51+00:00 OC80rhrbkNn3v9LMXoic5isMupkrhYBPyYd1Myhe IQojb3JpZ2luX2VjEB4aCXVzLXd1c3QtMSJ1IMEYCIQDhqvd39zBjw0hSlazo2r08p1XMarH1KaZCV/FMoogIhAK+DWZSfpw300UQXMeN93+Juxc0XC2GBRCf
zBjw0hSlazo2r08p1XMarH1KaZCV/FMoogIhAK+DWZSfpw300UQXMeN93+Juxc0XC2GBRCfR0mWid4nKpYCCbcABoAbM0DE2MDY5MTyjMDU3ig27ywdGHuzH7y6AQ1kq8w9bxvKebjjuMn0sBle8YTCjwRkf0o1JmtjQ+x45/zWNBoqkyEjtIa+PBqqP4Eo6eES1wg4g0jv517mhug5074b27NuXE/cPFkRT6WS4KF9
45/zWNBoqkyEjtIa+PBqqP4Eo6eES1wg4g0jv517mhug5074b27NuXE/cPFkRT6WS4KF9
6ev0P13UKLiq2KM1c78aUtniW96IzqayhBQz+1xt4e16m36qpfbjNzIzNGZy5APyra1l9vY8fxQjlxU0vfeJgbXGubApbP13tYPAQNf+F2Nx4Qz2oAw9y9WzVAy6nGtQ5j/Uwcq9ZLY96nPgi3Cg6etSts678Ub215
q0TTfkXMHXH2zWcPo6tJcy/ldD1f73aU0UT18b0p94HwKhns1JEH5Soz1E45wzhFmxskoth/YL27TP1zXXCG/rKe10myeEBAVbyLt2a0r7Lq21tB8dB3Lc5x0IPvUs1zqTVovxpvdVf8z2huNs/E1zSH/S0dmSkqc=
austin@Apostrofree:~ $ export AWS_ACCESS_KEY_ID=ASIA34AMCTAE2TGLZ53N
austin@Apostrofree:~ $ export AWS_SECRET_ACCESS_KEY=C080rhrbkNn3v9LMXoic5isMupkrhYBPyYd1Myhe
austin@Apostrofree:~ $ export AWS_SESSION_TOKEN=Iqojb3JpZ2luX2VjEB4aCXVzLXd1c3QtMSJ1IMEYCIQDhqvd39zRbjw0hSlazo2r08p1XMarH1KaZCV/FMoogIhAK+DWZSfpw300UQXMeN93+Juxc0XC2GBRCf
RQmWid4nKpYCCbcABoAbM0DE2MDY5MTyjMDU3ig27ywdGHuzH7y6AQ1kq8w9bxvKebjjuMn0sBle8YTCjwRkf0o1JmtjQ+x45/zWNBoqkyEjtIa+PBqqP4Eo6eES1wg4g0jv517mhug5074b27NuXE/cPFkRT6WS4KF9
x1xLG34t1sRmqfZ17+yv1+Fa45QDspriBmf5fKyqy16X/8iumxjQWeppmgkA13zrHEVpjeE74mCaNl12yjxqMP51QM08ubR6ev0P13UKLiq2KM1c78aUtnw961zqayhBQz+1xt4e16m36qpfbjNzIzNGZy5APyra1l9vY8
FoxJxU0vfeJgbXGubApbP13tYPAQNf+f2Nx4Qz2oAw9y9WzVAy6nGtQ5j/Uwcq9ZLY96nPgi3Cg6etSts678Ub215q0TTfkXMHXH2zWcPo6tJcy/ldD1f73aU0UT18b0p94HwKhns1JEH5Soz1E45wzhFmxskoth/YL27TP1zXXCG/rKe10myeEBAVbyLt2a0r7Lq21tB8dB3Lc5x0IPvUs1zqTVovxpvdVf8z2huNs/E1zSH/S0dmSkqc=
austin@Apostrofree:~ $

```

When we try again with the `--external-id` set we are provided with the appropriate credentials. We can then use `export` to set these corresponding variables and use the AWS CLI as the IAM role we created.

```
austin@aPostrofree:~
```

```
austin@aPostrofree:~$ aws sts assume-role --role-arm arn:aws:iam::816069122057:role/AuditFinDataExtID --role-session-name auditdemo --external-id "@Access2\!"
```

An error occurred (ValidationException) when calling the AssumeRole operation: 1 validation error detected: Value '@Access2\!' at 'externalId' failed to satisfy constraint: Member must satisfy regular expression pattern: [\\w+=.,@\\/-]\*

```
austin@aPostrofree:~$ aws sts assume-role --role-arm arn:aws:iam::816069122057:role/AuditFinDataExtID --role-session-name auditdemo --external-id Access2225
```

```
ASSUMEDROLEUSER arn:aws:sts::816069122057:assumed-role/AuditFinDataExtID/auditdemo AROA34AMCTAEVTSFALHB5:auditdemo
```

```
CREDENTIALS ASIA34AMCTAE2GLZ53N 2025-01-14T14:46:51+00:00 0C80rbkKn3v9LMxrc5isMupKhrYBpWyydMyeh IQoJb3jpZ2luX2VjEB4aCXvzLxd1c3QtMSJIMELYCIQDhpqCd39zRbjw0hSlazo2r08p1X7MarH1KaZCV/FMoogIhAK+DWZSfpw300uQXMeN93+Juxc0XC2GBRCfRQmWid4HnkPycCcQaB0MDE2MDY5MTyMDU3IgZ7ywDghuZH7y6Aq1kq8wE9bxvKbJjuMnfSb0lE8YTCjWrkf0o1lmtjQ+x45/zWNBoqqkYeJTr1a:P8qqaP4Eo6eE51gBa4q0zJ517mhug5074bzJ27NuxE/cPfKRT6wSAkF9xLx8LG3411srMqF217+yv1+Faa50Dspz8Mb5kyq16X/8iumXjQWeppmgkAl3x0HEVpJeE74mcAnLm12yXjqvMP51Qm08ubR6evPi13UkLiqzKMC78AuTxmW96IzqayBQz1Xt+e16m3eqpbfbjNzNG2y5Ap+y119yvX8pxqjxL0ufefJgBXGubbApp6P13tYPAQNF+r2Nx+Qz0Aw9yWzAVy6nAGQF5j/Uwcq9zLY96NPgi1j3CgE6etTs678udBz15q0TTfkXjMHXzH2mCp06tJcy/ldD17f34UjouT18bpd9aHwKh51JEHSSoZ1E45wzhFmxRsKo=austin@aPostrofree:~$ export AWS_ACCESS_KEY_ID=ASIA34AMCTAE2TGLZ53N
```

```
austin@aPostrofree:~$ export AWS_SECRET_ACCESS_KEY=0C80rbkKn3v9LMxrc5isMupKhrYBpWyydMyeh
```

```
austin@aPostrofree:~$ export AWS_SESSION_TOKEN=IqoJb3jpZ2luX2VjEB4aCXvzLxd1c3QtMSJIMELYCIQDhpqCd39zRbjw0hSlazo2r08p1X7MarH1KaZCV/FMoogIhAK+DWZSfpw300uQXMeN93+Juxc0XC2GBRCfRQmWid4HnkPycCcQaB0MDE2MDY5MTyMDU3IgZ7ywDghuZH7y6Aq1kq8wE9bxvKbJjuMnfSb0lE8YTCjWrkf0o1lmtjQ+x45/zWNBoqqkYeJTr1a:P8qqaP4Eo6eE51gBa4q0zJ517mhug5074bzJ27NuxE/cPfKRT6wSAkF9xLx8LG3411srMqF217+yv1+Faa50Dspz8Mb5kyq16X/8iumXjQWeppmgkAl3x0HEVpJeE74mcAnLm12yXjqvMP51Qm08ubR6evPi13UkLiqzKMC78AuTxmW96IzqayBQz1Xt+e16m3eqpbfbjNzNG2y5Ap+y119yvX8pxqjxL0ufefJgBXGubbApp6P13tYPAQNF+r2Nx+Qz0Aw9yWzAVy6nAGQF5j/Uwcq9zLY96NPgi1j3CgE6etTs678udBz15q0TTfkXjMHXzH2mCp06tJcy/ldD17f34UjouT18bpd9aHwKh51JEHSSoZ1E45wzhFmxRsKo=austin@aPostrofree:~$ aws sts get-caller-identity
```

```
816069122057 arn:aws:sts::816069122057:assumed-role/AuditFinDataExtID/auditdemo AROA34AMCTAEVTSFALHB5:auditdemo
```

```
austin@aPostrofree:~$ aws s3 ls
```

2024-12-27 09:53:44 detectionlabimagesimport

2025-01-13 17:07:35 financial-data-secure-company325

```
austin@aPostrofree:~$
```

Here we see that the role has been assumed using `aws sts get-caller-identity` and we see that the auditing team is now able to perform `aws s3 ls` and list the buckets. Read only access to S3 has been provided.

