

AWS Identity & Access Management (IAM)

Chetan Agrawal, Solutions Architect

Deven Suri, Account Manager

Agenda

Overview of Security in AWS

What is IAM – Authentication & Authorization

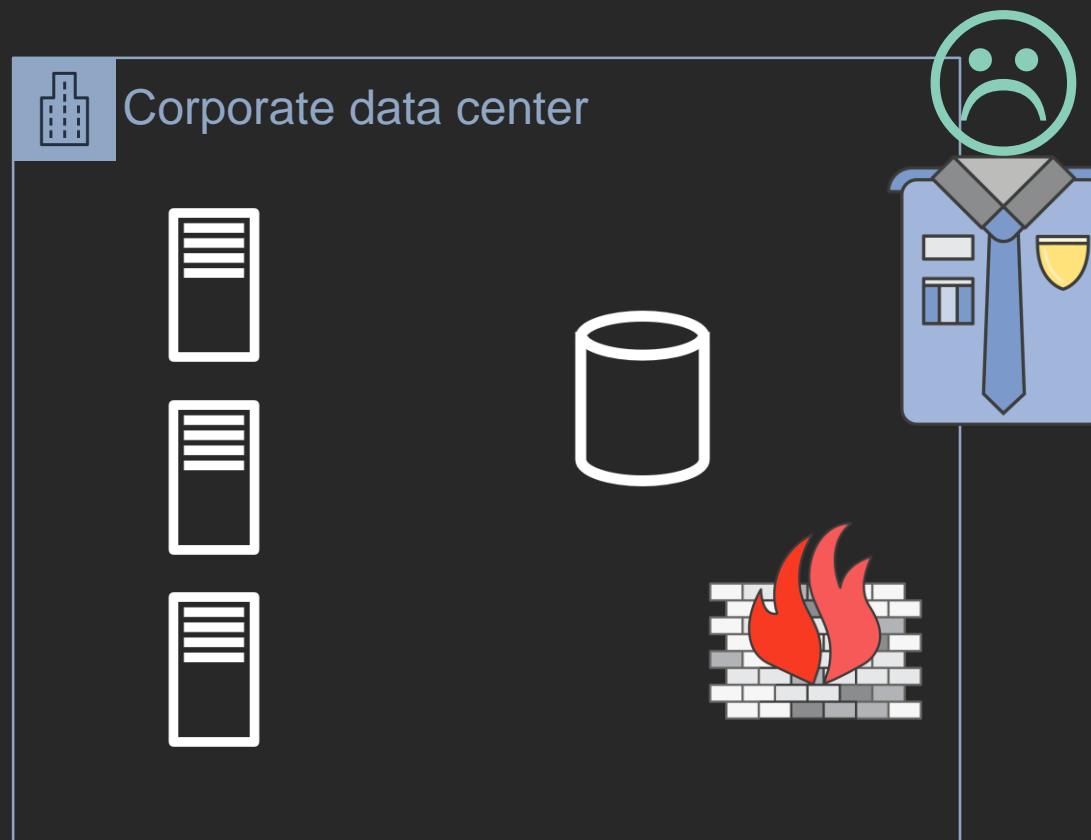
IAM components – Users, Group, Policy, Roles

Multi-account IAM access

IAM examples & best practices

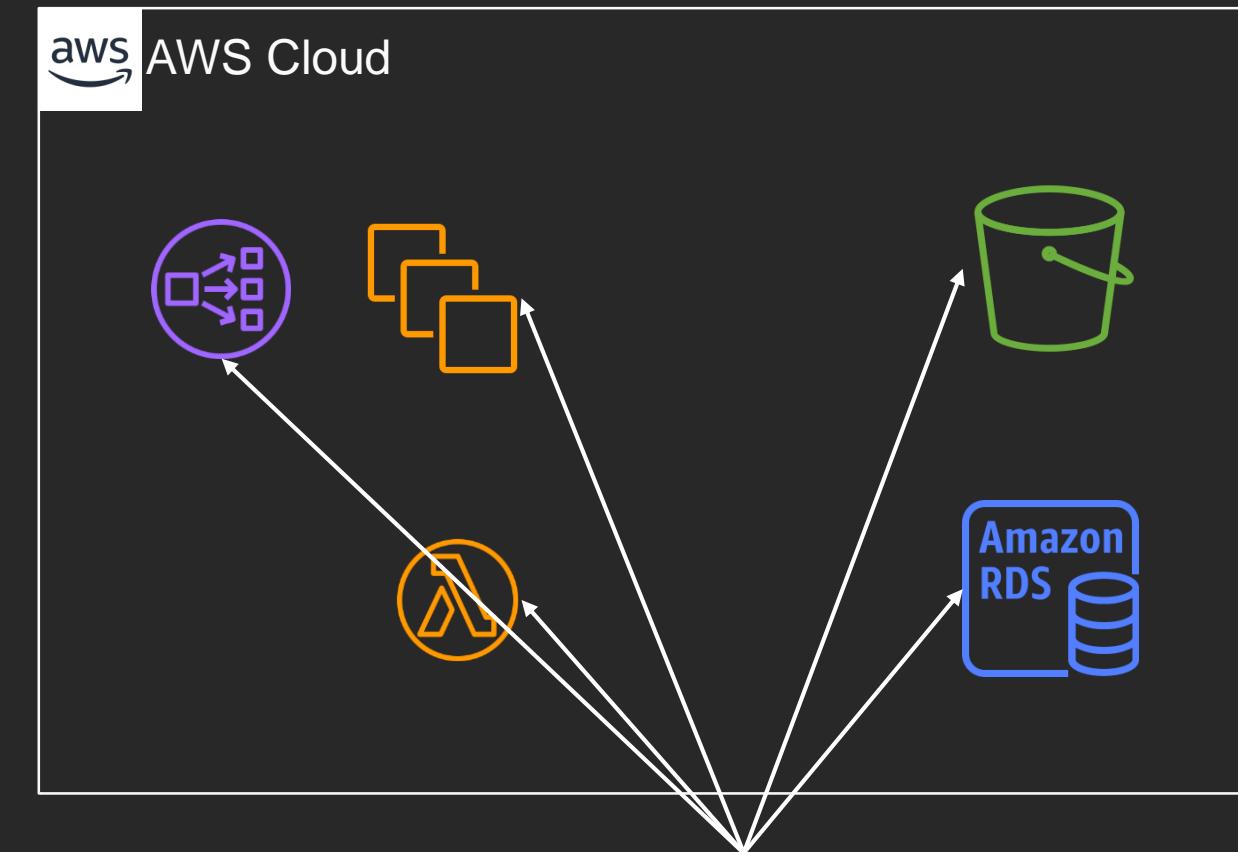
Why learn AWS Identity and Access Management (IAM)

Security before the cloud



Security implemented at perimeter

Security in the cloud



IAM authorization at every resource:
Pervasive security that's part of your applications

AWS security, identity, and compliance solutions



Identity & access management

AWS Identity & Access Management (IAM)
AWS Single Sign-On
AWS Organizations
AWS Directory Service
Amazon Cognito
AWS Resource Access Manager



Detection

AWS Security Hub
Amazon GuardDuty
Amazon Inspector
Amazon CloudWatch Metrics
AWS Config
AWS CloudTrail
VPC Flow Logs



Infrastructure protection

AWS Firewall Manager
AWS Shield
AWS WAF – Web application firewall
Amazon Virtual Private Cloud (VPC)
AWS PrivateLink
AWS Systems Manager



Data protection

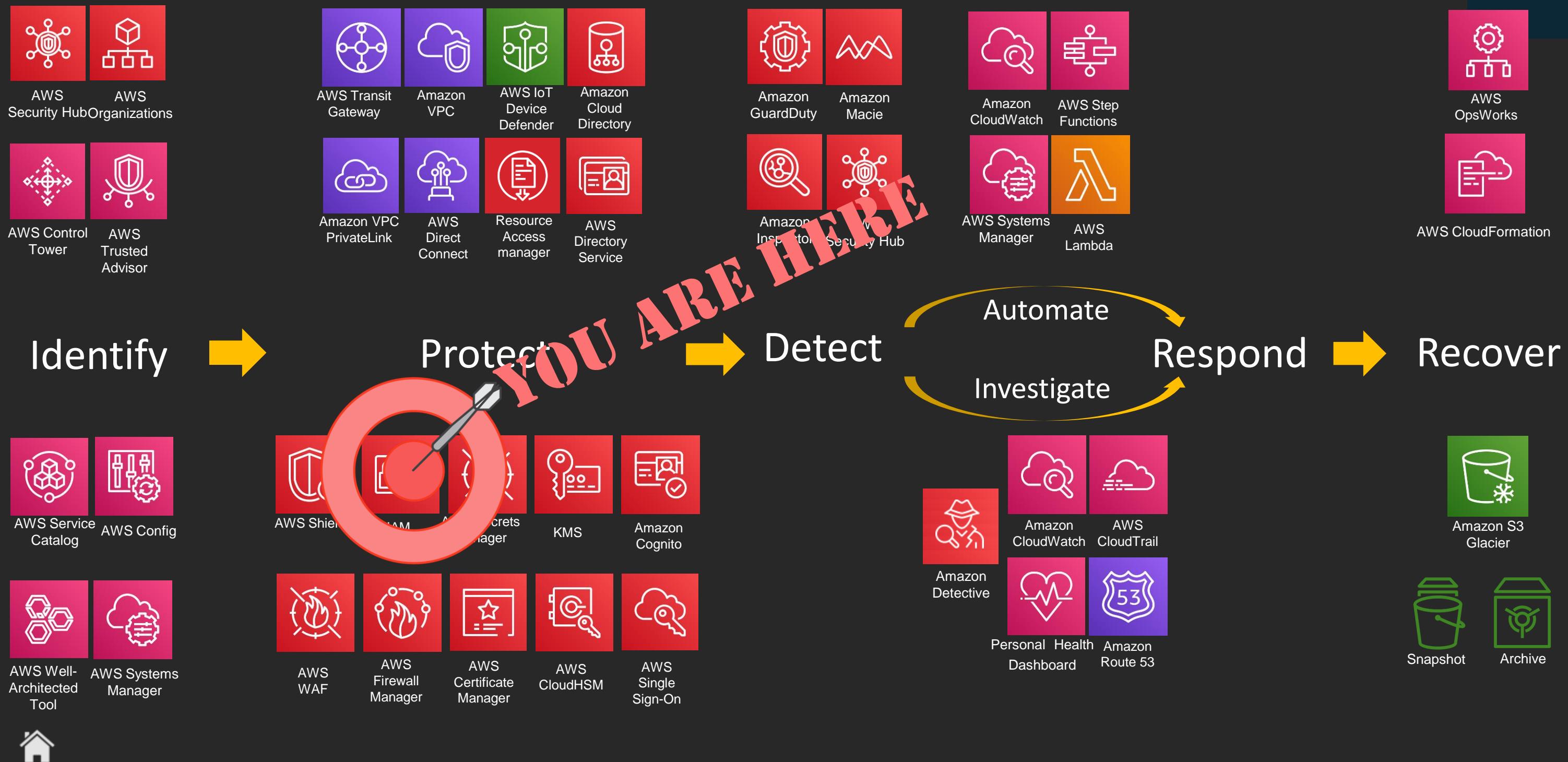
Amazon Macie
AWS Key Management Service (KMS)
AWS CloudHSM
AWS Certificate Manager
AWS Secrets Manager
AWS VPN
Server-Side Encryption



Incident response

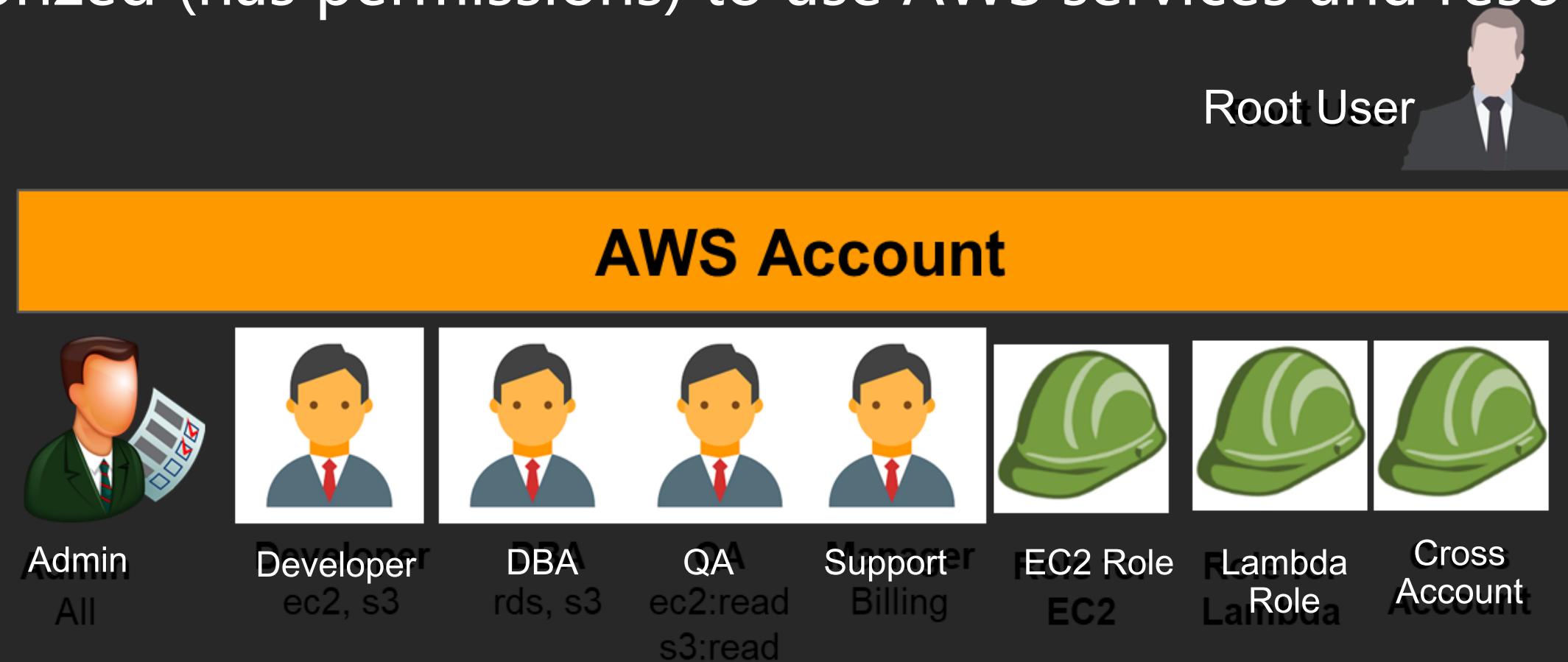
Amazon Detective
CloudEndure DR
AWS Config Rules
AWS Lambda

AWS Foundational and Layered Security Services



IAM

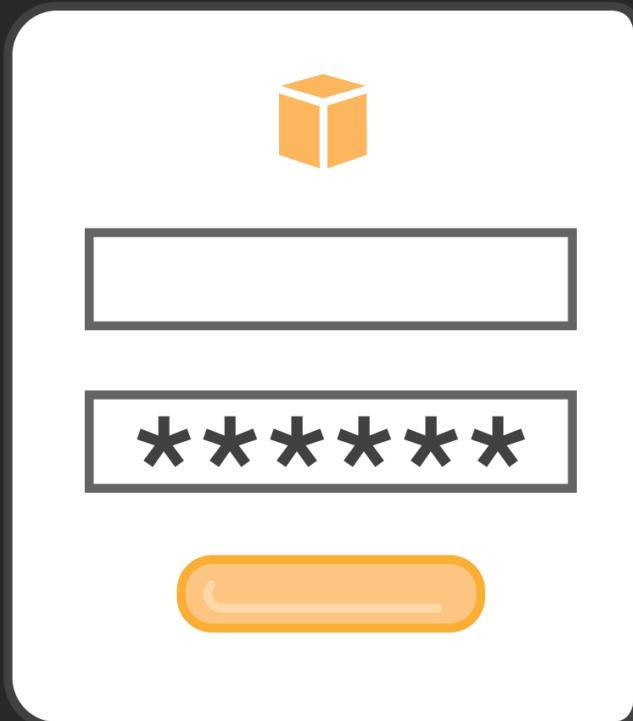
- Allows you to securely control access to AWS Services and Resources
- You use IAM to control who is authenticated (signed in) and authorized (has permissions) to use AWS services and resources.



IAM Authentication



AWS Console Access



IAM
User

AWS Services

Amazon Web Services

Compute

- EC2
- EC2 Container Service
- Elastic Beanstalk
- Lambda

Storage & Content Delivery

- S3
- CloudFront
- Elastic File System
- Glacier
- Import/Export Snowball
- Storage Gateway

Database

- RDS
- DynamoDB
- ElastiCache
- Redshift
- DMS

Networking

- VPC
- Direct Connect
- Route 53

Developer Tools

- CodeCommit
- CodeDeploy
- CodePipeline

Management Tools

- CloudWatch
- CloudFormation
- CloudTrail
- Config
- OpsWorks
- Service Catalog
- Trusted Advisor

Security & Identity

- Identity & Access Management
- Directory Service
- Inspector
- WAF
- Certificate Manager

Analytics

- EMR
- Data Pipeline
- Elasticsearch Service
- Kinesis
- Machine Learning

Internet of Things

- AWS IoT

Mobile Services

- Mobile Hub
- Cognito
- Device Farm
- Mobile Analytics
- SNS

Application Services

- API Gateway
- AppStream
- CloudSearch
- Elastic Transcoder
- SES
- SQS
- SWF

Enterprise Applications

- WorkSpaces
- WorkDocs
- WorkMail

Resource Groups

A resource group is a collection of resources that share one or more tags. Create a group for each project, application, or environment in your account.

Create a Group Tag Editor

Additional Resources

Getting Started Read our documentation or view our training to learn more about AWS.

AWS Console Mobile App View your resources on the go with our AWS Console mobile app, available from Amazon Appstore, Google Play, or iTunes.

AWS Marketplace Find and buy software, launch with 1-Click and pay by the hour.

AWS re:Invent Announcements Explore the next generation of AWS cloud capabilities. See what's new

Service Health

All services operating normally. Updated: Jan 28 2016 14:59:02 GMT+0800

Service Health Dashboard

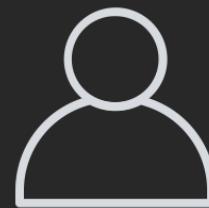
Feedback English

© 2008 - 2016, Amazon Web Services, Inc. or its affiliates. All rights reserved. Privacy Policy Terms of Use

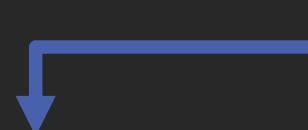


AWS IAM Authentication

- AWS CLI or SDK API
Access Key and Secret Key



IAM User



Access Key ID: AKIAIOSXXXXXXXXXXDORIK
Secret Access Key: wJalrXUtnFEXXXXXXXXXXXXXXXNIFLEKEY

AWS CLI

```
:~ $ aws configure
AWS Access Key ID [*****022A]:
AWS Secret Access Key [*****4m8i]:
Default region name [ap-southeast-1]:
Default output format [json]:
```

AWS SDK & API



Java

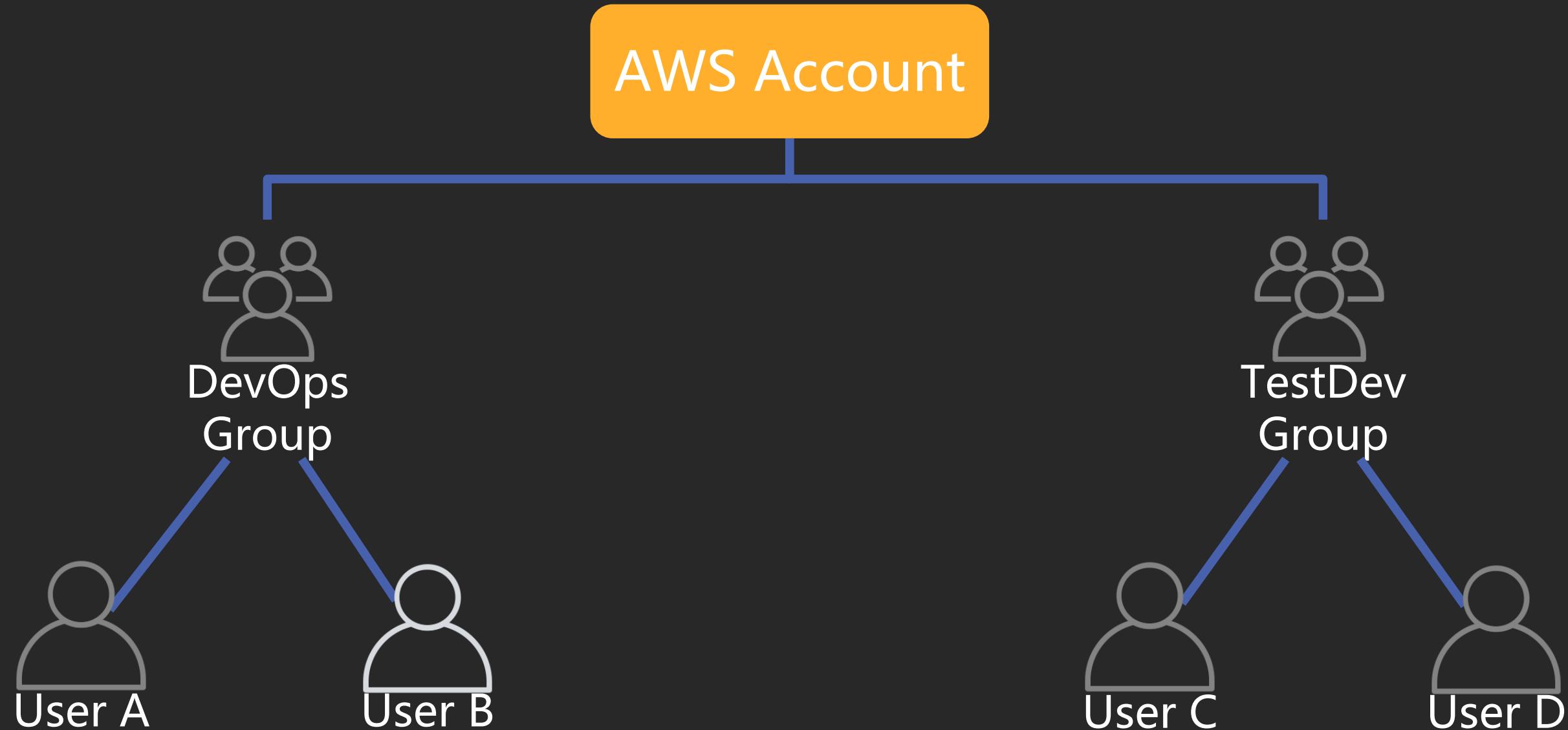


Python



.NET

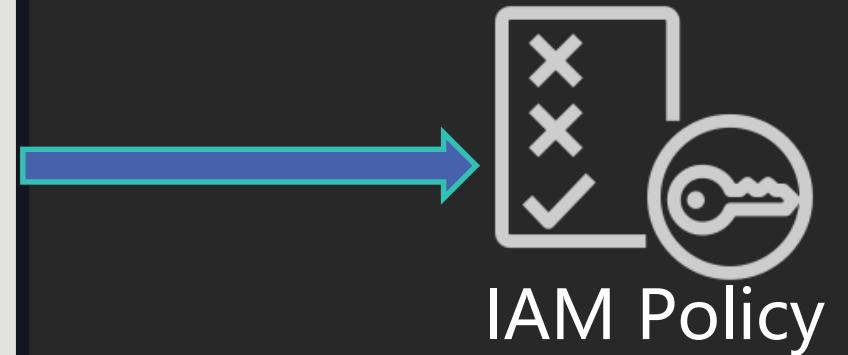
IAM Users and Groups



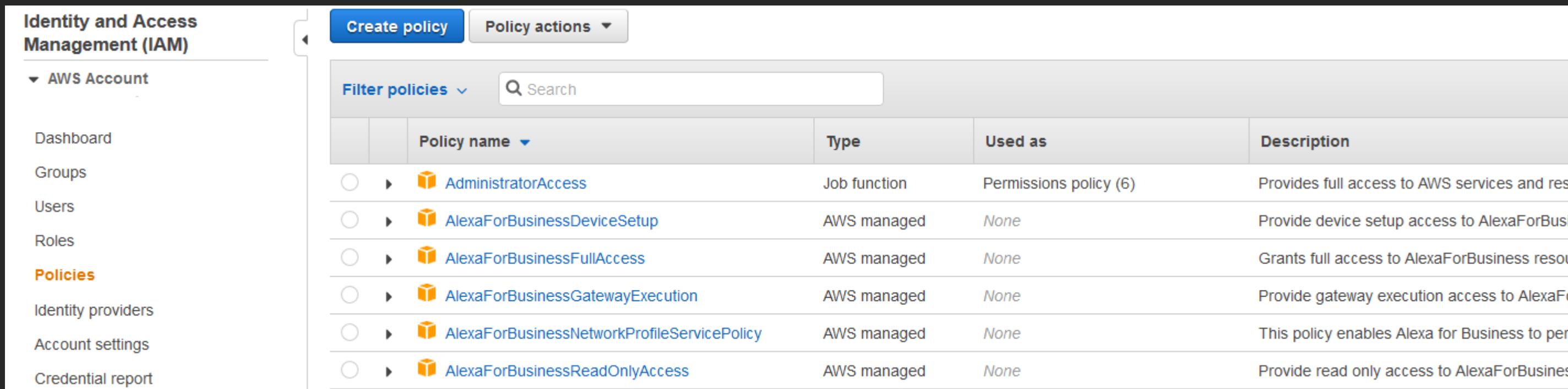


IAM Authorization – IAM Policy

```
{  
  "Version": "2012-10-17",  
  "Statement": [  
    {  
      "Sid": "Stmt1453690971587",  
      "Effect": "Allow",  
      "Action": [  
        "ec2:Describe*",  
        "ec2:StartInstances",  
        "ec2:StopInstances"  
      ],  
      "Resource": "*",  
      "Condition": {  
        "IpAddress": {  
          "aws:SourceIp": "54.64.34.65/32"  
        }  
      }  
    }  
  ]  
}
```



Assigning AWS managed policies



The screenshot shows the AWS Identity and Access Management (IAM) Policies page. The left sidebar includes options like AWS Account, Dashboard, Groups, Users, Roles, and Policies (which is selected). The main area has tabs for 'Create policy' and 'Policy actions'. A search bar and a 'Filter policies' dropdown are also present. The table lists six AWS-managed policies:

	Policy name	Type	Used as	Description
<input type="radio"/>	AdministratorAccess	Job function	Permissions policy (6)	Provides full access to AWS services and resources.
<input type="radio"/>	AlexaForBusinessDeviceSetup	AWS managed	None	Provide device setup access to AlexaForBusiness.
<input type="radio"/>	AlexaForBusinessFullAccess	AWS managed	None	Grants full access to AlexaForBusiness resources.
<input type="radio"/>	AlexaForBusinessGatewayExecution	AWS managed	None	Provide gateway execution access to AlexaForBusiness.
<input type="radio"/>	AlexaForBusinessNetworkProfileServicePolicy	AWS managed	None	This policy enables Alexa for Business to perform network profile service operations.
<input type="radio"/>	AlexaForBusinessReadOnlyAccess	AWS managed	None	Provide read only access to AlexaForBusiness resources.

. . . and many other AWS-written policies

Example: Administrator policy

Policies > AdministratorAccess

Summary

Policy ARN	arn:aws:iam::aws:policy/AdministratorAccess 		
Description	Provides full access to AWS services and resources.		
Permissions	Policy usage	Policy versions	Access Advisor
Policy summary 			
1	{		
2	"Version": "2012-10-17",		
3	"Statement": [
4	{		
5	"Effect": "Allow",		
6	>Action": "*",		
7	"Resource": "*"		
8	}		
9]		
10	}		

Example: Read-only policy

The screenshot shows the AWS IAM 'Summary' page for a policy named 'ReadOnlyAccess'. The 'Policy ARN' is highlighted with a yellow box and contains the value 'arn:aws:iam::aws:policy/ReadOnlyAccess'. The 'Description' field states 'Provides read-only access to AWS services and resources'. Below the summary, there are tabs for 'Permissions', 'Policy usage', 'Policy versions', and 'Access Advisor'. The 'Permissions' tab is selected, showing a JSON policy document. The JSON code is as follows:

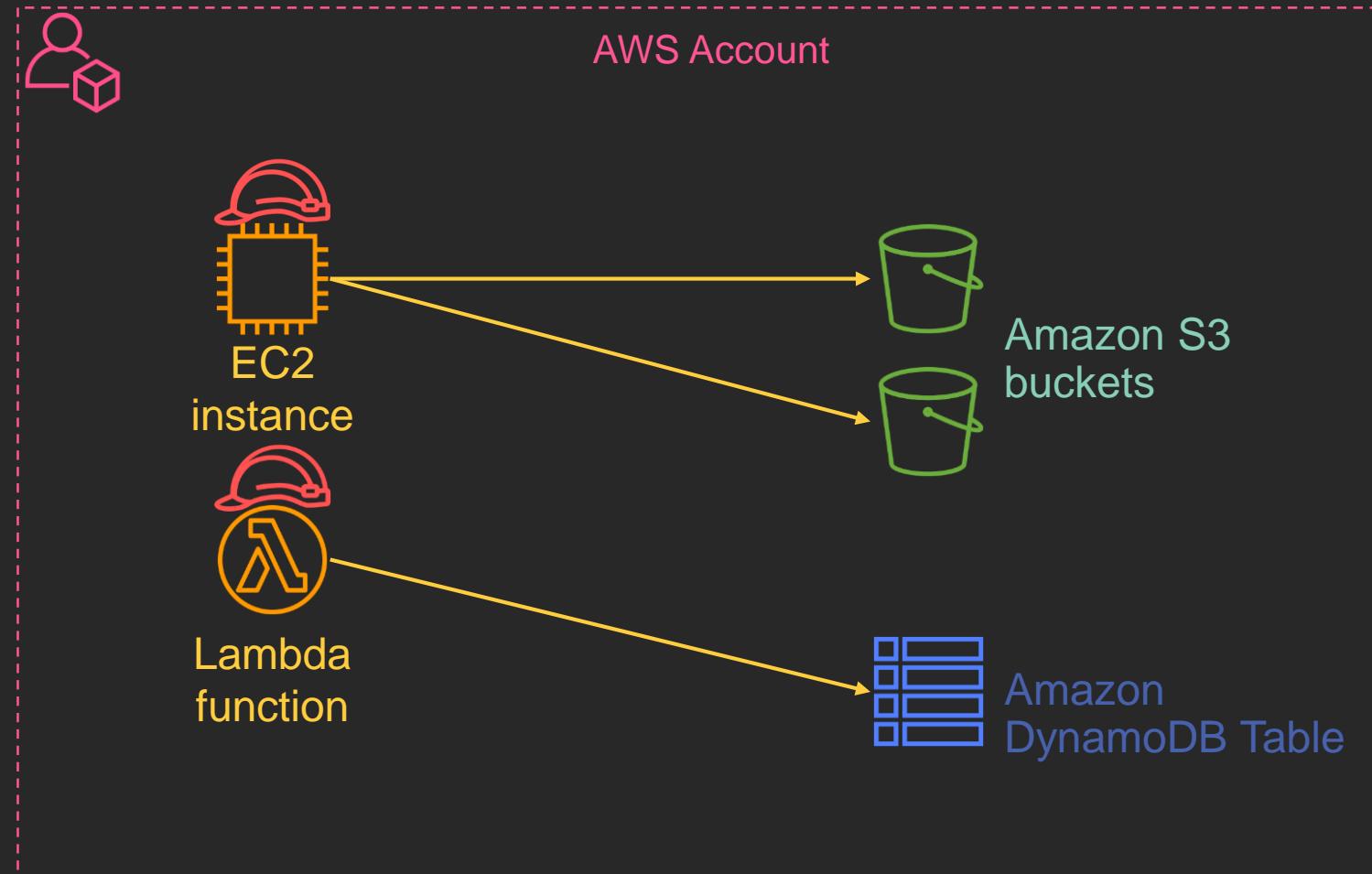
```
1 {  
2   "Version": "2012-10-17",  
3   "Statement": [  
4     {  
5       "Action": [  
6         "a4b:Get*",  
7         "a4b>List*",  
8         "a4b:Describe*",  
9         "a4b:Search*"  
10      ]  
11    }  
12  ]  
13}
```

. . . and many other ReadOnly APIs

Let's create new IAM User and assign some permissions..