```
austin@aPostrofree:~
```

```
austin@aPostrofree:~$ aws sts assume-role --role-arm arn:aws:iam::816069122057:role/AuditFinDataExtID --role-session-name auditdemo --external-id Access2225
```

```
ASSUMEDROLEUSER arn:aws:sts::816069122057:assumed-role/AuditFinDataExtID/auditdemo AROA34AMCTAEVTSFALHB5:auditdemo
```

```
CREDENTIALS ASIA34AMCTAE2GLZ53N 2025-01-14T14:46:51+00:00 0C80rbkKn3v9LMxrc5isMupKhrYBpWyydMyeh IQoJb3jpZ2luX2VjEB4aCXvzLxd1c3QtMSJIMELYCIQDhpqCd39zRbjw0hSlazo2r08p1X7MarH1KaZCV/FMoogIhAK+DWZSfpw300uQXMeN93+Juxc0XC2GBRCfRQmWid4HnkPycCcQaB0MDE2MDY5MTyMDU3IgZ7ywDghuZH7y6Aq1kq8wE9bxvKbJjuMnfSb0lE8YTCjWrkf0o1lmtjQ+x45/zWNBoqqkYeJTr1a:P8qqaP4Eo6eE51gBa4q0zJ517mhug5074bzJ27NuxE/cPfKRT6wSAkF9xLx8LG3411srMqF217+yv1+Faa50Dspz8Mb5kyq16X/8iumXjQWeppmgkAl3x0HEVpJeE74mcAnLm12yXjqvMP51Qm08ubR6evPi13UkLiqzKMC78AuTxmW96IzqayBQz1Xt+e16m3eqpbfbjNzNG2y5Ap+y119yvX8pxqjxL0ufefJgBXGubbApp6P13tYPAQNF+r2Nx+Qz0Aw9yWzAVy6nAGQF5j/Uwcq9zLY96NPgi1j3CgE6etTs678udBz15q0TTfkXjMHXzH2mCp06tJcy/ldD17f34UjouT18bpd9aHwKh51JEHSSoZ1E45wzhFmxRsKo=austin@aPostrofree:~$ export AWS_ACCESS_KEY_ID=ASIA34AMCTAE2TGLZ53N
```

```
austin@aPostrofree:~$ export AWS_SECRET_ACCESS_KEY=0C80rbkKn3v9LMxrc5isMupKhrYBpWyydMyeh
```

```
austin@aPostrofree:~$ export AWS_SESSION_TOKEN=IqoJb3jpZ2luX2VjEB4aCXvzLxd1c3QtMSJIMELYCIQDhpqCd39zRbjw0hSlazo2r08p1X7MarH1KaZCV/FMoogIhAK+DWZSfpw300uQXMeN93+Juxc0XC2GBRCfRQmWid4HnkPycCcQaB0MDE2MDY5MTyMDU3IgZ7ywDghuZH7y6Aq1kq8wE9bxvKbJjuMnfSb0lE8YTCjWrkf0o1lmtjQ+x45/zWNBoqqkYeJTr1a:P8qqaP4Eo6eE51gBa4q0zJ517mhug5074bzJ27NuxE/cPfKRT6wSAkF9xLx8LG3411srMqF217+yv1+Faa50Dspz8Mb5kyq16X/8iumXjQWeppmgkAl3x0HEVpJeE74mcAnLm12yXjqvMP51Qm08ubR6evPi13UkLiqzKMC78AuTxmW96IzqayBQz1Xt+e16m3eqpbfbjNzNG2y5Ap+y119yvX8pxqjxL0ufefJgBXGubbApp6P13tYPAQNF+r2Nx+Qz0Aw9yWzAVy6nAGQF5j/Uwcq9zLY96NPgi1j3CgE6etTs678udBz15q0TTfkXjMHXzH2mCp06tJcy/ldD17f34UjouT18bpd9aHwKh51JEHSSoZ1E45wzhFmxRsKo=austin@aPostrofree:~$ aws sts get-caller-identity
```

```
816069122057 arn:aws:sts::816069122057:assumed-role/AuditFinDataExtID/auditdemo AROA34AMCTAEVTSFALHB5:auditdemo
```

```
austin@aPostrofree:~$ aws s3 ls
```

2024-12-27 09:53:44 detectionlabimagesimport

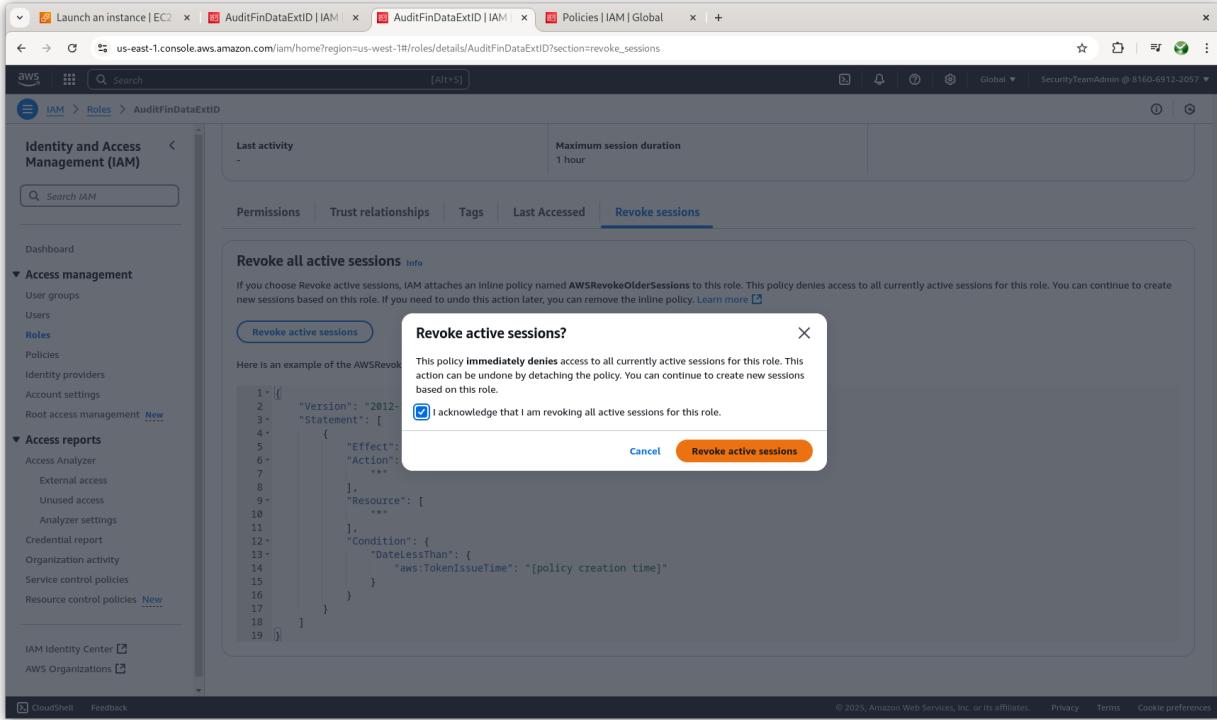
2025-01-13 17:07:35 financial-data-secure-company325

```
austin@aPostrofree:~$ aws s3 rb s3://financial-data-secure-company325
```

remove\_bucket failed: s3://financial-data-secure-company325 An error occurred (AccessDenied) when calling the DeleteBucket operation: User: arn:aws:sts::816069122057:assumed-role/AuditFinDataExtID/auditdemo is not authorized to perform: s3:DeleteBucket on resource: "arn:aws:s3:::financial-data-secure-company325" because no identity-based policy allows the s3:DeleteBucket action

```
austin@aPostrofree:~$
```

Above we show that the auditing team role is unable to delete S3 resources as `aws s3 rb s3://financial-data-secure-company325` returns a not authorized to perform error.



Next we revoke all active sessions under IAM > Access Management > Roles this immediately revokes the access tokens, useful for instance when credentials have been leaked or stolen.

The screenshot shows the AWS IAM Roles page. The role 'AuditFinDataExtID' is selected. The 'Permissions' tab is active, displaying two managed policies: 'AWSRevokeOlderSessions' (Customer inline) and 'S3ListAndRead' (Customer managed). The ARN of the role is listed as arn:aws:iam:816069122057:role/AuditFinDataExtID.

The screenshot shows the 'Edit policy' page for the 'AWSRevokeOlderSessions' policy. The 'Visual' tab is selected. The policy document is displayed in JSON format:

```

1  {
2   "Version": "2012-10-17",
3   "Statement": [
4     {
5       "Effect": "Deny",
6       "Action": [
7         "*"
8       ],
9       "Resource": [
10        "*"
11     ],
12     "Condition": {
13       "DateLessThan": {
14         "aws:TokenIssueTime": "2025-01-14T14:35:58.582Z"
15       }
16     }
17   ]
18 }
19

```

The sidebar shows the policy editor interface with tabs for Visual, JSON, Actions, and a button to Add new statement. A note at the bottom indicates 10075 of 10240 characters remaining.

This creates the permission policy shown.

A