- Create new IAM User
- Grant AWS Console Access
- Grant EC2 Read only Access
- Verify the Access
- Now add EC2 Full Access permissions
- Check the effect

IAM roles for non-human access



Use IAM roles for access to AWS resources from:

- Your application running on an AWS compute environment, e.g., EC2 instance, Lambda function, etc.
- Permission to an AWS service to access your resources (not shown)

Creating IAM roles for non-human access

Create role

Select type of trusted entity

1 2 3 4

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 the service that will use this role

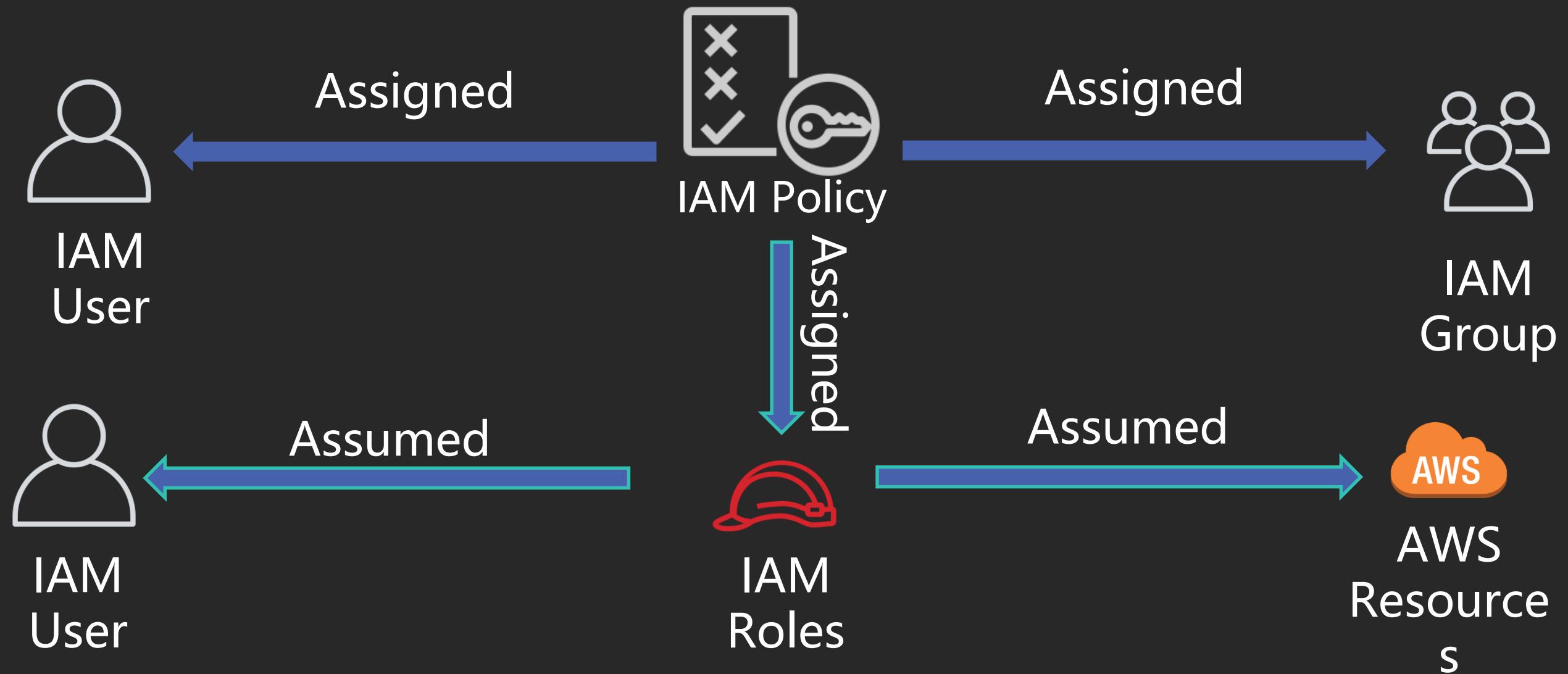
EC2
Allows EC2 instances to call AWS services on your behalf.

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

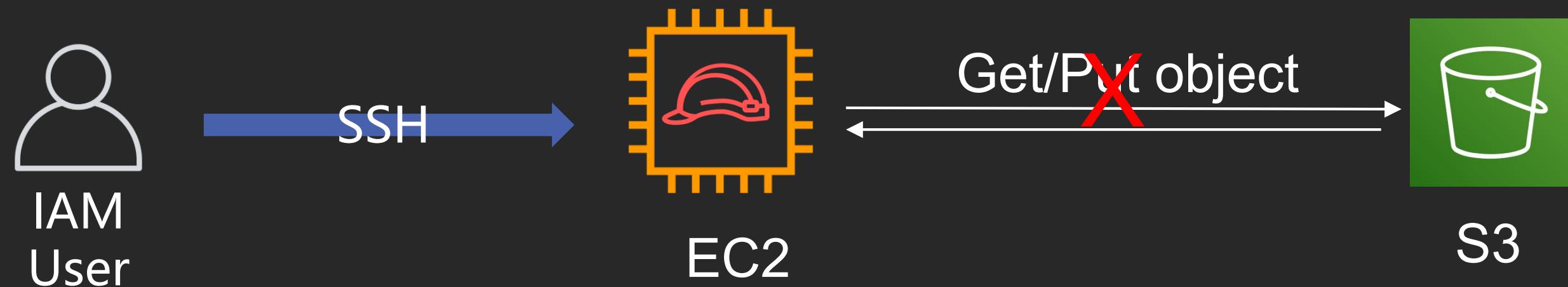
API Gateway Comprehend ElastiCache Lambda S3
AWS Backup Config Elastic Beanstalk Lex SMS
AWS Support Connect Elastic Container Service License Manager SNS

EC2 instances will have access to short-term credentials to access AWS resources

IAM Policy Assignment



Let's understand how IAM role works



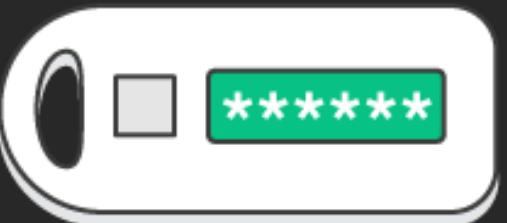
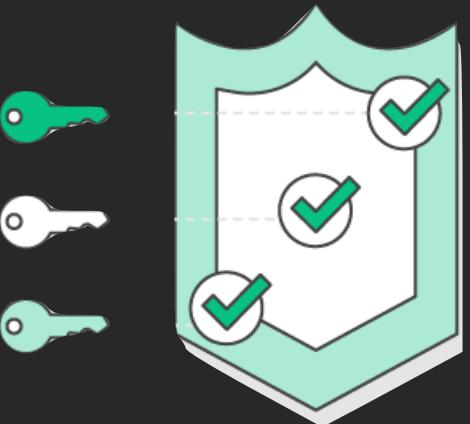
IAM policy for S3 access

```
{  
  "Version": "2012-10-17",  
  "Statement": [  
    {  
      "Effect": "Allow",  
      "Action": [  
        "s3>List*",  
        "s3:GetObject"  
      ],  
      "Resource": [  
        "arn:aws:s3:::bucket-name",  
        "arn:aws:s3:::bucket-name/*"  
      ]  
    }  
  ]  
}
```

AWS IAM Best Practices

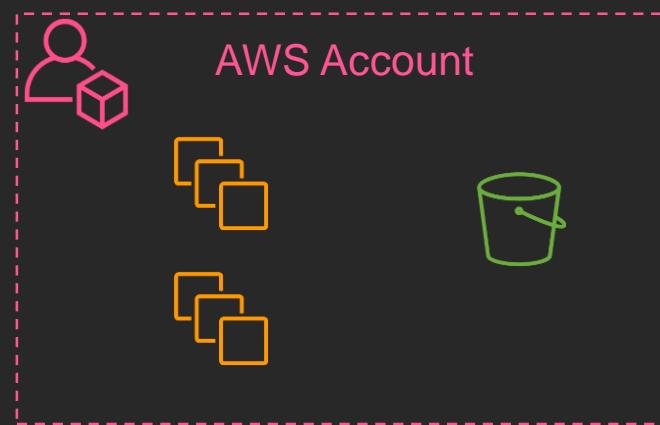


- Create individual IAM users. Don't share user/credentials.
- Grant least privilege.
- Enable MFA.
- Use roles for applications (instead of sharing credentials).
- Rotate credentials regularly.
- Use policy conditions for extra security.
- Monitor activity in your AWS account.



How to manage IAM access when there are more than one AWS accounts?

Accounts in AWS



Accounts in AWS



AWS Account: ★
MASTER

AWS Organization



Organizational Unit



Organizational Unit



AWS Account



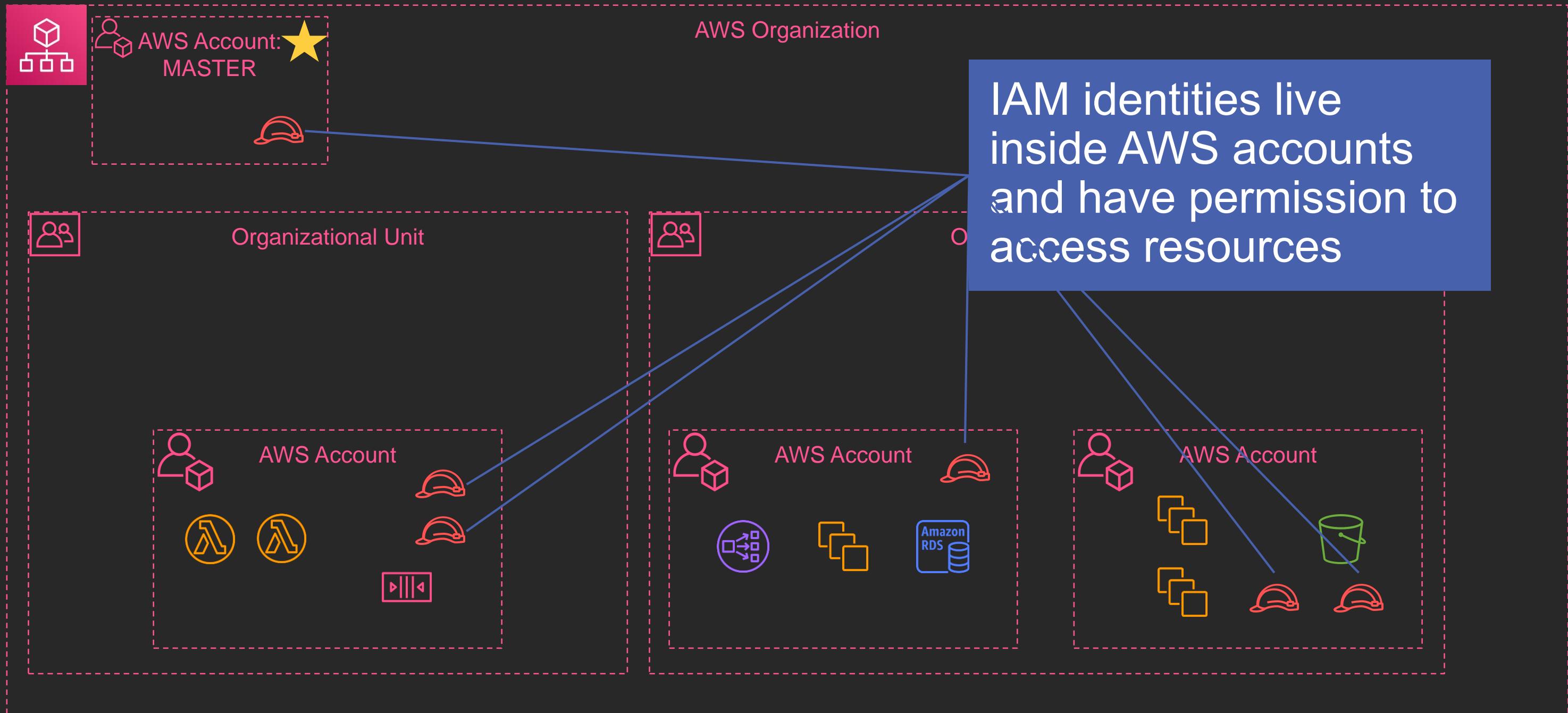
AWS Account



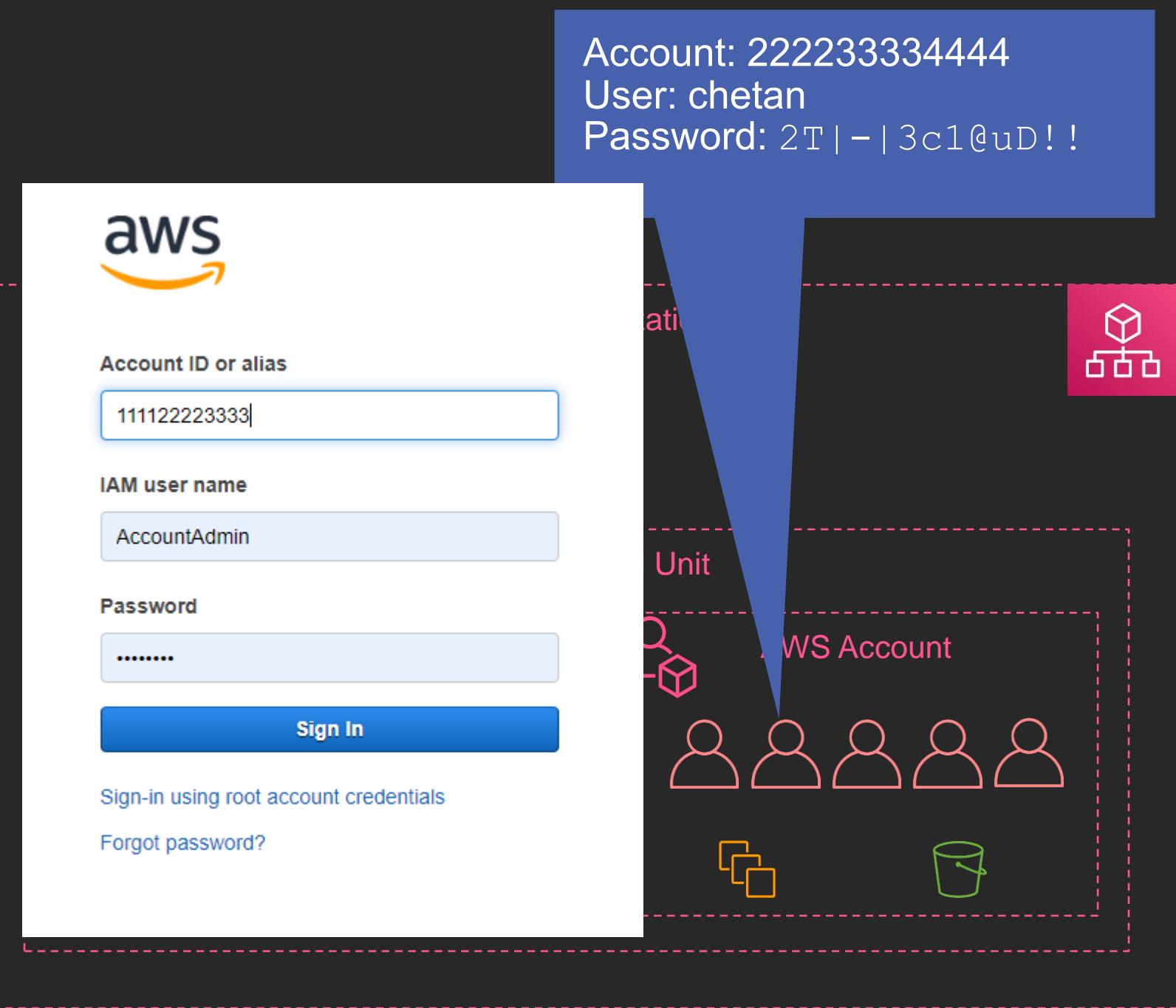
AWS Account



Accounts in AWS



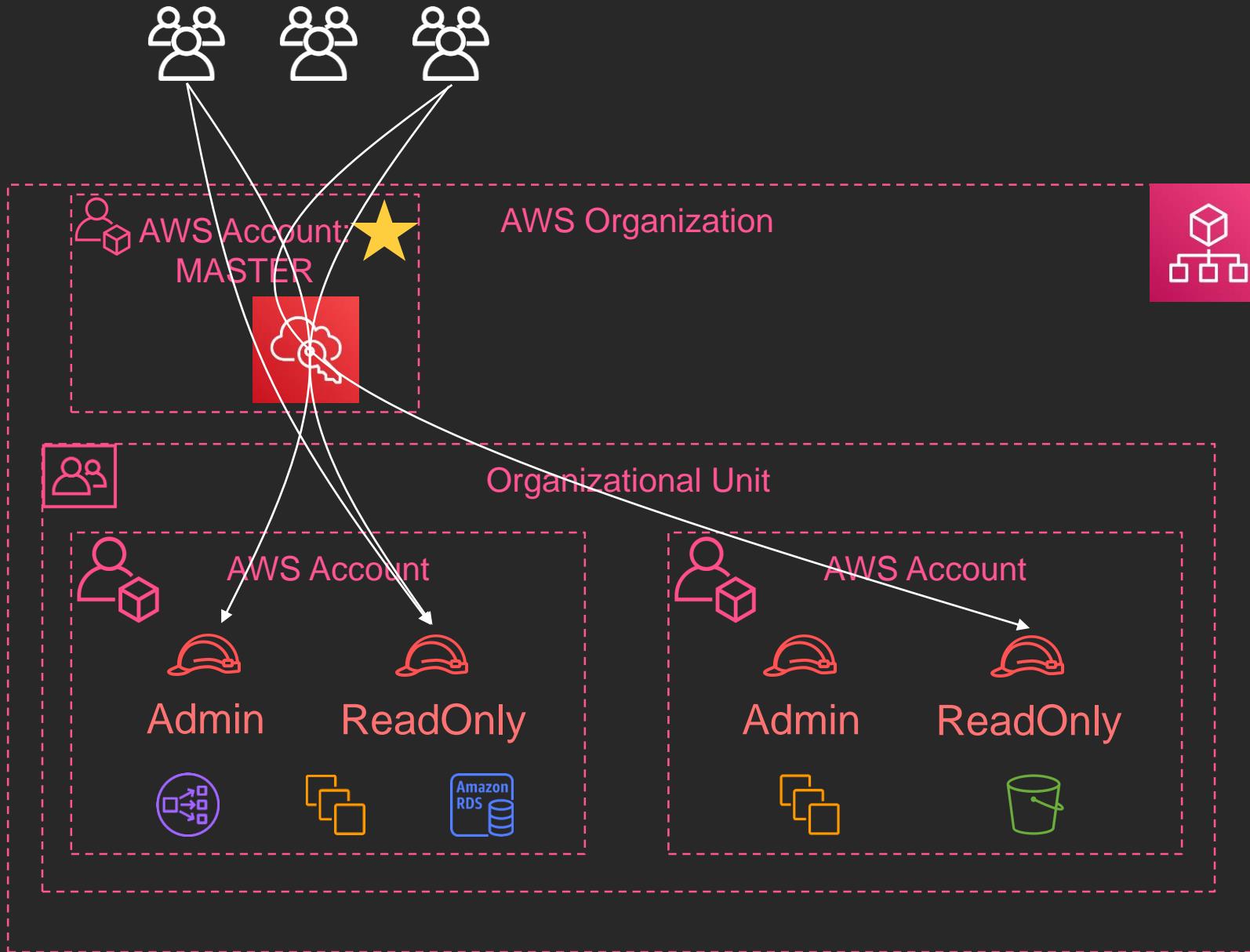
IAM users



Works best when you have:

- A relatively small number of users
- One AWS account, or a relatively small number of them
- A need for long-term credentials
- No user directory, or no ability to connect your directory to AWS
- Your very first AWS account

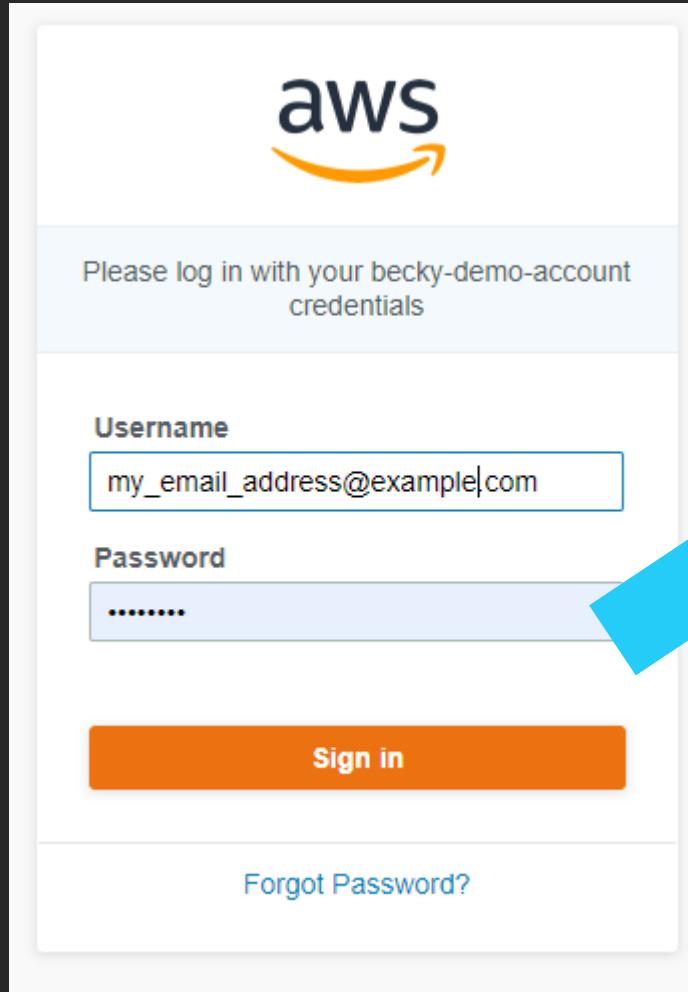
AWS Single Sign-On user pool



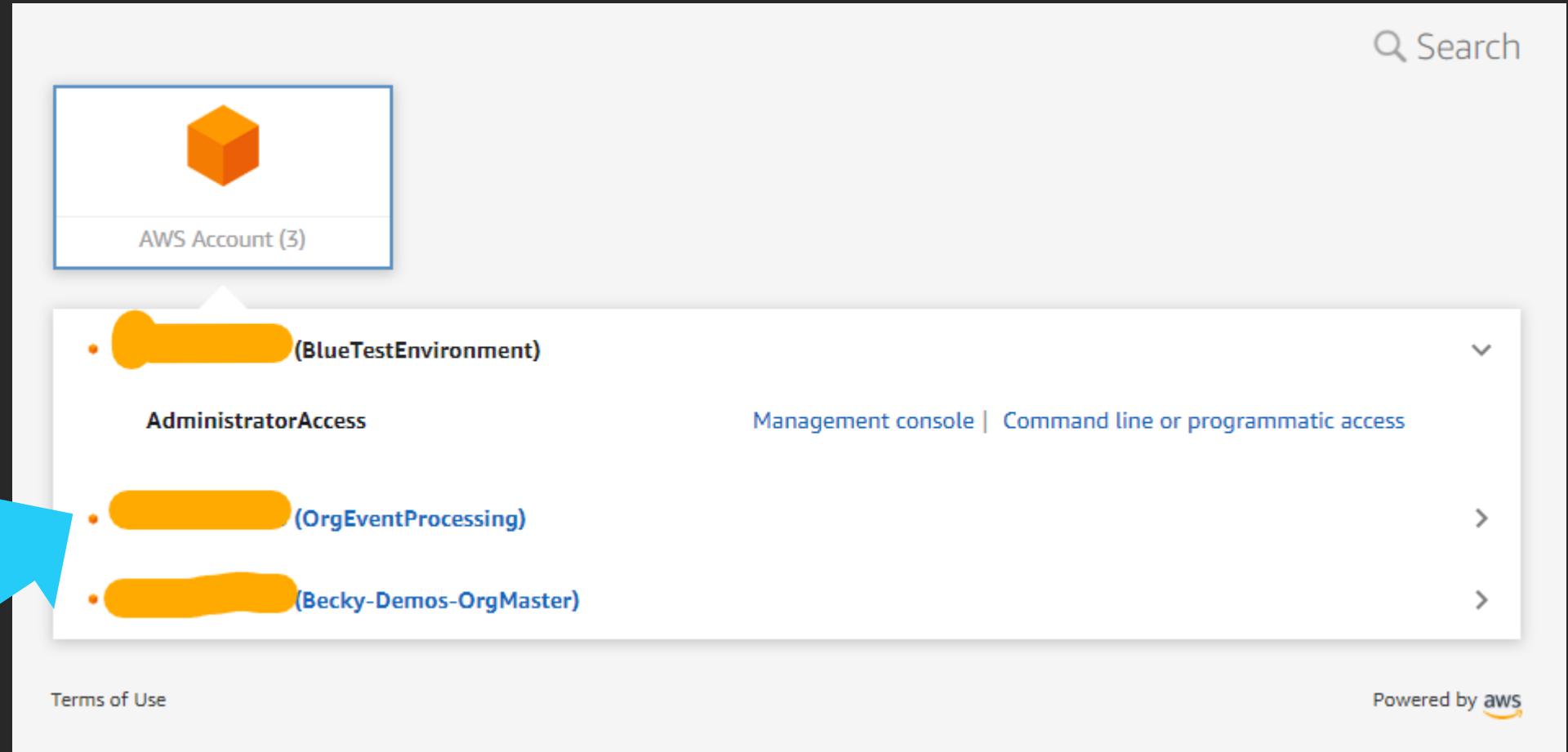
Works best when you have:

- A relatively small number of users
- Simple authorization schemes of humans into AWS
- Rules to map groups of users to AWS environments
- No user directory, or no ability to connect your directory to AWS

AWS SSO user pool: Sign-in workflow

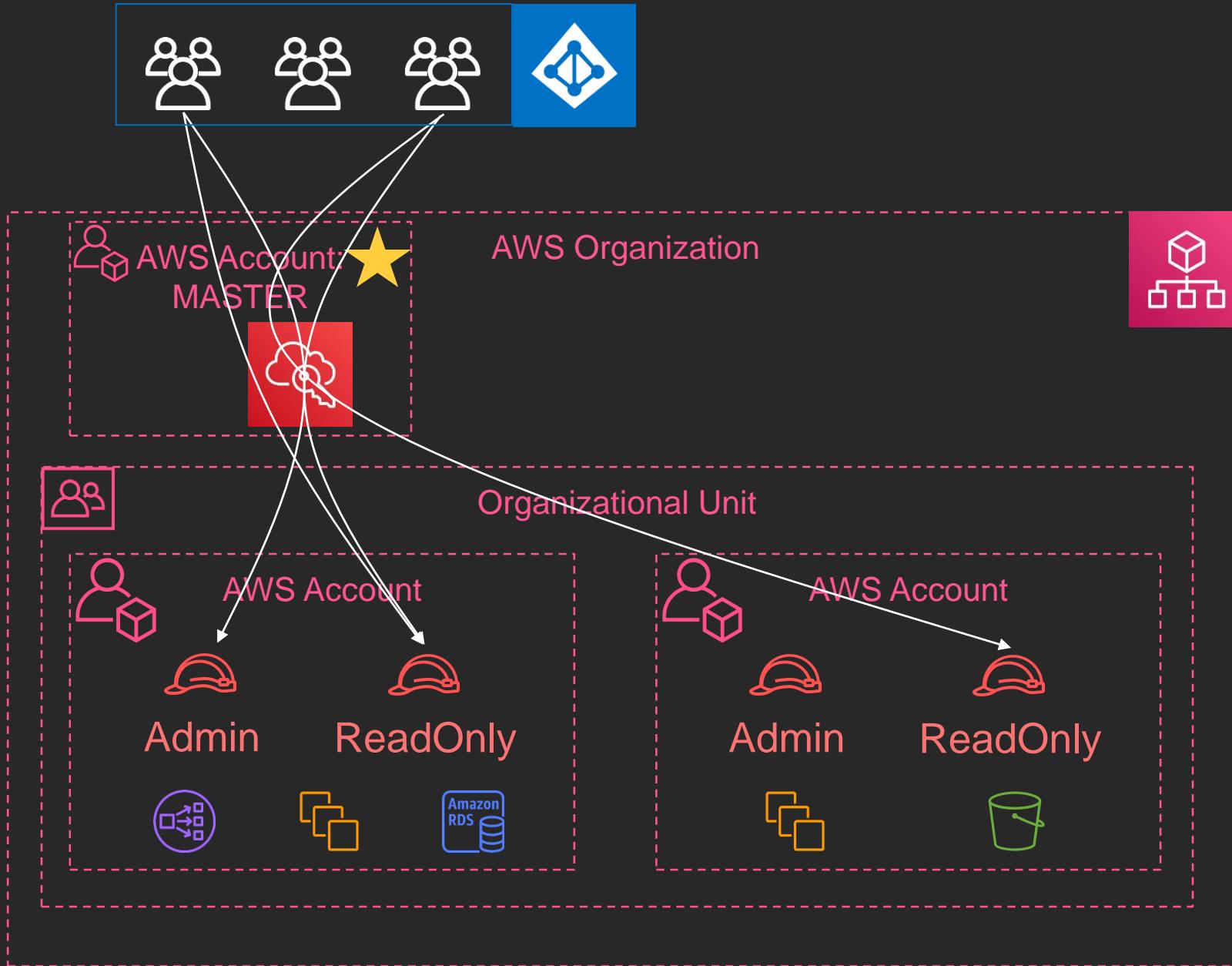


The screenshot shows the AWS SSO sign-in interface. It features the AWS logo at the top left. Below it, a message reads "Please log in with your becky-demo-account credentials". The form has two fields: "Username" containing "my_email_address@example.com" and "Password" with a redacted value. A large orange "Sign in" button is at the bottom, and a "Forgot Password?" link is below it.



The screenshot shows the AWS SSO sign-in workflow. On the right, there's a navigation bar with a search bar and links for "Management console" and "Command line or programmatic access". The main area displays the "AWS Account (3)" section, which contains three items: "BlueTestEnvironment", "AdministratorAccess", and "Becky-Demos-OrgMaster". A blue arrow points from the "Sign in" button on the left towards the "AdministratorAccess" item. At the bottom, there are links for "Terms of Use" and "Powered by aws".

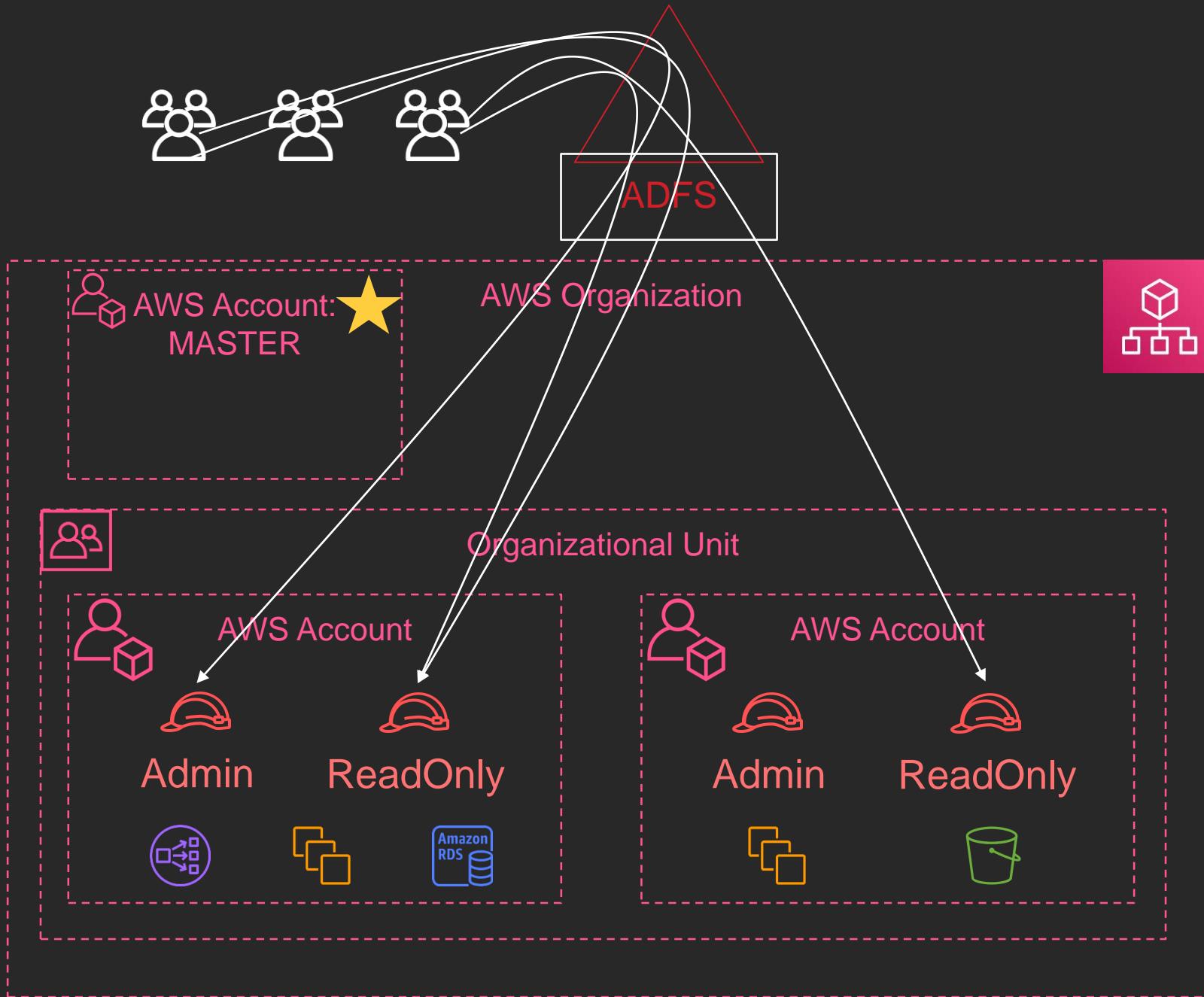
New: AWS Single Sign-On with Azure AD



Works best when you have:

- Azure Active Directory as identity source of truth
- Ability to enable SCIM
- Simple authorization schemes of humans into AWS
- Rules to map groups of users to AWS environments