```
austin@Apostrofree:~ austin@Apostrofree:~  
zRBjw0hSlazo2r08p1X7MarH1KaZCV/FMoogIhAK+DWZSfwp300UQXMeN93+Juxc0XC2GBRCfR0mWd4HnKpYCCBcOaboMODE2MDY5MTIyMDU3Iqz7ywDghUZH7Y6A0ikq8wE9bxvKEbjjuMn6Sb0LE8YTCjWirkfo0ljmjt0+x  
45+zWNBoqkyejTria-PBqaP4Eo6eES1w9g4g0zjv517mhug074bj27NuxE/cPFkRT6wSAKF9x18Lg34llsMrQF2l7+yv1+a450Dspz8Mb5kyyq16x/8iumxj0WepmgkAl3x0HEvPje74mcanaLm1zYxjvMP5l0M08ubR  
6evQPi3UKLiqzKMC78AuTxnw96IzqayNBQz1xt4-e1m36qprfbjNzRNGY5APya119vYx8FQjxLU0vfeJgbxGubbApbPr3tYPAQNF+f2Nx4Qz2tOAwy9WzvAY6nA6gtQF5j/Uicq9zLY96Npgij3cgE6etSTsu678udBz1s  
q0TTfkjxMHXHZmWCPO6tJcy/lld17f34Uiu0t718b0pp9aHKhS1JEHSSo1zE45wzHmxskoth/YL27TP12xxCG/kIE0myeBAVbyblyt2a0r7lq21TB8dB3csx0IPvUs1zqTvovxpvdvF8z2huNas/E1zSH/S0dmSkqc=  
austin@Apostrofree:~ export AWS_ACCESS_KEY_ID=ASIA34AMCTAE2zTGL53N  
austin@Apostrofree:~ export AWS_SECRET_ACCESS_KEY=C080rbhkkn3v9LMXrc5isUpkHRYBPWYdIMyeh  
austin@Apostrofree:~ export AWS_SESSION_TOKEN=IqoJb3jp22lu2VjEcMaCXvzLxd1c3QtMSJHEUCIE1HE0szU  
austin@Apostrofree:~ export AWS_SESSION_TOKEN=IqoJb3jp22lu2VjEcMaCXvzLxd1c3QtMSJHEUCIE1HE0szU  
RQmwid4HnkpYCCbQAbMODE2MDY5MTIyMDU3Iqz7ywDghUZH7Y6A0ikq8wE9bxvKEbjjuMn6Sb0LE8YTCjWirkfo0ljmjt0+x  
zWNBoqkyejTria-PBqaP4Eo6eES1w9g4g0zjv517mhug074bj27NuxE/cPFkRT6wSAKF9  
x1xLG34t1sRmQf217+yv1+Fa45QDspz8Mb5kyyq16x/8iumxj0WepmgkAl3x0HEvPje74mcanaLm1zYxjvMP5l0M08ubR  
FoxjLxU0vfeJgbxGubbApbPr3tYPAQNF+f2Nx4Qz2tOAwy9WzvAY6nA6gtQF5j/Uicq9zLY96Npgij3cgE6etSTsu678udBz1s  
q0TTfkjxMHXHZmWCPO6tJcy/lld17f34Uiu0t718b0pp9aHKhS1JEHSSo1zE45wzHmxskoth/YL27TP12xxCG/kIE0myeBAVbyblyt2a0r7lq21TB8dB3csx0IPvUs1zqTvovxpvdvF8z2huNas/E1zSH/S0dmSkqc=  
austin@Apostrofree:~ $ aws sts get-caller-identity  
816069122057 arn:aws:sts::816069122057:assumed-role/AuditFinDataExtID/auditdemo AROA34AMCTAEVSFALH5B:auditdemo  
austin@Apostrofree:~ $ aws s3 ls  
2024-12-27 09:53:44 detectionlabimagesimport  
2025-01-13 17:07:35 financial-data-secure-company325  
austin@Apostrofree:~ $ aws s3 rb s3://financial-data-secure-company325  
remove_bucket failed: s3://financial-data-secure-company325 An error occurred (AccessDenied) when calling the DeleteBucket operation: User: arn:aws:sts::816069122057:assumed-role/AuditFinDataExtID/auditdemo is not authorized to perform: s3:DeleteBucket on resource: "arn:aws:s3:::financial-data-secure-company325" because no identity-based policy allows the s3:DeleteBucket action  
austin@Apostrofree:~ $ aws s3 ls  
An error occurred (AccessDenied) when calling the ListBuckets operation: User: arn:aws:sts::816069122057:assumed-role/AuditFinDataExtID/auditdemo is not authorized to perform: s3:ListAllMyBuckets with an explicit deny in an identity-based policy  
austin@Apostrofree:~ $
```

As seen above the Auditing Team using the credentials that were revoked is no longer able to list or read S3 resources!

```
austin@Apostrofree:~ austin@Apostrofree:~  
austin@Apostrofree:~ $ aws sts assume-role --role-arn arn:aws:iam::816069122057:role/AuditFinDataExtID --role-session-name auditnewest2 --external-id Access2225  
ASSUMEDROLEUSER arn:aws:sts::816069122057:assumed-role/AuditFinDataExtID/auditnewest2 AROA34AMCTAEVSFALH5B:auditnewest2  
CREDENTIALS ASIA34AMCTAE60PDC0  
2025-01-14T19:27:21+00:00 JzzY0315f18brYElzxHq312dmle2D22n7s/Ke+uf IQoJb3jp22lu2VjEcMaCXvzLxd1c3QtMSJHEUCIE1HE0szU  
EP12Yu9gbxsYiml67KkomA9Kwv702+A1EA14lLAkep6AT1/RLoM/c/Wp/Jhyhsu6da/iyI+RLZBBMggmQI1HbAAggw4MTYwNjkMjIwNTciDAMVEmZ0R2dAnfxs1zAfWz8uuu+PtqmSE/F3Gscfm0WxQwgEP/EOajKf3t  
QxaXnxJCFW75m+7Ww067QmkBF00LC8q3Xv2BmK/Gk5aY8EcaxMKTMOUFvCyeV3H3zX/f0Xod42J0Ine4vrgtE14AXGS/21sRK/jko1RteJliqdY1owrnIs4dDnx/dcN8d2emhAxns008phGbX888gszpx9zAiEn64uqt+  
sturGvYUmt4n158e11LXRTFq55BoxvB3np9w9y/UgUe2X15zrGin09sfjtmaeCpl655NMFGtV4rNkfZqyFMNHx/UQ2k8afWpV8bx7cszAtbmNUyPfR1y4TCj22qBjqdAT1zhVTsNHKEoe9p3A11nN68bnR84qvMRZdHoZx  
Uuk5gwW9nbK0X+wmbujg+80@HgznruGrDaXqzgFvb81c/W4zawvwNCloodwRx19Mcgjx5h8yReAKYU5/tW3c5KEYNsrlkeK2BSXIR+xpykw9xmpnfh45FyMAK9YKJ88x1jigkfa4smz11vN7p0JKB133CIkC/lv/ULq3jp  
M=  
austin@Apostrofree:~ $ export AWS_ACCESS_KEY_ID=ASIA34AMCTAE60PDC0  
austin@Apostrofree:~ $ export AWS_SECRET_ACCESS_KEY=jzzY0315f18brYElzxHq312dmle2D22n7s/Ke+uf  
austin@Apostrofree:~ $ export AWS_SESSION_TOKEN=IqoJb3jp22lu2VjEcMaCXvzLxd1c3QtMSJHEUCIE1HE0s2zUEPi2YugMbXs5Yjmml6G7komA9Kwv702+A1EA14lLAkep6AT1/RLoM/c/Wp/Jhyhsu6da/iyI  
+RLZBBMggmQI1HbAAggw4MTYwNjkj1wNt1dAMYeEMZ0R2dAnfxs1zAfWz8uuu+PtqmSE/F3Gscfm0WxQwgEP/EOaJKF3tQxaXnxJCFW75m+7Ww067QmkBF00LC8q3Xv2BmK/Gk5aY8EcaxMKTMOUFvCyeV3H3zX/f0Xo  
d4210lNe4vrgtE14AXGS/21sRK/jko1RteJliqdY1owrnIs4dDnx/dcN8d2emhAxns008phGbX888gszpx9zAiEn64uqt+sturGvYUmt4n158e11LXRTFq55BoxvB3np9w9y/UgUe2X15zrGin09sfjtmaeCpl655NMFGtV4  
41NKFZqy8FMNHx/UQ2k8afWpV8bx7csaAtmBuYpFfR1y4TCj22qBjqdAT1zhVTsNHKEoe9p3A11nN68bnR84qvMRZdHoZx  
ReAKYU5/tW3c5KEYNsrlkeK2BSXIR+xpykw9xmpnfh45FyMAK9YKJ88x1jigkfa4smz11vN7p0JKB133CIkC/lv/ULq3jp  
austin@Apostrofree:~ $ aws sts get-caller-identity  
816069122057 arn:aws:sts::816069122057:assumed-role/AuditFinDataExtID/auditnewest2 AROA34AMCTAEVSFALH5B:auditnewest2  
austin@Apostrofree:~ $ aws s3 ls  
2024-12-27 09:53:44 detectionlabimagesimport  
2025-01-13 17:07:35 financial-data-secure-company325  
austin@Apostrofree:~ $ aws s3 rb s3://financial-data-secure-company325  
remove_bucket failed: s3://financial-data-secure-company325 An error occurred (AccessDenied) when calling the DeleteBucket operation: User: arn:aws:sts::816069122057:assumed-role/AuditFinDataExtID/auditnewest2 is not authorized to perform: s3:DeleteBucket on resource: "arn:aws:s3:::financial-data-secure-company325" because no identity-based policy allows the s3:DeleteBucket action  
austin@Apostrofree:~ $
```

However the role can be assumed again now with the new access tokens and the same permissions provided to the role are in place once again, however the leaked or stolen credentials remain invalid.