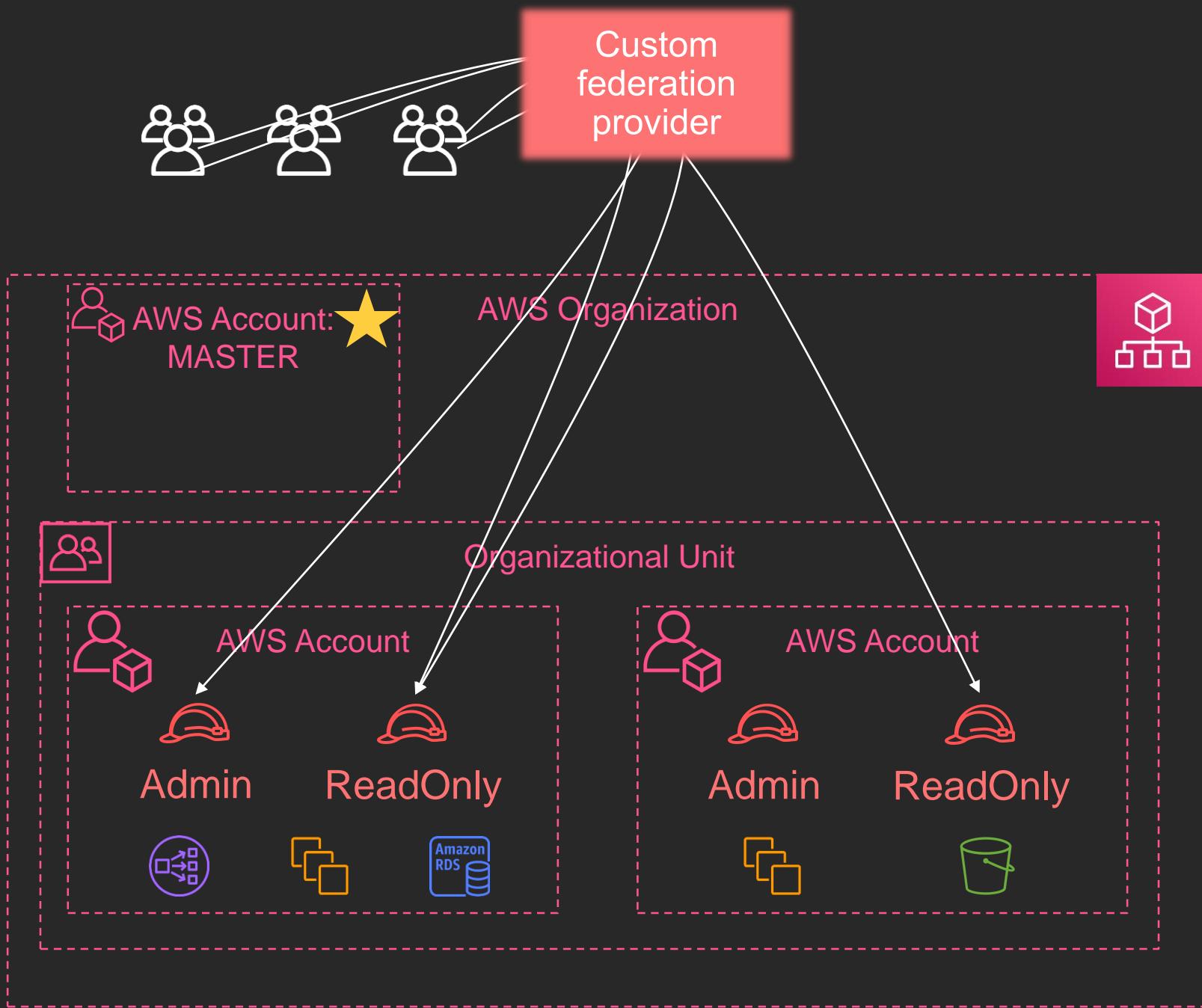
Active Directory Federation Services



Works best when you have:

- Corporate users in a Microsoft Active Directory, either on-premises or managed in AWS
- An ADFS connected to your directory
- Control over ADFS claims
- A need for granular control over user permissions

Custom directory integration



Works best when you have:

- Corporate users in a directory
- Either a third-party SaaS offering or a custom federation solution
- Rules to map groups of users to AWS environments
- A need for granular control over user permissions

Authenticating to AWS: Quick decision framework

If you have an existing user directory:

- AWS SSO with directory integrations
- Bring your own SAML federation (e.g., ADFS)
- Advanced use cases: Custom federation

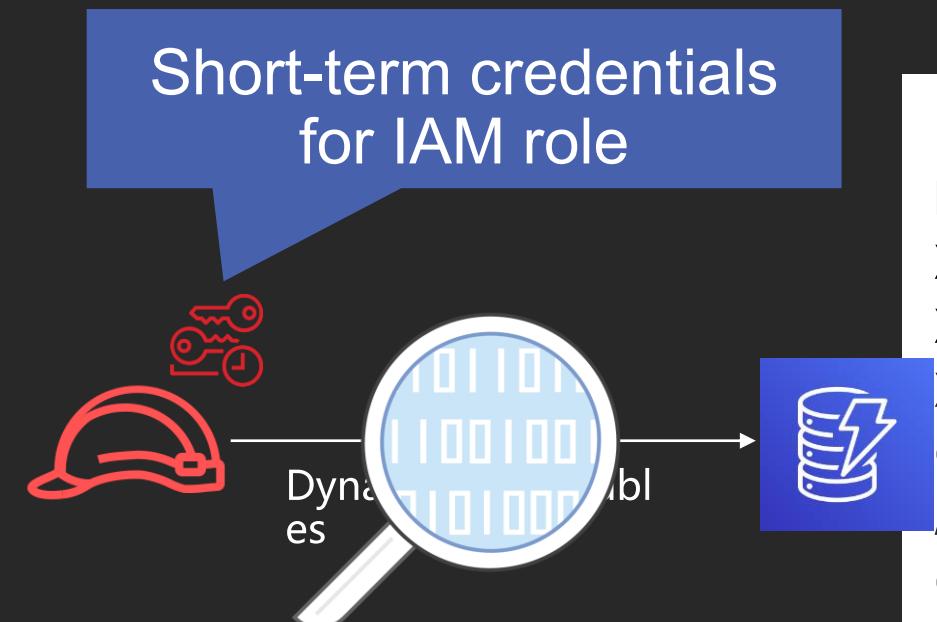
If you don't have an existing user directory:

- AWS SSO with user pools
- IAM Users

How IAM authentication and authorization works?

How authentication works in AWS

(You don't need to know this part)



```
POST https://dynamodb.us-east-2.amazonaws.com/ HTTP/1.1
Host: dynamodb.us-east-2.amazonaws.com
X-Amz-Date: 20180918T150746Z
X-Amz-Target: DynamoDB_20120810.ListTables
X-Amz-Security-Token: FQoGZXIVYXdxZEKH////////// ...
Content-Type: application/x-amz-json-1.0
Authorization: AWS4-HMAC-SHA256 Credential=ASIAXXXXXXXXXXXXXX/20180918/us-east-1/dynamodb/aws4_request, SignedHeaders=content-type;host;x-amz-date;x-amz-security-token;x-amz-target, Signature=c1b4bc2df0c47c86cbcfa54d932e8aaa455b6b7c38e65d840f722254add1ea9e
```

Access key ID:
Identifies calling IAM principal to AWS

HMAC signature:
Generated with secret key;
validated by AWS

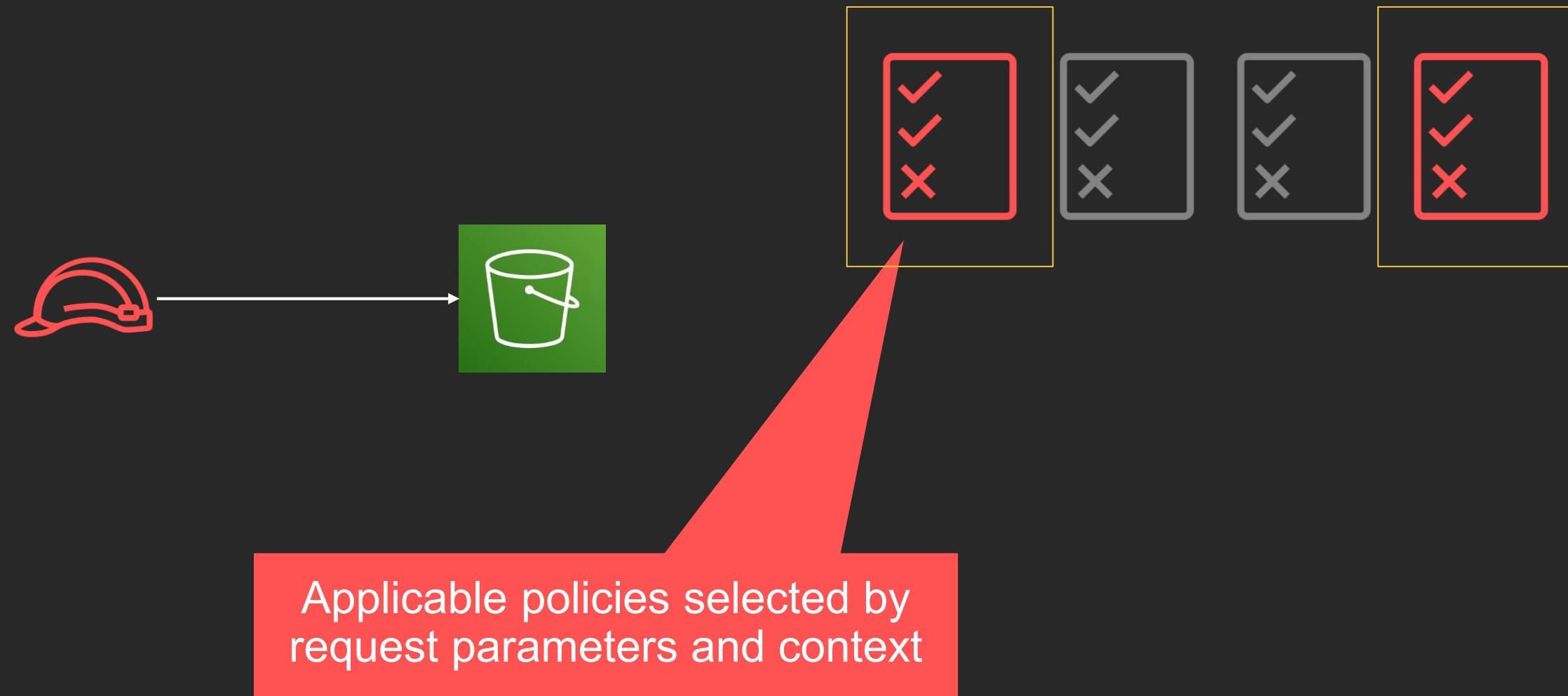
How authorization works in AWS

(You DO need to know this part)



How authorization works in AWS

(You DO need to know this part)



How authorization works in AWS

(You DO need to know this part)

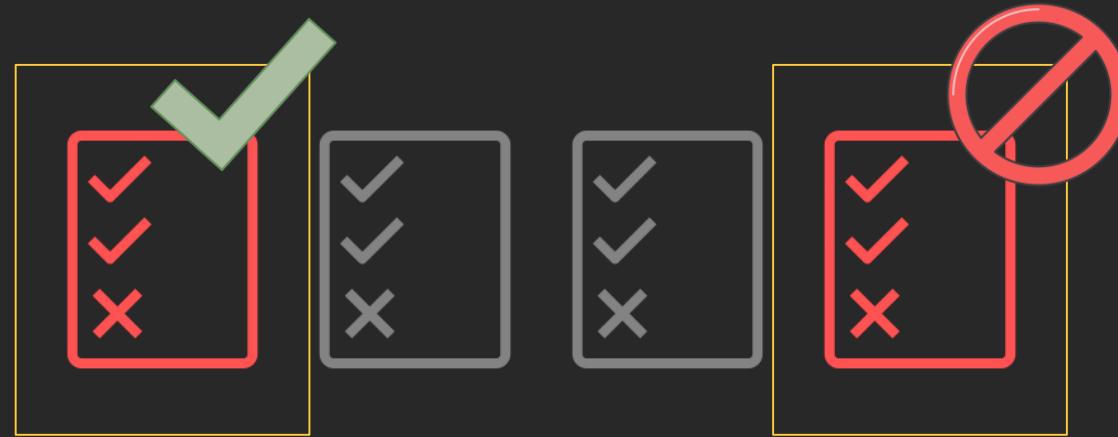


The rules:

- If any policy denies → AccessDenied
- If some policy allows → Allow
- Otherwise AccessDenied

How authorization works in AWS

(You DO need to know this part)

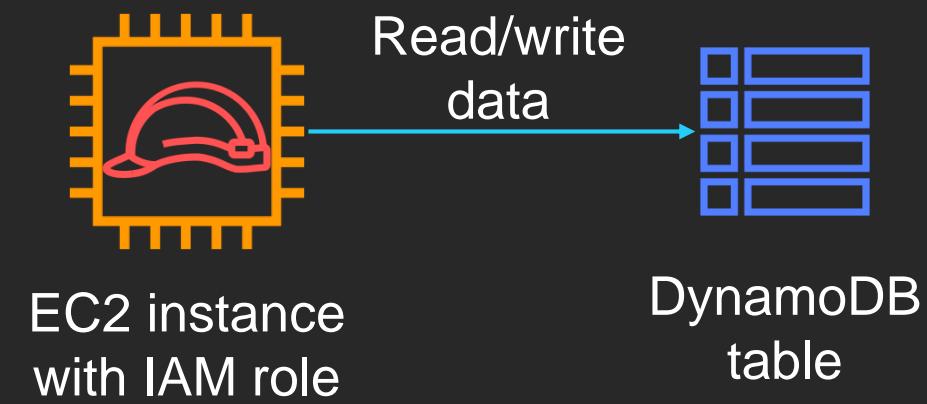


The rules:

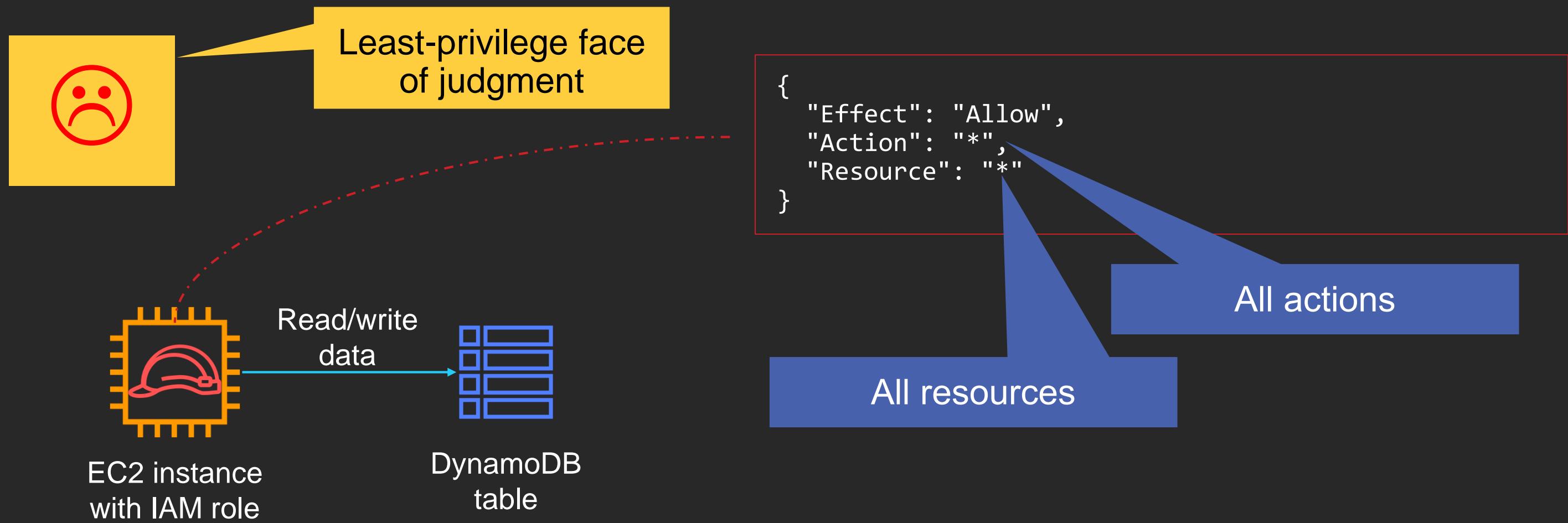
- If any policy denies → AccessDenied
- If some policy allows → Allow
- Otherwise AccessDenied

Reading and writing IAM policy

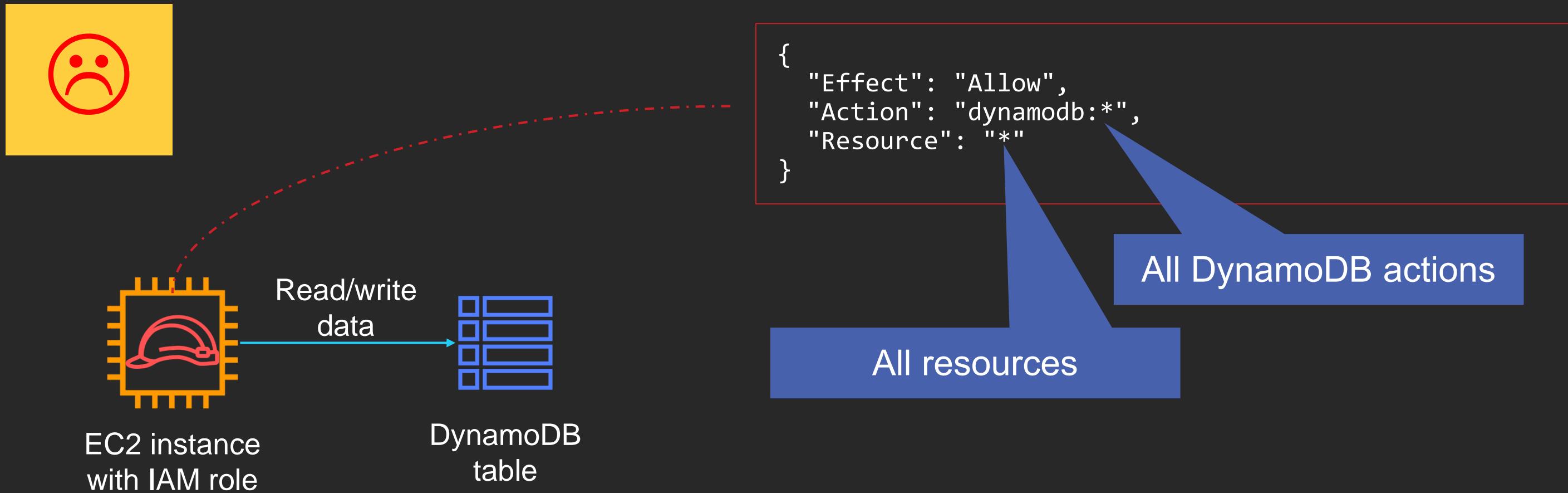
Example 1: Read data from DynamoDB



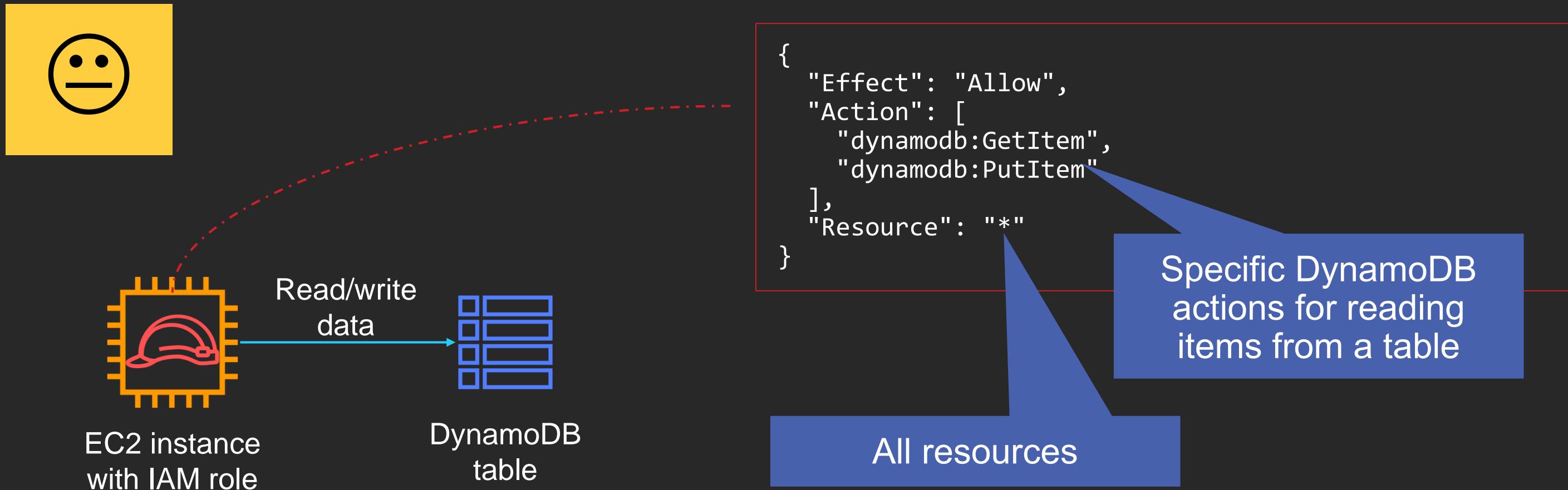
Example 1: Read data from DynamoDB



Example 1: Read data from DynamoDB



Example 1: Read data from DynamoDB



Reading the IAM documentation page

Bookmark this page:

https://docs.aws.amazon.com/IAM/latest/UserGuide/reference_policies.html

The screenshot shows the AWS IAM JSON Policy Reference page. The left sidebar contains navigation links for Tutorials, Best Practices and Use Cases, IAM Console and Sign-in Page, Identities, Access Management, Troubleshooting IAM, Reference (with sub-links for IAM Identifiers, Limits, and Services That Work with IAM), and Policy Reference (with sub-links for JSON Element Reference, Policy Evaluation Logic, and Policy Grammar). The main content area has a title 'IAM JSON Policy Reference' with PDF, Kindle, and RSS links. It includes sections for detailed syntax, descriptions, and examples of elements, and a list of sections such as IAM JSON Policy Elements Reference, Policy Evaluation Logic, Grammar of the IAM JSON Policy Language, AWS Managed Policies for Job Functions, AWS Global Condition Context Keys, and Actions, Resources, and Condition Keys for AWS Services. The 'Actions, Resources, and Condition Keys for AWS Services' section is highlighted with a yellow border.

IAM JSON Policy Reference

PDF | Kindle | RSS

This section presents detailed syntax, descriptions, and examples of the elements used in IAM policies.

This reference includes the following sections.

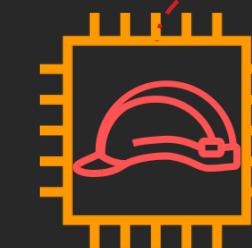
- [IAM JSON Policy Elements Reference](#) — Learn more about the elements and how they are used in various services.
- [Policy Evaluation Logic](#) — This section describes AWS requests, how the logic is evaluated, and how it applies to IAM policies.
- [Grammar of the IAM JSON Policy Language](#) — This section presents a formal grammar for the IAM JSON Policy Language.
- [AWS Managed Policies for Job Functions](#) — This section lists all the AWS managed policies for job functions that are needed to carry out the tasks expected of someone in a specific IAM role.
- [AWS Global Condition Context Keys](#) — This section includes a list of all global condition context keys.
- [Actions, Resources, and Condition Keys for AWS Services](#) — This section includes a list of specific condition keys that can be used to further refine the request.

Reading the IAM documentation page

Action	Resource	Condition
GetItem	The GetItem operation returns a set of attributes for the item with the given primary key	Read <code>table*</code> <code>dynamodb:Attributes</code> <code>dynamodb:EnclosingOperation</code> <code>dynamodb:LeadingKeys</code> <code>dynamodb:ReturnConsumedCapacity</code> <code>dynamodb:Select</code>

Reference: https://docs.aws.amazon.com/IAM/latest/UserGuide/list_amazondynamodb.html

Example 1: Read data from DynamoDB



EC2 instance
with IAM role

Read/write
data

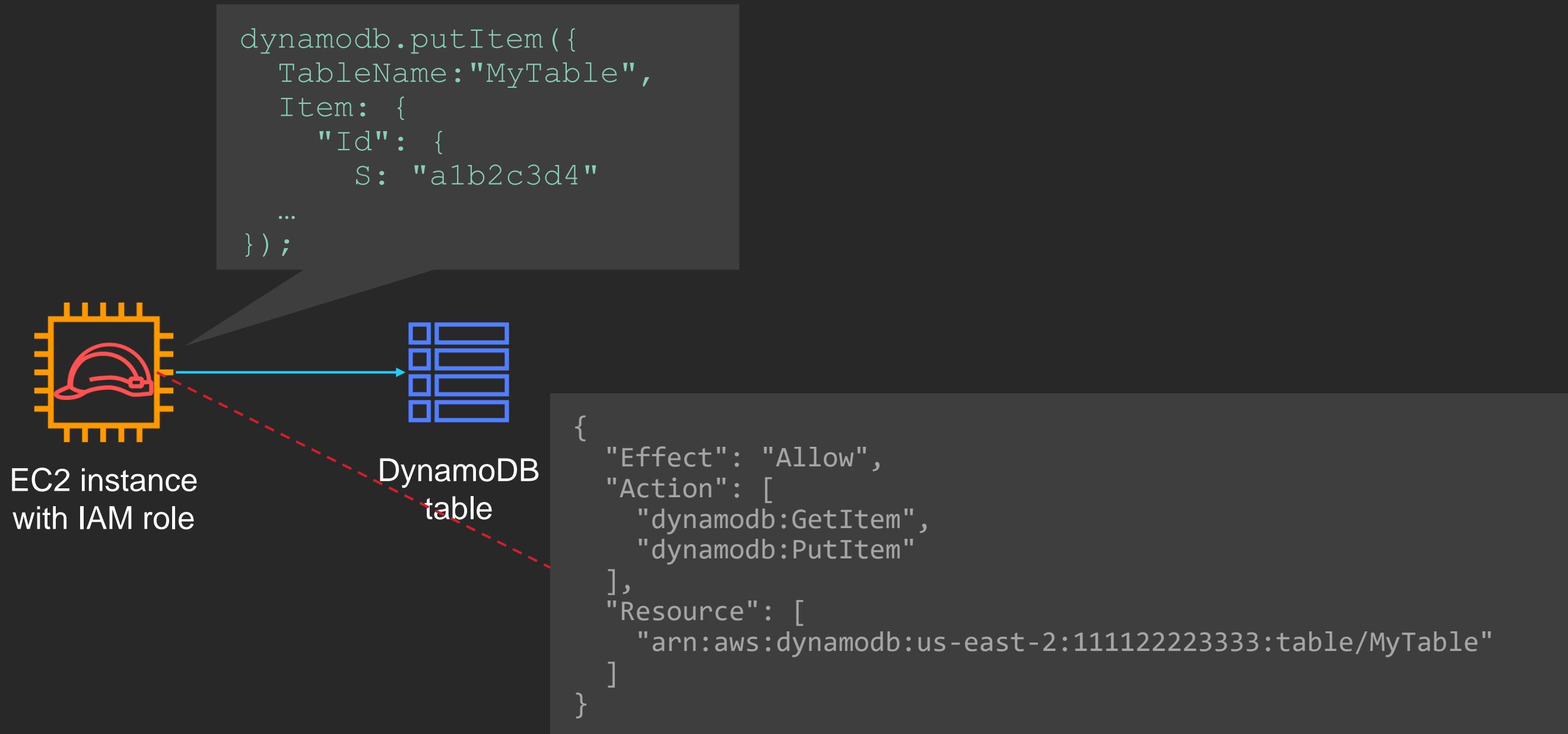
DynamoDB
table

```
{  
  "Effect": "Allow",  
  "Action": [  
    "dynamodb:GetItem",  
    "dynamodb:PutItem"  
  ],  
  "Resource": [  
    "arn:aws:dynamodb:us-east-2:111122223333:table/MyTable"  
  ]  
}
```

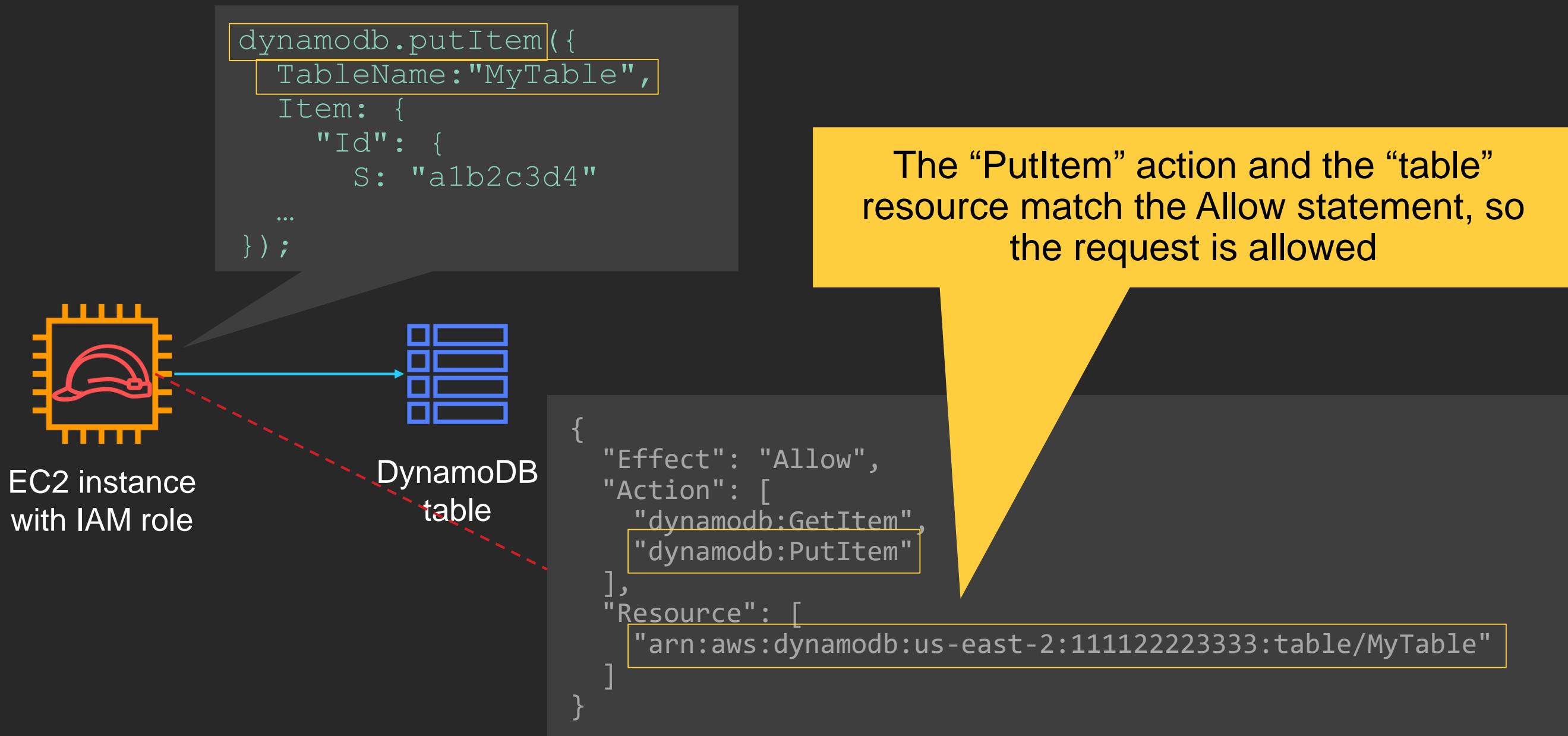
DynamoDB row:
Reading/writing actions

Specific resource: Table

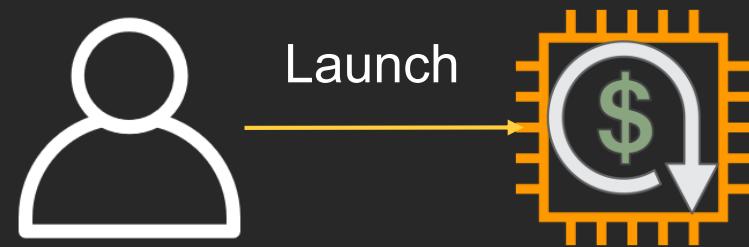
How authorization works in AWS



How authorization works in AWS



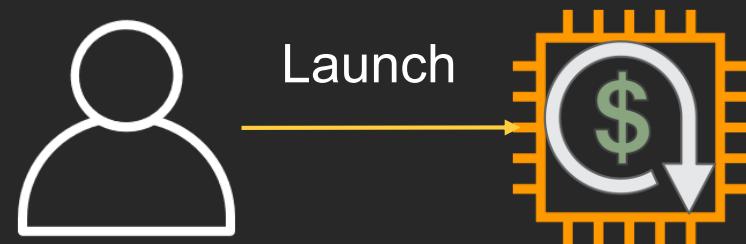
Example 2: Launch specific EC2 instance types



```
{  
    "Effect": "Allow",  
    "Action": "ec2:RunInstances",  
    "Resource": "*"  
}
```

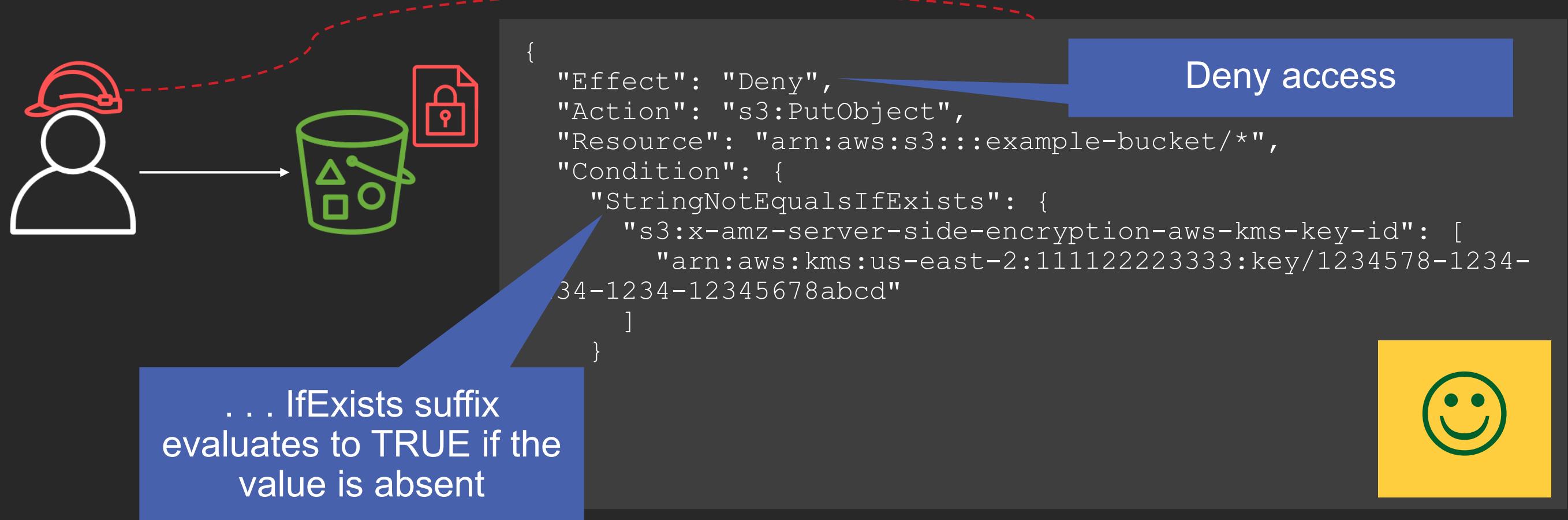
Is this policy good?