```

austin@Apostrofree:~$ aws configure
AWS Access Key ID [*****HKN6]: AKIA34AMCTAERWKUIV60
AWS Secret Access Key [*****dNcC]: XLERZy6NT5d8n7F5lbCX652APch55fB2MTefpULM
Default region name [us-west-1]:
Default output format [text]:
austin@Apostrofree:~$ aws sts get-caller-identity
arn:aws:iam::816069122057:user/SecurityTeamAdmin
austin@Apostrofree:~$ aws iam create-user --user-name James
USER arn:aws:iam::816069122057:user/James 2025-01-15T12:56:17+00:00 / AIDA34AMCTAEU5DQ07GYQ James
austin@Apostrofree:~$ 

```

Next we switch back to the SecurityTeamAdmin IAM user and create a new user James with `aws iam create-user --user-name James` to show the permissions of SecurityTeamAdmin are assumed.

The screenshot shows the 'Create policy' wizard in the AWS IAM console. The current step is 'Specify permissions'. The JSON editor displays the following policy statement:

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": "iam:DeleteUserPermissionsBoundary",
      "Resource": "arn:aws:iam::816069122057:policy/userboundary"
    }
  ]
}

```

The right side of the screen shows the 'Actions' tab of the policy editor, where the 'Edit statement' button is highlighted. The 'Available' section lists services like AI Operations, AMP, API Gateway, and others. The 'Included' section shows 'IAM'.

Next we create a new policy with a permission boundary, a permission boundary explicitly denies an action overriding other more privileged permissions in a policy. This permission boundary is applied to our new administrator James and prevents an admin from changing their own permissions, or other admins permissions, or from creating new admins with more privileges.

The screenshot shows the AWS IAM 'Create policy' wizard at Step 2: Review and create. The policy name is set to 'IAMPermissionBoundaryLimitAdmin'. The description field contains: 'A permission boundary policy to prevent a newly created admin from changing their own permissions or from creating new admins with more permissive privileges'. In the 'Permissions defined in this policy' section, there is one explicit deny rule for the IAM service:

Service	Access level	Resource	Request condition
IAM	Limited: Permissions management	Multiple	None

We name the permission boundary **IAMP*ermission*BoundaryLimitAdmin** and save the permission

**Add permissions**

Add user to an existing group or create a new one. Using groups is a best-practice way to manage user's permissions by job functions. [Learn more](#)

**Permissions options**

- Add user to group
- Copy permissions
- Attach policies directly

**Permissions policies (1/1321)**

Policy name	Type	Attached entities
IAMPermissionBoundaryLimitAdmin	Customer managed	0
IAMReadPolicy	Customer managed	1
S3ListAndRead	Customer managed	1

**Filter by Type:** Customer managed | 3 matches

**Next**

**James | IAM | Global**

**James**

**Identity and Access Management (IAM)**

**Summary**

ARN: arn:aws:iam::816069122057:user/James

Created: January 15, 2025, 07:56 (UTC-05:00)

**Console sign-in**

Console sign-in link: https://816069122057.signin.aws.amazon.com

**Multi-factor authentication**

No MFA devices assigned.

**Access keys (0)**

**Enable console access**

Enable console access for James.

**Console password**

Autogenerated password  
 Custom password  
 Password

**Update login details?**

Title: 816069122057.signin.aws.amazon.com

Email or Username: James

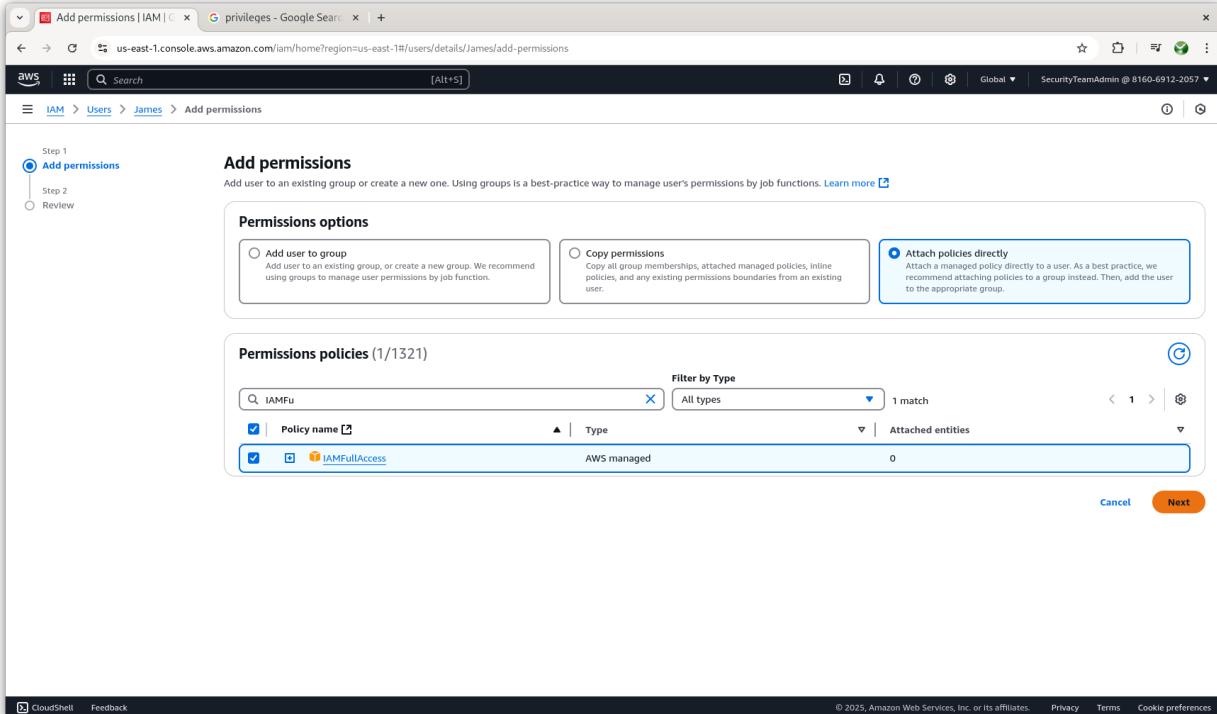
Password:

Show More

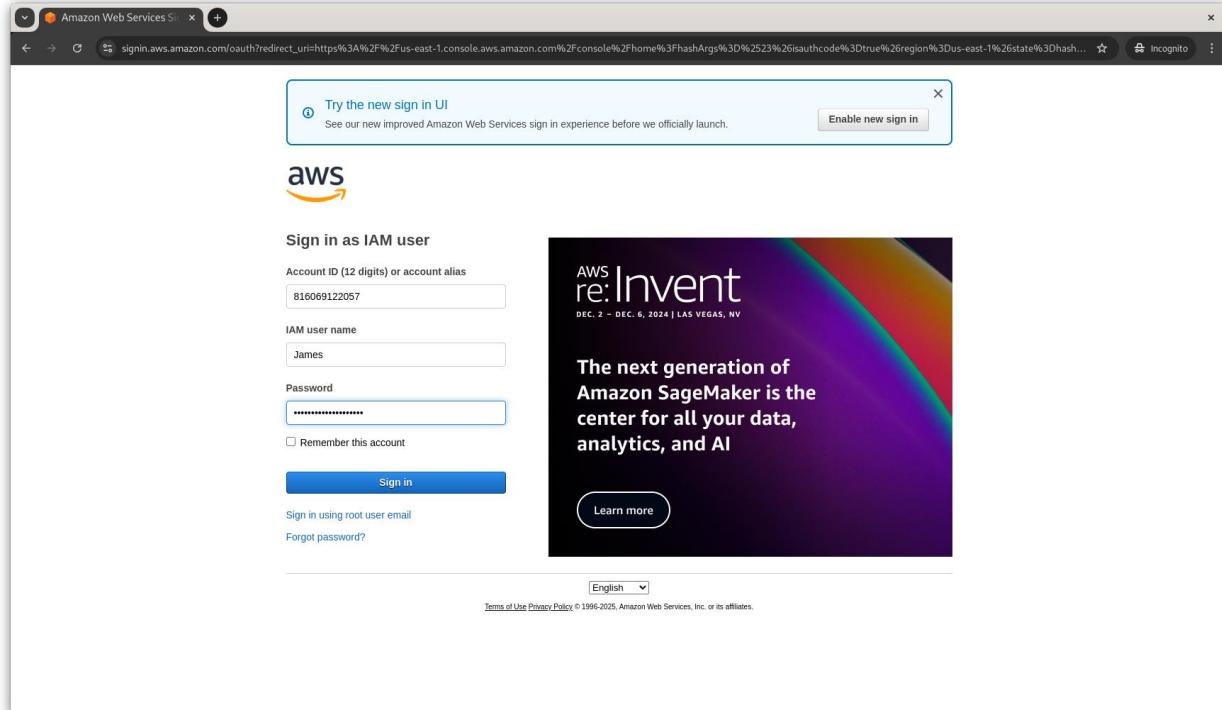
**Update**

**Save as New**

Next we attach the policy to James and enable console access for James our new admin.



The admin James now has the permission policy IAMFullAccess as well as the permission boundary we created to limit admin permissions.



We log into the console as James.

A screenshot of the AWS IAM "Add permissions" page. The URL is us-east-1.console.aws.amazon.com/iam/home?region=us-east-1#users/details/Matt/add-permissions. The page shows a navigation path: IAM &gt; Users &gt; Matt &gt; Add permissions. It's Step 1 of 2, titled "Add permissions". The "Permissions options" section contains three radio button choices: "Add user to group" (disabled), "Copy permissions" (disabled), and "Attach policies directly" (selected). The "Permissions policies (1321)" section lists 1321 policies, filtered by type (All types). The table includes columns for Policy name, Type, and Attached entities. Policies listed include "AccessAnalyzerServiceRolePolicy", "AdministratorAccess", "AdministratorAccess-Amplify", "AdministratorAccess-AWSElasticBeanstalk", "AIOpsAssistantPolicy", "AIOpsConsoleAdminPolicy", "AIOpsOperatorAccess", "AIOpsReadOnlyAccess", and "AlexaForBusinessDeviceSetup". At the bottom of the page are links for "CloudShell", "Feedback", "Privacy", "Terms", and "Cookie preferences".

James is unable to modify his own permissions or those of other administrators.

Create policy | IAM | Global

us-east-1.console.aws.amazon.com/iam/home?region=us-east-1#/policies/create

IAM > Policies > Create policy

Step 2 Review and create

**Policy editor**

```

1  {
2   "Version": "2012-10-17",
3   "Statement": [
4     {
5       "Sid": "Statement1",
6       "Effect": "Deny",
7       "Action": "s3:CreateBucket",
8       "Resource": "*"
9     }
10   ]
11 }
```

+ Add new statement

JSON Ln 7, Col 19

6027 of 6144 characters remaining

Edit statement Statement1 Remove

Add actions Choose a service Filter services

Included S3

Available AI Operations AMP API Gateway API Gateway V2 ARC Zonal Shift ASC Access Analyzer

Add a resource Add

Add a condition (optional) Add

CloudShell Feedback © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Add permissions | IAM | Global

us-east-1.console.aws.amazon.com/iam/home?region=us-east-1#/users/details/Matt/add-permissions

IAM > Users > Matt > Add permissions

Step 1 Add permissions Step 2 Review

### Add permissions

Add user to an existing group or create a new one. Using groups is a best-practice way to manage user's permissions by job functions. [Learn more](#)

#### Permissions options

- Add user to group Add user to an existing group, or create a new group. We recommend using groups to manage user permissions by job function.
- Copy permissions Copy all group memberships, attached managed policies, inline policies, and any existing permissions boundaries from an existing user.
- Attach policies directly Attach a managed policy directly to a user. As a best practice, we recommend attaching policies to a group instead. Then, add the user to the appropriate group.

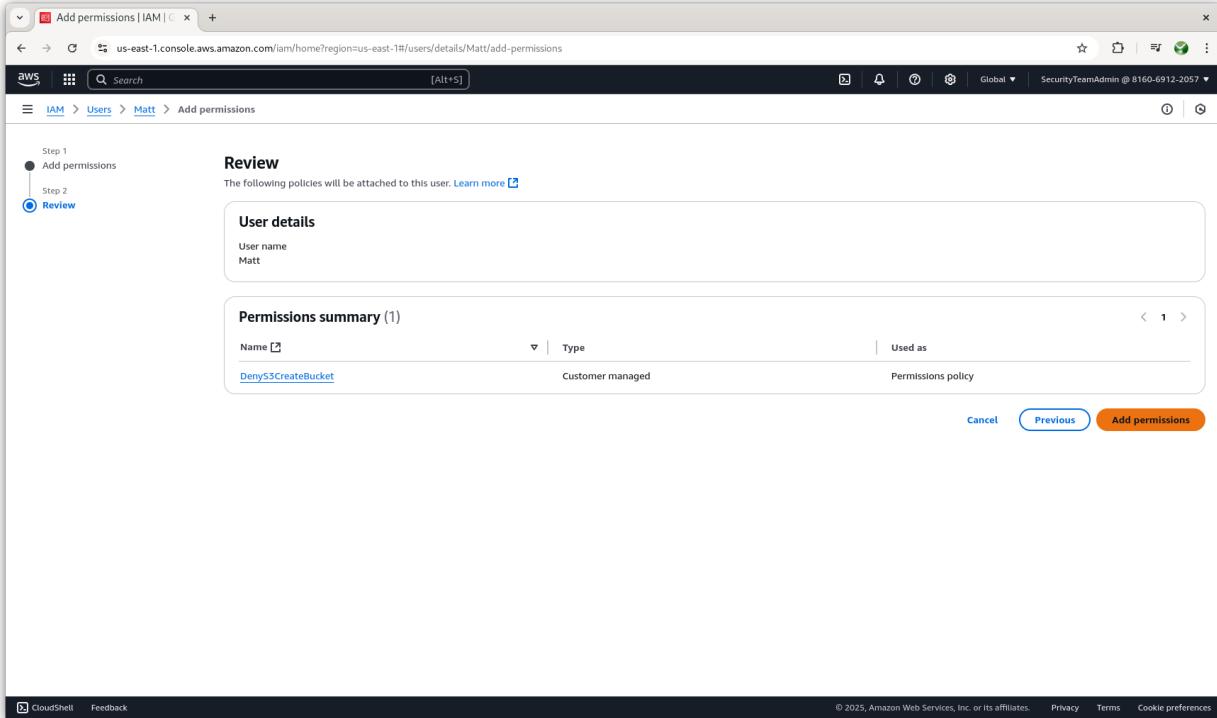
#### Permissions policies (1/1322)

Filter by Type Customer managed 4 matches

Policy name	Type	Attached entities
<input checked="" type="checkbox"/> DenyS3CreateBucket	Customer managed	0
<input type="checkbox"/> IAMPermissionBoundaryLimitAdmin	Customer managed	0
<input type="checkbox"/> IAMReadPolicy	Customer managed	1
<input type="checkbox"/> S3ListAndRead	Customer managed	1

Cancel Next

CloudShell Feedback © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences



Now we attach a permission policy denying the creation of S3 buckets and attach it to the user Matt.

Matt | IAM | Global    IAM policy testing with t...    IAM Policy Simulator

policsim.aws.amazon.com/home/index.jsp?#users/Matt

IAM Policy Simulator

Policies    Back    Create New Policy

Selected user: Matt

IAM Policies

Filter

- DenyS3CreateBucket
- AmazonS3FullAccess
- IAMReadPolicy

Custom IAM Policies

There are no policies to display!

Permissions Boundary Policy

You can simulate a maximum of one permissions boundary policy per user or role.

There are no policies to display!

Custom IAM Permissions Boundary Policy

There are no policies to display!

Resource Policies

The IAM policy simulator is a testing environment that may not model all of the factors present in your production environment, so we cannot guarantee the accuracy of the results. Authorization results in your production environment may differ from what's shown in the IAM policy simulator therefore we recommend testing your policies with production IAM users and AWS requests.

© 2013, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Mode : Existing Policies    SecurityTeamAdmin

Policy Simulator

Amazon S3    1 Action(s) sele...    Select All    Deselect All    Reset Contexts    Clear Results    Run Simulation

Global Settings

Action Settings and Results [1 actions selected. 0 actions not simulated. 0 actions allowed. 1 actions denied.]

Service	Action	Resource Type	Simulation Resource	Permission
Amazon S3	CreateBucket	bucket	*	<span style="color: red;">denied</span> 1 matching statements.