Example 2: Launch specific EC2 instance types



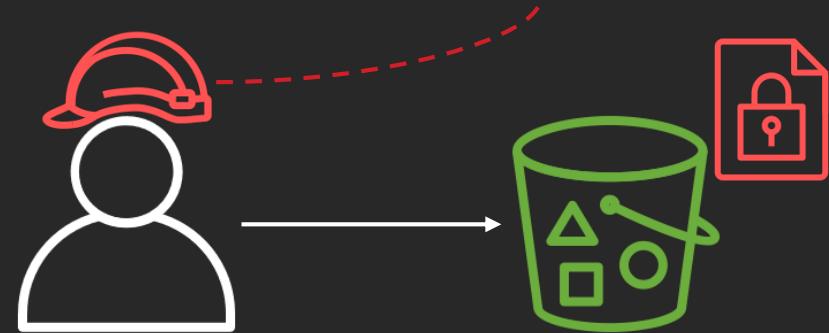
```
{  
    "Effect": "Allow",  
    "Action": "ec2:RunInstances",  
    "Resource": "*",  
    "Condition": {  
        "StringEquals": {  
            "ec2:InstanceType": [  
                "t3.nano",  
                "t3.micro"  
            ]  
        }  
    }  
}
```

Example 3: Prevent unencrypted writes to S3



Can I write my encrypted data?

Example 3: Prevent unencrypted writes to S3

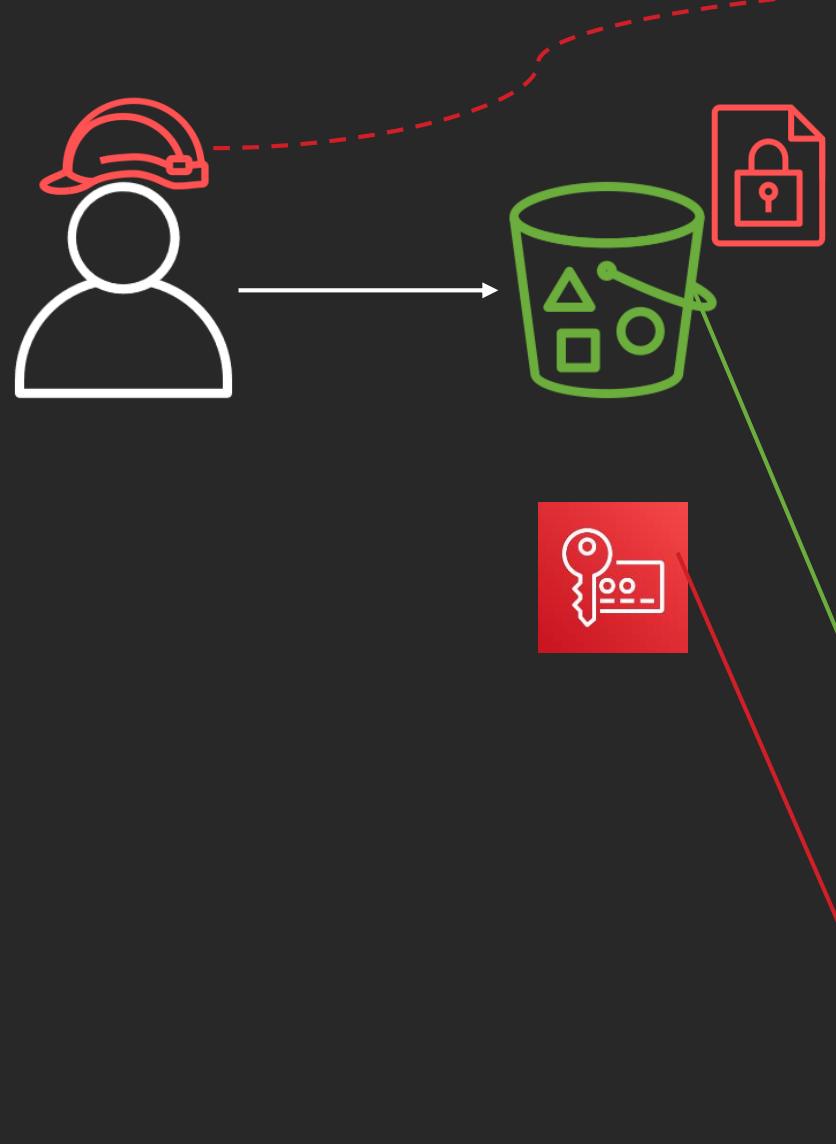


```
{  
    "Effect": "Deny",  
    "Action": "s3:PutObject",  
    "Resource": "arn:aws:s3:::example-bucket/*",  
    "Condition": {  
        "StringNotEqualsIfExists": {  
            "s3:x-amz-server-side-encryption-aws-kms-key-id": [  
                "arn:aws:kms:us-east-2:111122223333:key/1234578-1234-  
                1234-1234-12345678abcd"  
            ]  
        }  
    }  
,  
{  
    "Effect": "Allow",  
    "Action": [ "s3:GetObject", "s3:PutObject" ],  
    "Resource": "arn:aws:s3:::example-bucket/*"  
}
```



Can I write my encrypted data?

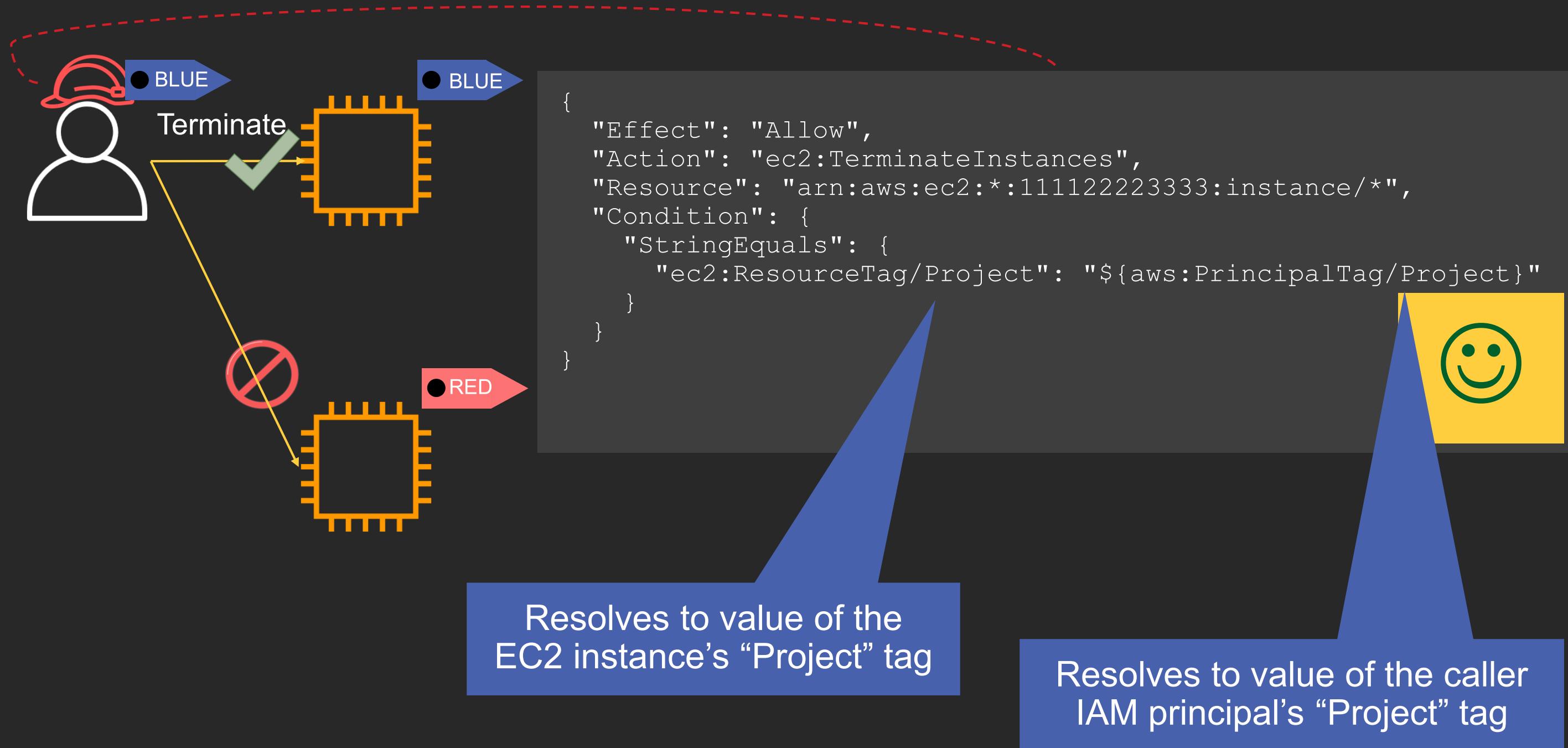
Example 3: Prevent unencrypted writes to S3



```
{  
    "Effect": "Deny",  
    "Action": "s3:PutObject",  
    "Resource": "arn:aws:s3:::example-bucket/*",  
    "Condition": {  
        "StringNotEqualsIfExists": {  
            "s3:x-amz-server-side-encryption-aws-kms-key-id": [  
                "arn:aws:kms:us-east-2:111122223333:key/12345678-  
                1234-1234-1234-12345678abcd"  
            ]  
        }  
    },  
    {  
        "Effect": "Allow",  
        "Action": [ "s3:GetObject", "s3:PutObject" ],  
        "Resource": "arn:aws:s3:::example-bucket/*"  
    },  
    {  
        "Effect": "Allow",  
        "Action": [ "kms:GenerateDataKey*", "kms:Decrypt" ],  
        "Resource": "arn:aws:kms:us-east-  
        2:111122223333:key/12345678-1234-1234-1234-12345678abcd"  
    }  
}
```

A yellow box with a green smiley face is positioned in the top right corner of the code area, indicating a successful outcome or best practice.

Example 4: Terminate instances in your project



Summary of what just happened

- AWS authorizations perform string-matching, and you need to find:
 - A matching Allow statement
 - No matching Deny statements
- Each resource for an action get authorized, and all must be allowed
- Many resources in AWS offer additional authorization conditions
- Tags can be used in IAM policy for authorization

Accessing AWS resource beyond your AWS account

Accessing resources in another AWS account



AWS Account: ★
MASTER

AWS Organization



```
{  
  "Effect": "Allow",  
  "Action": [  
    "s3>ListBucket",  
    "s3GetObject"  
  ],  
  "Resource": [  
    "arn:aws:s3:::example-bucket",  
    "arn:aws:s3:::example-bucket/*"  
  ]  
}
```



AWS Account
111122223333



AWS Account
444455556666



example-bucket

Accessing resources in another AWS account



AWS Account: ★

Resource-based policy for S3 bucket

```
{  
  "Effect": "Allow",  
  "Action": [  
    "s3>ListBucket",  
    "s3GetObject"  
  ],  
  "Resource": [  
    "arn:aws:s3:::example-bucket",  
    "arn:aws:s3:::example-bucket/*"  
  ]  
}
```

AWS

{

```
  "Effect": "Allow",  
  "Principal": {  
    "AWS": "111122223333"  
  },  
  "Action": [  
    "s3>ListBucket",  
    "s3GetObject"  
  ],  
  "Resource": [  
    "arn:aws:s3:::example-bucket",  
    "arn:aws:s3:::example-bucket/*"  
  ]  
}
```



AWS Account
111122223333



AWS Account
444455556666



example-bucket



Accessing resources in another AWS account



AWS Account: ★
MASTER

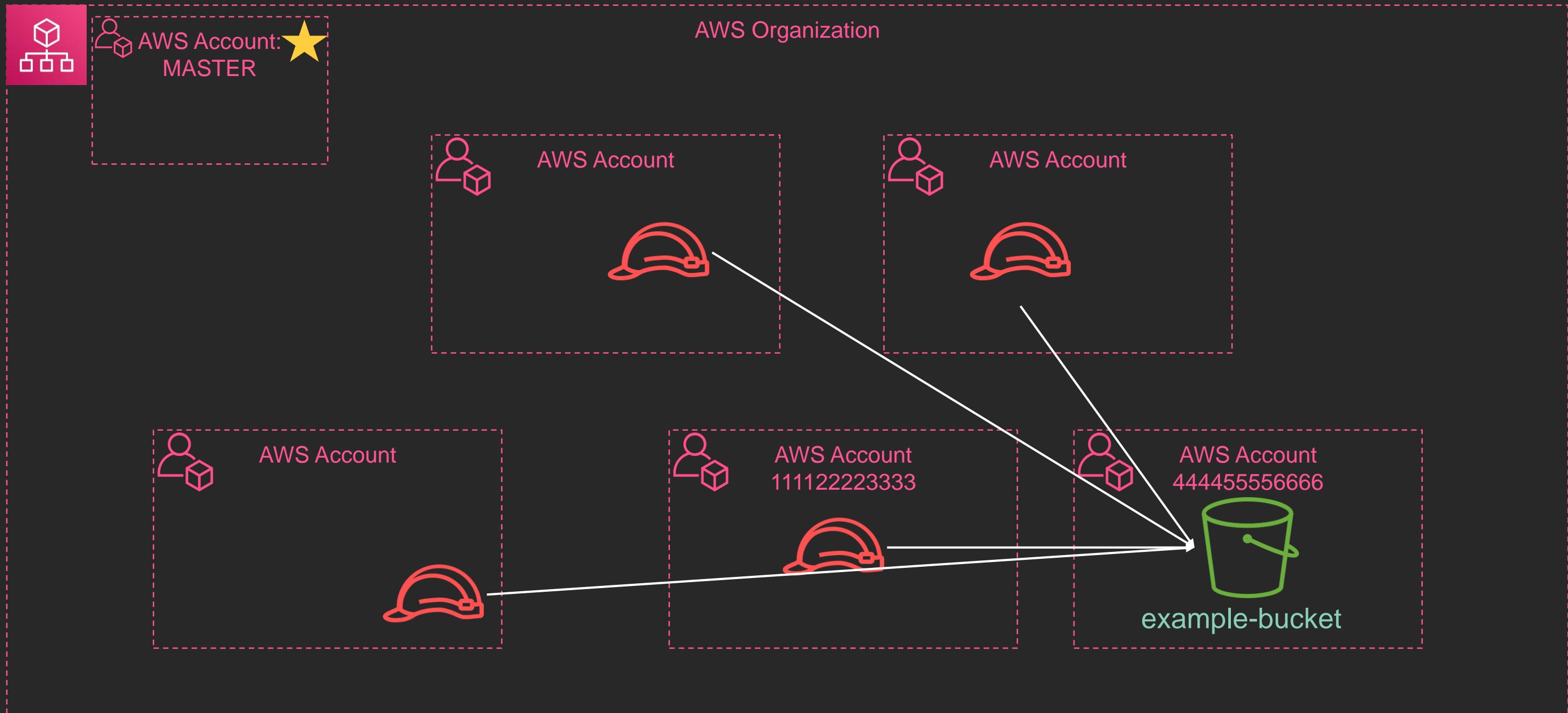
Allow the other account

```
{  
  "Effect": "Allow",  
  "Action": [  
    "s3>ListBucket",  
    "s3GetObject"  
  ],  
  "Resource": [  
    "arn:aws:s3:::example-bucket",  
    "arn:aws:s3:::example-bucket/*"  
  ]  
}
```