Matt | IAM | Global    IAM policy testing with t...    IAM Policy Simulator

policsim.aws.amazon.com/home/index.jsp?#users/Matt

IAM Policy Simulator

Policies    Back    Create New Policy

Selected user: Matt

IAM Policies

Filter

- DenyS3CreateBucket
- AmazonS3FullAccess
- IAMReadPolicy

Custom IAM Policies

There are no policies to display!

Permissions Boundary Policy

You can simulate a maximum of one permissions boundary policy per user or role.

There are no policies to display!

Custom IAM Permissions Boundary Policy

There are no policies to display!

Resource Policies

The IAM policy simulator is a testing environment that may not model all of the factors present in your production environment, so we cannot guarantee the accuracy of the results. Authorization results in your production environment may differ from what's shown in the IAM policy simulator therefore we recommend testing your policies with production IAM users and AWS requests.

© 2013, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Mode : Existing Policies    SecurityTeamAdmin

Policy Simulator

Amazon S3    3 Action(s) sele...    Select All    Deselect All    Reset Contexts    Clear Results    Run Simulation

Global Settings

Action Settings and Results [3 actions selected. 0 actions not simulated. 2 actions allowed. 1 actions denied.]

Service	Action	Resource Type	Simulation Resource	Permission
Amazon S3	CreateBucket	bucket	*	<span style="color: red;">denied</span> 1 matching statements.
Amazon S3	DeleteObject	object	*	<span style="color: green;">allowed</span> 1 matching statements.
Amazon S3	GetObject	object	*	<span style="color: green;">allowed</span> 1 matching statements.

**Policies**

Selected user: Matt

**IAM Policies**

- DenyS3CreateBucket
- AmazonS3FullAccess
- IAMReadPolicy

**Custom IAM Policies**

There are no policies to display!

**Permissions Boundary Policy**

You can simulate a maximum of one permissions boundary policy per user or role.

There are no policies to display!

**Custom IAM Permissions Boundary Policy**

There are no policies to display!

**Resource Policies**

The IAM policy simulator is a testing environment that may not model all of the factors present in your production environment, so we cannot guarantee the accuracy of the results. Authorization results in your production environment may differ from what's shown in the IAM policy simulator therefore we recommend testing your policies with production IAM users and AWS requests.

© 2013, Amazon Web Services, Inc. or its affiliates. All rights reserved.

**Policies**

Selected user: Matt

**IAM Policies**

- DenyS3CreateBucket
- AmazonS3FullAccess
- IAMReadPolicy

**Custom IAM Policies**

There are no policies to display!

**Permissions Boundary Policy**

You can simulate a maximum of one permissions boundary policy per user or role.

There are no policies to display!

**Custom IAM Permissions Boundary Policy**

There are no policies to display!

**Resource Policies**

The IAM policy simulator is a testing environment that may not model all of the factors present in your production environment, so we cannot guarantee the accuracy of the results. Authorization results in your production environment may differ from what's shown in the IAM policy simulator therefore we recommend testing your policies with production IAM users and AWS requests.

© 2013, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Using the policy simulator we can troubleshoot IAM policies and see that:

In IAM the Deny action overrides the Allow action and if you do not have explicit permission to access AWS resources or perform an action, you are automatically denied permission to do so.

The screenshot shows the 'Add permissions' step of the IAM role creation wizard. On the left, a vertical navigation bar lists three steps: 'Select trusted entity', 'Add permissions' (which is selected), and 'Name, review, and create'. The main area is titled 'Add permissions' with a sub-section 'Permissions policies (1/1024)'. It displays a search bar with 's3full' and a dropdown filter set to 'All types'. A single policy, 'AmazonS3FullAccess', is listed, selected by a checked checkbox. The policy is described as 'AWS managed' and provides 'full access to all buckets via the ...'. Below this is a section titled 'Set permissions boundary - optional', which is currently empty. At the bottom right are 'Cancel', 'Previous', and 'Next' buttons.

The screenshot shows the 'Name, review, and create' step of the IAM role creation wizard. The left sidebar shows the completed steps: 'Select trusted entity', 'Add permissions', and 'Name, review, and create' (selected). The main area is titled 'Name, review, and create' and contains a 'Role details' section. Under 'Role name', the value 'EC2FullAccessS3Role' is entered. Under 'Description', the text 'Allows EC2 Instances to call AWS services on your behalf.' is provided. Below this is a code editor for the 'Trust policy' with the following JSON content:

```
1-{| "Version": "2012-10-17",
2-|   "Statement": [
3-|     {
4-|       "Effect": "Allow",
5-|       "Action": [
6-|         "sts:AssumeRole"
7-|       ],
8-|       "Principal": {
9-|         "Service": [
10-|           "ec2.amazonaws.com"
11-|         ]
12-|       }
13-|     }
14-|   ]
15-| }
```

Next, we create an IAM role to grant full S3 access to an EC2 virtual machine.

**Launch an instance | EC2**

us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#LaunchInstances:

**Name and tags** [Info](#)

Name:  Add additional tags

**Application and OS Images (Amazon Machine Image)** [Info](#)

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below.

Search our full catalog including 1000s of application and OS images

**Quick Start**

Amazon Linux macOS Ubuntu Windows Red Hat SUSE Linux Debian

[Browse more AMIs](#) Including AMIs from AWS, Marketplace and the Community

**Amazon Machine Image (AMI)**

Amazon Linux 2023 AMI ami-09115b7fbfc5c64 (64-bit (x86), uefi-preferred) / ami-0d152f93a71fbdb6 (64-bit (Arm), uefi) Free tier eligible

Virtualization: hvm ENA enabled: true Root device type: ebs

**Description**

Amazon Linux 2023 is a modern, general purpose Linux-based OS that comes with 5 years of long term support. It is optimized for AWS and designed to provide a secure, stable and high-performance execution environment to develop and run your cloud applications.

Amazon Linux 2023 AMI 2023.6.20250114.0-x86\_64-AMZ-Kernel-6.1

**Summary**

Number of instances: 1

Software Image (AMI) Amazon Linux 2023 AMI 2023.6.2... [read more](#)

Virtual server type (instance type) t2.micro

Firewall (security group) New security group

Storage (volumes) 1 volume(s) - 8 GiB

**Free tier:** In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) Instance usage on free tier AMIs per month, 750 hours of public IPv4 address usage per month, 30 GiB of EBS storage, 2 million I/Os, 1 GB of snapshots, and 100 GiB of bandwidth to the internet.

Cancel [Launch instance](#) [Preview code](#)

**Launch an instance | EC2**

us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#LaunchInstances:

**Key pair name** [- required](#)

Select

**Network settings** [Info](#)

Network:  Subnet: [Info](#) No preference (Default subnet in any availability zone)

Auto-assign public IP:  Enable Additional charges apply when outside of free tier allowance

**Firewall (security groups)** [Info](#)

A security group is a set of firewall rules that control the traffic for your instance. Add rules.

Create security group  Select existing security group

We'll create a new security group called 'launch-wizard-1' with the following rules:

- Allow SSH traffic from  Anywhere
- Allow HTTPS traffic from the internet To set up an endpoint, for example when creating a web server
- Allow HTTP traffic from the internet To set up an endpoint, for example when creating a web server
- Rules with source of 0.0.0.0/ allow all IP addresses to access your instance known IP addresses only.

**Select an existing key pair or create a key pair**

We noticed that you didn't select a key pair. If you want to be able to connect to your instance it is recommended that you create one or select an existing one.

Create new key pair  Proceed without key pair

**Key pair name**  The name can include up to 255 ASCII characters. It can't include leading or trailing spaces.

**Key pair type**

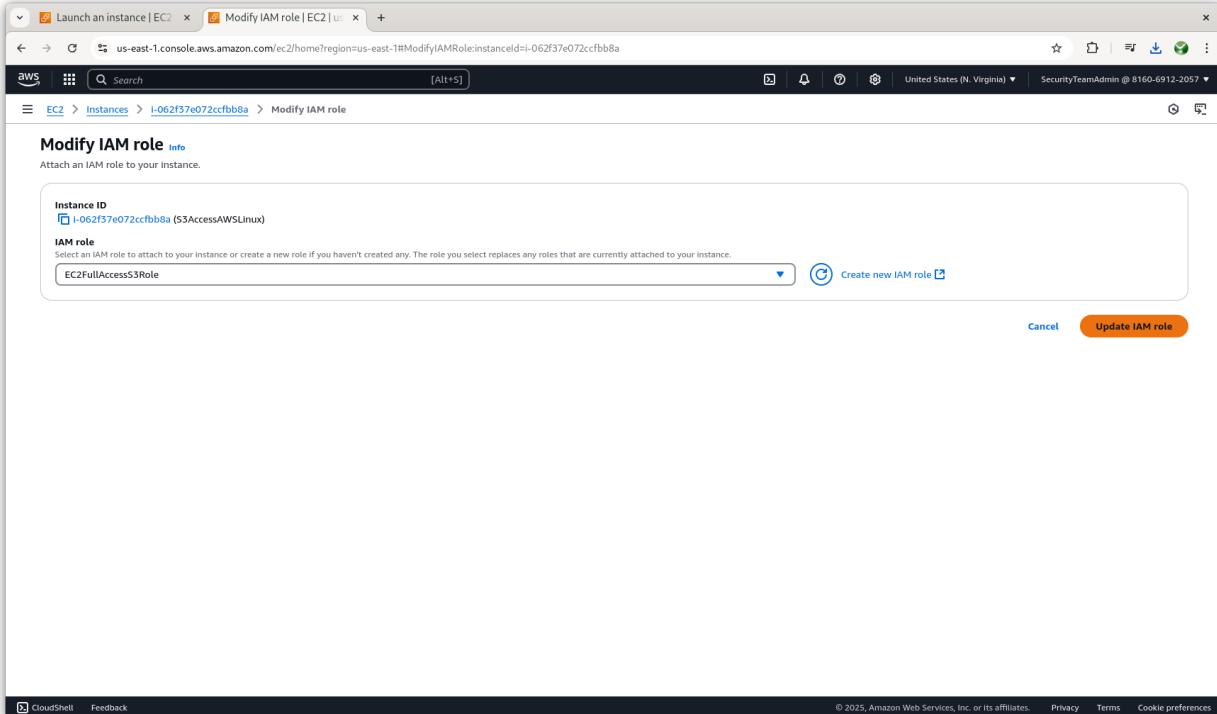
- RSA RSA encrypted private and public key pair
- ED25519 ED25519 encrypted private and public key pair