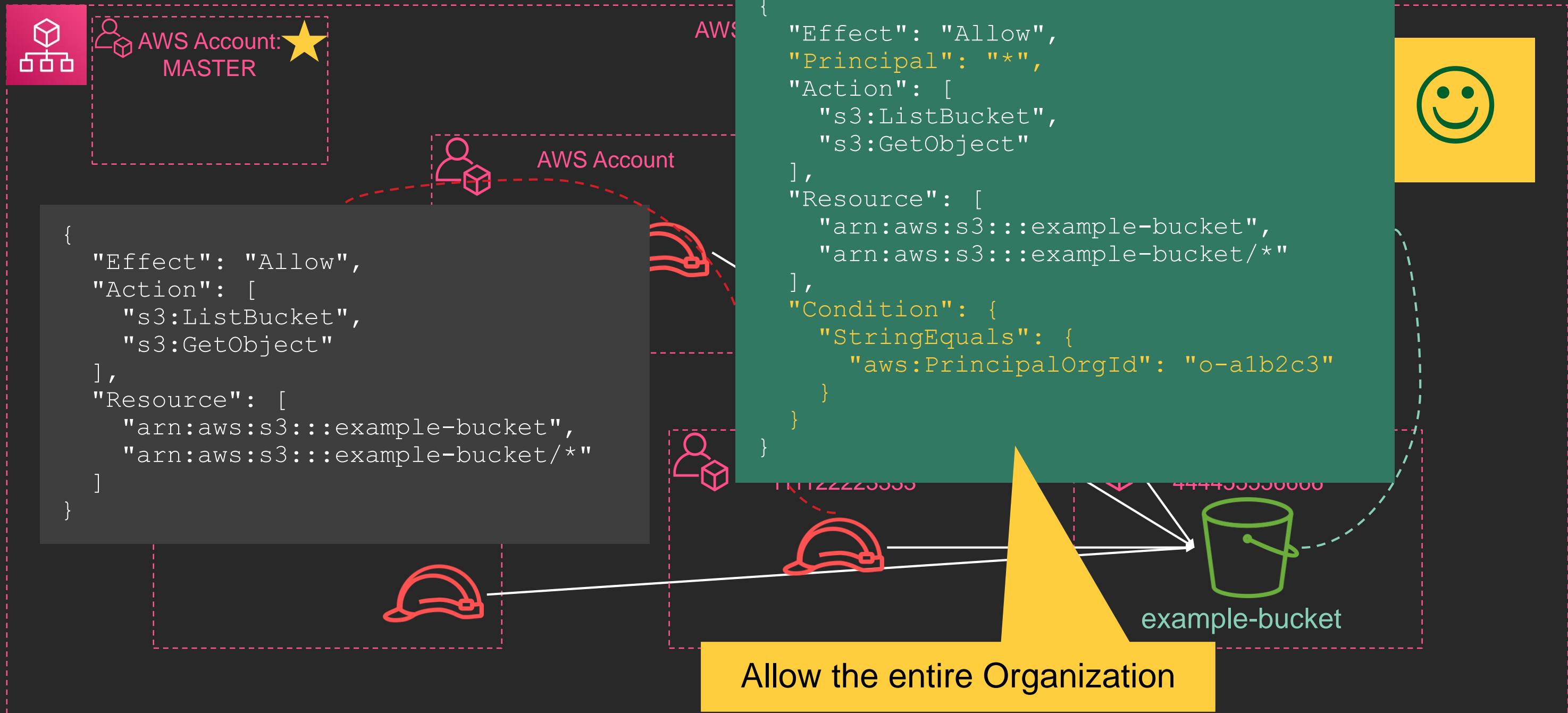
```
{  
  "Effect": "Allow",  
  "Principal": {  
    "AWS": "111122223333"  
  },  
  "Action": [  
    "s3>ListBucket",  
    "s3GetObject"  
  ],  
  "Resource": [  
    "arn:aws:s3:::example-bucket",  
    "arn:aws:s3:::example-bucket/*"  
  ]  
}
```



Accessing resources in another AWS account



Accessing resources in another AWS account

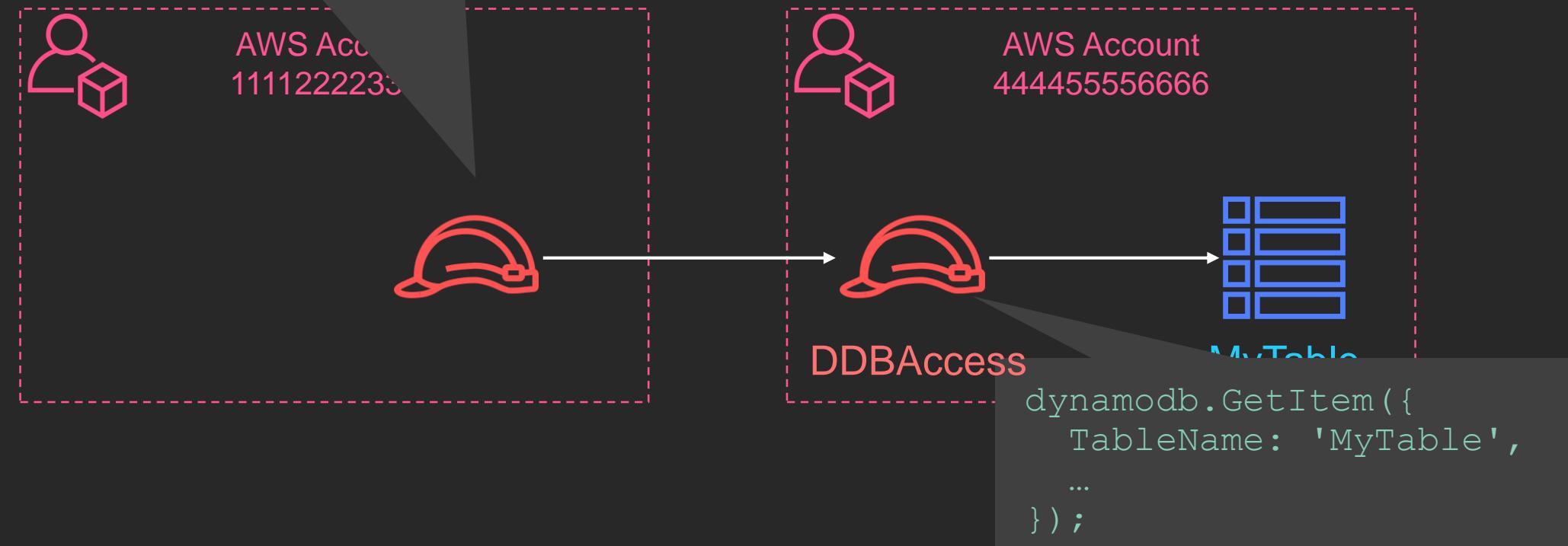


Accessing resources in another AWS account



Accessing resources in another AWS account: Using IAM roles across accounts

```
sts.AssumeRole({  
    RoleArn: 'arn:aws:iam::44445556666:role/DDBAccess',  
    ...  
});
```



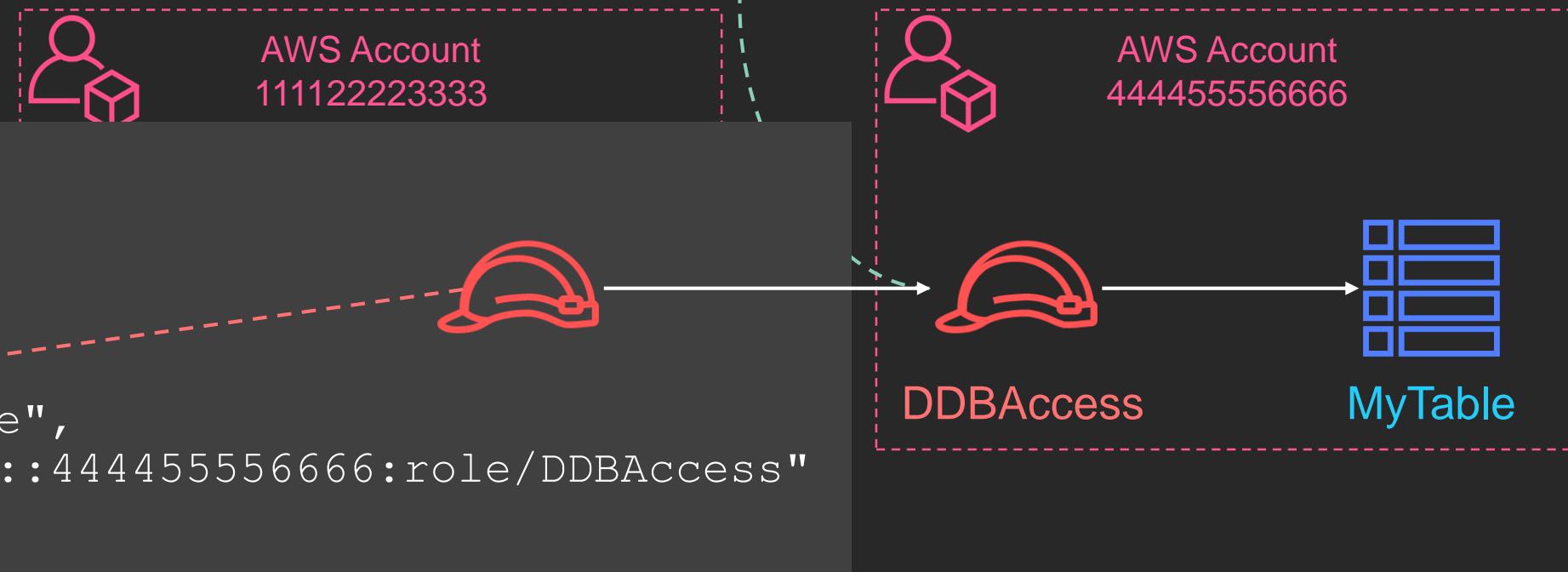
Accessing resources in another AWS account: Using IAM roles across accounts

IAM role trust policy:

```
{  
  "Effect": "Allow",  
  "Principal": {  
    "AWS": "111122223333"  
  },  
  "Action": "sts:AssumeRole",  
}
```

Principal policy:

```
{  
  "Effect": "Allow",  
  "Action": "sts:AssumeRole",  
  "Resource": "arn:aws:iam::444455556666:role/DDBAccess"  
}
```

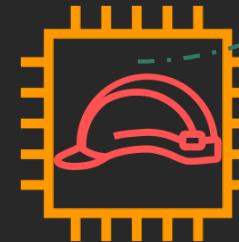


Recommendations for cross-account access

Keep it simple:

- Use resource-based policies when available
- Unless you have a specific reason to do otherwise:
 - Trust the entire other account, or
 - Trust the AWS Organization
- Use IAM roles if resource-based policies are not available
 - Follow the above rules for their trust policies (i.e., resource-based policies for IAM roles)

Role trust policy for aws services



EC2 instance
with IAM role

IAM role trust policy:

```
{  
  "Effect": "Allow",  
  "Principal": {  
    "Service": "ec2.amazonaws.com"  
  },  
  "Action": "sts:AssumeRole",  
}
```

The EC2 service can assume
this role and vend short-term
credentials to your application

New: Access Analyzer

The screenshot shows the AWS IAM Access Analyzer interface. On the left, the navigation menu includes Identity and Access Management (IAM), Dashboard, Access management (Groups, Users, Roles, Policies), Identity providers, Account settings, and Access reports (Access analyzer, Archive rules, Analyzer details). The Access analyzer section is currently selected. At the bottom of the sidebar, the AWS account ID is listed as 267276985313.

The main content area displays the Access Analyzer dashboard. It features two large sections: "Resource" on the left and "External principal" on the right. The "Resource" section shows an S3 Bucket named "my-bucket-3". The "External principal" section shows an AWS Account with the ID "111222333444". Below these sections is a table titled "Active findings".

<input type="checkbox"/>	Finding ID	Resource	External principal	Condition	Access level	Updated
<input type="checkbox"/>	24e55490-b4a3-43a2-8674-1cd8c0...	S3 Bucket my-bucket-3	AWS Account 111222333444	-	Write, Permissions	a day ago
<input type="checkbox"/>	331e7c5b-23b0-4025-b613-3aafc7...	S3 Bucket my-bucket-3	AWS Account 1234567889012	-	Write, Permissions	a day ago
<input type="checkbox"/>	623a7754-80d6-4c78-9ce7-12da21...	S3 Bucket my-bucket-4	IAM User 111222333444:user/admin	Source VPC vpc-abc	Write, Tagging, Read, List, Permissions	a day ago
<input type="checkbox"/>	ea39d435-ff1f-433e-989c-1c76a39...	IAM Role RoleForInternalPrincipals	AWS Account 901234567890	-	Write	2 days ago

IAM pro move: Guardrails with Organizations

Guardrails for IAM: Organizations service control policies



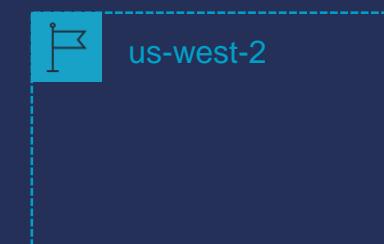
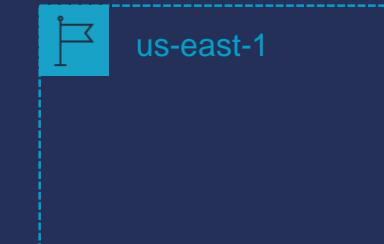
AWS Account: ★
MASTER



Organizational Unit

AWS Organization

Expected Regions



AWS Account



AWS Account



AWS Account



Guardrails for IAM: Organizations service control policies



AWS Account: ★
MASTER



Organizational Unit



AWS Account



AWS Account



AWS Organization

Expected Regions via
Organizations Service
Control Policy

```
{  
  "Effect": "Deny",  
  "Principal": "*",  
  "Action": "*",  
  "Resource": "*",  
  "Condition": {  
    "StringNotEquals": {  
      "aws:RequestedRegion": [  
        "us-east-1",  
        "us-west-2"  
      ]  
    }  
  }  
}
```

Guardrails for IAM: Organizations service control policies



AWS Account: ★
MASTER



Organizational Unit



AWS Organization



AWS Account



Expected Regions via
Organizations Service
Control Policy

```
{  
  "Effect": "Deny",  
  "Principal": "*",  
  "Action": "*",  
  "Resource": "*",  
  "Condition": {  
    "StringNotEquals": {  
      "aws:RequestedRegion": [  
        "us-east-1",  
        "us-west-2"  
      ]  
    }  
  }  
}
```

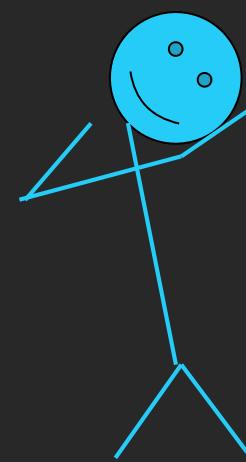
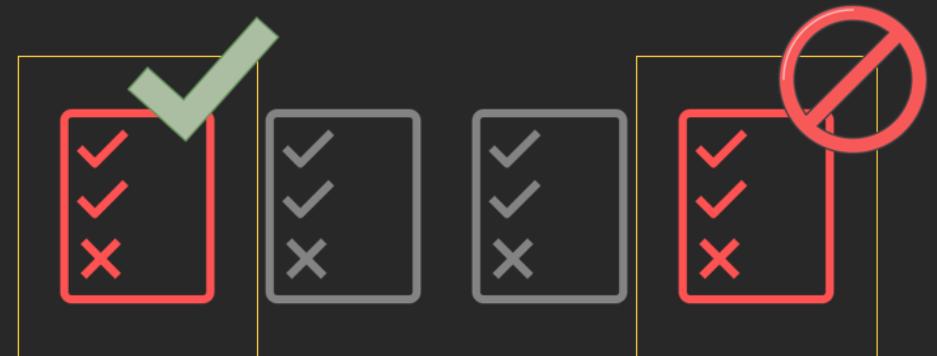
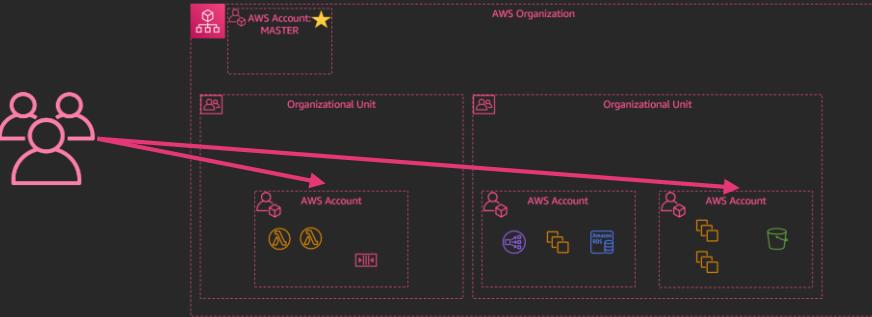
Let's try SCP

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "Statement1",  
            "Effect": "Deny",  
            "Action": [  
                "*"  
            ],  
            "Resource": [  
                "*"  
            ],  
            "Condition": {  
                "StringNotEquals": {  
                    "aws:RequestedRegion": [  
                        "ap-south-1"  
                    ]  
                }  
            }  
        }  
    ]  
}
```

Wrapping up

IAM: Getting in and getting around

- Patterns for human access to an AWS environment
- How an authorization decision is made in AWS
- How to construct a great IAM policy



```
{  
  "Effect": "Allow",  
  "Action": [  
    "s3:GetObject"  
  ],  
  "Resource": "arn:aws:s3:::example-bucket/*",  
  "Condition": {  
    "StringEquals": {  
      ...  
    }  
  }  
}
```

Thank you!

Chetan Agrawal

agrcheta@amazon.com

Deven Suri

dsuri@amazon.com