**Private key file format**

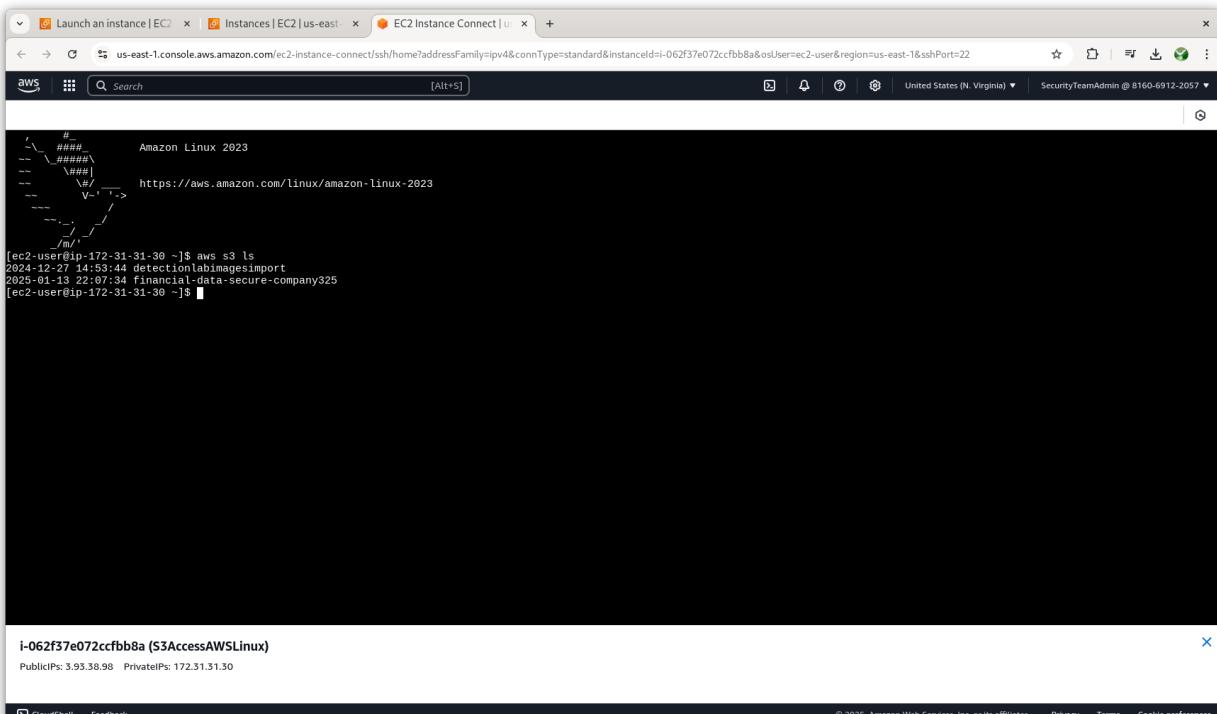
- .pem For use with OpenSSH
- .ppk For use with PuTTY

When prompted, store the private key in a secure and accessible location on your computer. You will need it later to connect to your instance. [Learn more](#)

Cancel [Launch instance](#) [Preview code](#)



We launch the instance and update its role to the newly created EC2FullAccessS3Role.



Here we see from the shell of the instance we can access S3 resources.

```

austin@austin-Apostrofree:~ 
austin@austin-Apostrofree:~ +cYqmWuR57D2P/zcIOW+u42547Afwn9Tz4E09fQMANJb/ZPe2pMUtIfy7/bh7+udRC6/Yoh1ZVRmn1T1/Gac+RrS2xh/RTX0tjfkwuGuF0+Y+E8QwfBVkeWwCwLfsWipyftG8QWwN4IabNt35piKWhCapa8PGvmtCuJ8+vE3+ /33avCY83wL1Xzqbx1ybxullRSUxTP+tyf2t21qJaTbf/XlDytrcIIDIGCX4Smrifwmmazp+M86ytq02axpD+xScg5x0acCAENVNSbeSVqs/EoMfKxtwVM/16Wd2j7Tyu219gx0hIEo0lg2zALjYwBjcn4GTEsQcjb8u/+1Fo0YCd036asneBK6vL6jv+NxeicvHpkB0EntcojojBRH1f8/P01/70LoT/JJx5F/ofF0d8cd1VG9VdVJGR1ytJ.EGvTkeIRQuAmT6WzAHGrCMF9xf20COpwQ05eJWK07tvb5M27RRHTN9n+s8jes1DCG82mZ8xwOWhu4PfewPx5hP331h0Y1b3zFSQ50P6rYSPPvs4Rn4l91UmIAx4zv+0/FvktwlbfSTxuF3MsyuF1pUpujSuUca1M26Aw/5ufvAY6sQH7InZ7vSwBpJCTQ1CJb/mUaq50/eJII21Xk20dhmOucRadE9W1kamCRTjZHiB4N hzHpdh6KH1Q21Y5fa18nig910b9az2JULQNKSWQHPriK2eyz1tN+eG2h9UM9DHC/a2f8dhbGHDidb7jwv905W5k9A9Qzmag5GNCAL9UmGkbyNwNMxel81hTFqzZ8wgW6yUM31YRP0ceCmUaOb81A4EMvRk8cQzavcqs+5jnA= austin@austin-Apostrofree:~ $ export AWS_ACCESS_KEY_ID=ASIA34AMCTAEUF47YMU austin@austin-Apostrofree:~ $ export AWS_SECRET_ACCESS_KEY=CeGPohkP//0xM3J0B4edGk4Wxu4s1T+fhyUDkP5 austin@austin-Apostrofree:~ $ export AWS_SESSION_TOKEN=I0qJb3jpz2luX2VjeEdacXVzLWvhc3QtMSJHEMEUCQCTL81Wk352Uun1Msep2ykFR561E1E1cnm+VKucF1FqqjwIgLdDqQ50Bbb0uFxdkJhq7KH/UEFYb1ZVxRc n/P26lv8quwIMBAAGq4MTWnjkxMj1wTc1dC2EK35n210dhngC8qYBwOr1VS01b80Ypdk97500qxv7o718/+tmd5fAU0Mfpxqf6tnHj8/6auFvnLKzKqkrvkuilAMsvqt1CusPeD3M3MpkeVFnC3j5sRxR1GtKgmurUR in12IszshY3dd2tKVU832hcDVokmok33440JkDHfCqE4ANKQXzp0wxwGBe4B11dnuaR9R1Nv3U1VsQ01mkPPTaVemtaebvNTNa9yFm5cXFtR03v3AUUJ51ooyji0impBk/G5c3kRQ/+djmfs+qfcfndE51sibgjx8Y1Nog +cYqmWuR57D2P/zcIOW+u42547Afwn9Tz4E09fQMANJb/ZPe2pMUtIfy7/bh7+udRC6/Yoh1ZVRmn1T1/Gac+RrS2xh/RTX0tjfkwuGuF0+Y+E8QwfBVkeWwCwLfsWipyftG8QWwN4IabNt35piKWhCapa8PGvmtCuJ8+vE3+ /33avCY83wL1Xzqbx1ybxullRSUxTP+tyf2t21qJaTbf/XlDytrcIIDIGCX4Smrifwmmazp+M86ytq02axpD+xScg5x0acCAENVNSbeSVqs/EoMfKxtwVM/16Wd2j7Tyu219gx0hIEo0lg2zALjYwBjcn4GTEsQcjb8u/+1Fo0YCd036asneBK6vL6jv+NxeicvHpkB0EntcojojBRH1f8/P01/70LoT/JJx5F/ofF0d8cd1VG9VdVJGR1ytJ.EGvTkeIRQuAmT6WzAHGrCMF9xf20COpwQ05eJWK07tvb5M27RRHTN9n+s8jes1DCG82mZ8xwOWhu4PfewPx5hP331h0Y1b3zFSQ50P6rYSPPvs4Rn4l91UmIAx4zv+0/FvktwlbfSTxuF3MsyuF1pUpujSuUca1M26Aw/5ufvAY6sQH7InZ7vSwBpJCTQ1CJb/mUaq50/eJII21Xk20dhmOucRadE9W1kamCRTjZHiB4N hzHpdh6KH1Q21Y5fa18nig910b9az2JULQNKSWQHPriK2eyz1tN+eG2h9UM9DHC/a2f8dhbGHDidb7jwv905W5k9A9Qzmag5GNCAL9UmGkbyNwNMxel81hTFqzZ8wgW6yUM31YRP0ceCmUaOb81A4EMvRk8cQzavcqs+5jnA= austin@austin-Apostrofree:~ $ aws sts get-caller-identity 816069122057 ainst:aws:sts:816069122057:assumed-role/EC2FullAccessS3Role/1-062f37e072ccfb8a AROA34AMCTAE356NPGBJH:1-062f37e072ccfb8a austin@austin-Apostrofree:~ $ 

```

We then assume the role on our AWS CLI using the credentials from the EC2 instance including the role's session token. And show that we have access to S3 resources.

Simulating the theft of the credentials from the instance.

The screenshot shows the AWS IAM Role details page for 'EC2FullAccessS3Role'. The role is described as allowing EC2 Instances to call AWS services on behalf of the user. It was created on January 15, 2025, at 09:17 UTC-05:00. The ARN is arn:aws:iam::816069122057:role/EC2FullAccessS3Role. The maximum session duration is 1 hour. An instance profile ARN is also listed: arn:aws:iam::816069122057:instance-profile/EC2FullAccessS3Role. The 'Revoke sessions' tab is selected, showing a warning about revoking active sessions. A modal window titled 'Revoke active sessions?' is open, asking if the user wants to revoke all active sessions for this role. The user has checked the checkbox 'I acknowledge that I am revoking all active sessions for this role.' Below the modal, there is a sample AWSRevokeOlderSessions policy document.

This screenshot is identical to the one above, but it includes a prominent 'Revoke active sessions?' confirmation dialog box in the center. The dialog contains the same text and checkbox as the previous screenshot. The background page remains the same, showing the IAM role details and the AWSRevokeOlderSessions policy sample.

In response we revoke the active session.

```
austin@astrofree:~
```

```
/33avCY83WL1Xzqbx1ybnullR5UrT+tyf2T21qJaTbf/X1DytrcIIDIGCX41SmrifrwMNaZp+M86ytq0xaXpD+xPScg5r0acCAENNvSbeSVq5/EoAMfkxTwVM/16Wd2jTYpu219gX0hIEo0lGzALJYwBjcnN4GTESQcjzb
8u/+1F0oYcd03asneBk6vL6jv+NxeicvKpkB0EntcojoBJRH1f8/P01/70LoT/JJx5f/ofF0d8cd1vG9VVDVGR1ytjE6v1keIRQuam76WzAHGrcMf9xf2QCPwQ05eJK07jvB5M27RRHtN9n+s8je5iDCG82MzB8xW0Hu
4PfewPx531HOY1b3sFSQ506YSP0vs4RN4191UmI4Ax4zv+0/FvktWlbfSCtxXuf3MyuF1pUpjsUuCa1M26Aw/5ufvAY6sQH67InZ7vVSwBpJtOC1Jb/mUaq50/eJi21Xk20dhmOucRAde9W1kamCRtjZHB4N
hrHpdh6KHi2QiY5Fa18nig910b9az2JVLQNKSQHprT2keyS1tN+sEG2h9UM9DHc/a2f8dhbGHDidb7jwv905W5k9A9Qzmag5GncAL9UmGkbyNwVmxel81hTfqzZ8wgW6yUM31YRP0ceCmuab8iA4EMvRk8CzQavcqs+Sj
nA=
austin@astrofree:~$ export AWS_ACCESS_KEY_ID=ASIA34AMCTAEQUF47YMU
austin@astrofree:~$ export AWS_SECRET_ACCESS_KEY=CbeGphohP//0xm3J0B4edGk4Wxu4s1t+fhyUDkP5
austin@astrofree:~$ export AWS_SESSION_TOKEN=IqcJb3jp2l1u2v2jEdcaXvzLWlh3QtmS.HMEUC1QCTL81Wk352Uu1Msep2ykFR561Elcnm+VKucF1FqgjwIgLdDqQS0Bbxb0ufxdkJh7KH/UEFYb1ZvxRc
n/P261v8quwIMBAAGw4MTVwnjkxMj1wNt1dC2EKn5n10dhngC8CqYBWR1VS01bb0Ypdk97500pxv7o718/+tmd5fAU0mfqxf6tnH8/6aufvNLKzKqkrkvuiAMsvXq1tCusPeD3M3JMpkeVfnCj35sRxR1GtGmuUR
in12IszshC3dd2TkU83zchDMVkmok334a0JkHFcqE4ANkGxpk0wxhGBe4B11dnuaR9R1Nv3U1vSq70lmkPptaVemtaebvNTNA9yFm5CxFR03v3AAUj51oojy1ompBk/GSc3kRQ/djMfs+qfcfndE51s18gtXa8yINog
+cYqmnNu57D2P/zc1Ow-u42547Afwn9t24E89f7QMANJb/2Pe2pMu1f7y/bh7+U8RC6/Yoh1ZVrmn1TT1gac+r1s2Xh/RTXotfkwuGuFy+Y+e80WfBVkehWcLwFswIpy+tg80WWN41abnt35pkhWhCapa8PvmmtUj8+ve3+
/33avCY83WL1Xzqbx1ybnullR5UrT+tyf2T21qJaTbf/X1DytrcIIDIGCX41SmrifrwMnaZp+M86ytq0xaXpD+xPScg5r0acCAENNvSbeSVq5/EoAMfkxTwVM/16Wd2jTYpu219gX0hIEo0lGzALJYwBjcnN4GTESQcjzb
8u/+1F0oYcd03asneBk6vL6jv+NxeicvKpkB0EntcojoBJRH1f8/P01/70LoT/JJx5f/ofF0d8cd1vG9VVDVGR1ytjE6v1keIRQuam76WzAHGrcMf9xf2QCPwQ05eJK07jvB5M27RRHtN9n+s8je5iDCG82MzB8xW0Hu
4PfewPx531HOY1b3sFSQ506YSP0vs4RN4191UmI4Ax4zv+0/FvktWlbfSCtxXuf3MyuF1pUpjsUuCa1M26Aw/5ufvAY6sQH67InZ7vVSwBpJtOC1Jb/mUaq50/eJi21Xk20dhmOucRAde9W1kamCRtjZHB4N
hrHpdh6KHi2QiY5Fa18nig910b9az2JVLQNKSQHprT2keyS1tN+sEG2h9UM9DHc/a2f8dhbGHDidb7jwv905W5k9A9Qzmag5GncAL9UmGkbyNwVmxel81hTfqzZ8wgW6yUM31YRP0ceCmuab8iA4EMvRk8CzQavcqs+Sj
nA=
austin@astrofree:~$ aws sts get-caller-identity
81669122057 arn:aws:sts::81669122057:assumed-role/EC2FullAccessS3Role/i-062f37e072ccfb8a AROA34AMCTAE356NPGBJH:i-062f37e072ccfb8a
austin@astrofree:~$ aws s3 ls
2024-12-27 09:53:44 detectionlabimagesimport
2025-01-13 17:07:35 financial-data-secure-company325
austin@astrofree:~$ aws s3 ls

An error occurred (AccessDenied) when calling the ListBuckets operation: User: arn:aws:sts::81669122057:assumed-role/EC2FullAccessS3Role/i-062f37e072ccfb8a is not authorized to perform: s3>ListAllMyBuckets with an explicit deny in an identity-based policy
austin@astrofree:~$
```

Now our stolen credentials are no longer active, and the session is given an access denied error when trying to list buckets.

The screenshot shows the AWS Management Console with the URL <https://us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#Instances:instanceState=running>. The main pane displays a green success message: "Successfully initiated termination (deletion) of i-062f37e072ccfb8a". Below this, the "Instances (1/1) Info" section shows one terminated instance: "i-062f37e072ccfb8a (S3AccessAWSLinux)". The instance details include its state as "Terminated", type as "t2.micro", and public IP as "18.194.111.11". The "Actions" dropdown menu is open, showing options like "Launch instances", "Stop", "Terminate", and "Reboot". The left sidebar contains navigation links for EC2 Global View, Events, Instances, Images, Elastic Block Store, Network & Security, and Load Balancing.

Finally we terminate the instance and clean up the resources.