

AWS re:Inforce

JUNE 13 - 14, 2023 | ANAHEIM, CA

I A M 2 0 1

A first-principles approach: AWS Identity and Access Management (IAM)

Becky Weiss

Senior Principal Engineer
Amazon Web Services



Agenda: Practical skills ++

- The **what** of IAM: Skills for builders – Learn how to interpret and write good IAM policies
- The **why** of IAM: Working backward from customer needs over time
- The **how** of IAM: What happens in an authorization

Agenda: Practical skills ++



- The **what** of IAM: Skills for builders – Learn how to interpret and write good IAM policies
- The **why** of IAM: Working backward from customer needs over time
- The **how** of IAM: What happens in an authorization

The backstory



AWS existed before IAM



Me

Log in with email
and password
("root credentials")



AWS account



Cloud infrastructure that I own

Launch of IAM: 2011

AWS Identity and Access Management (IAM) Announces General Availability and Support in the AWS Management Console

Posted On: May 3, 2011

We are excited to announce the availability of AWS Identity and Access Management (IAM) in the AWS Management Console. You can now add users to your AWS Account, set groups and permissions for these users, and enable users to call AWS Service APIs, all from a simple point-and-click browser interface.

To learn more about the IAM console, please visit the [Using IAM guide](#) or open the [IAM console](#).

AWS without IAM: Me and my stuff



Me

Login with
email/password
("root credentials")

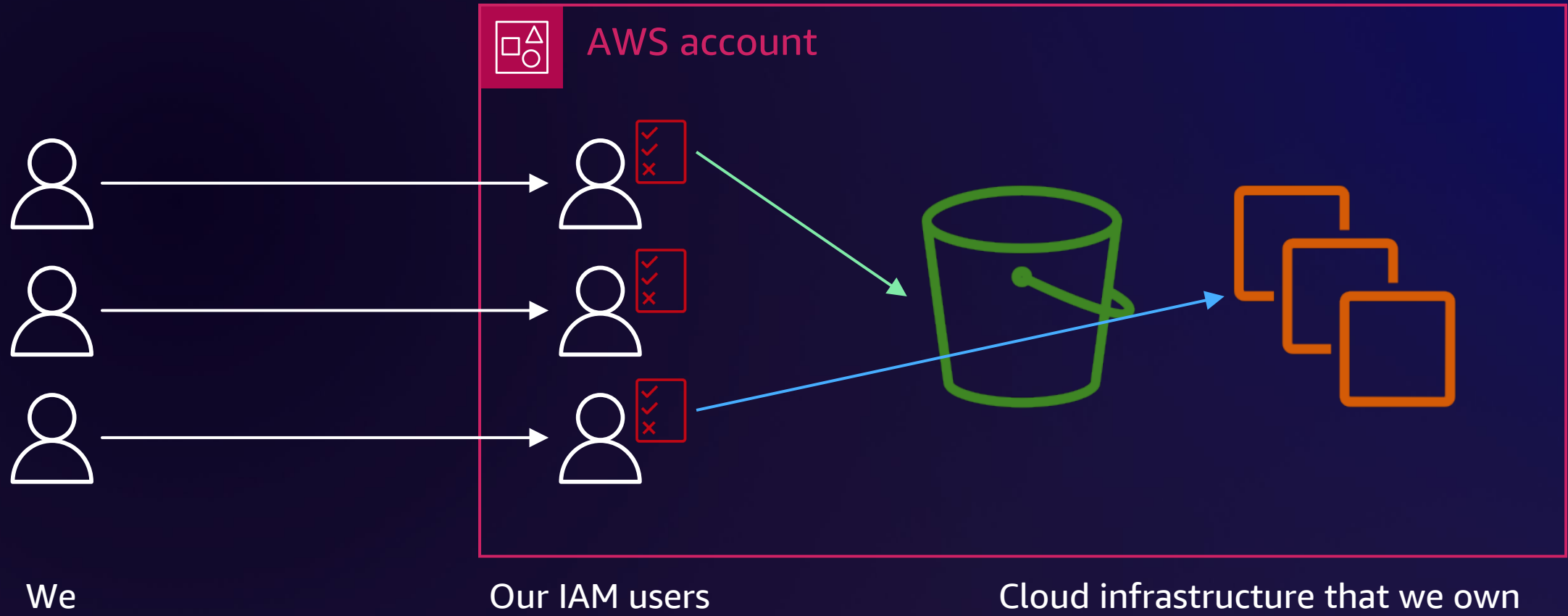


AWS account

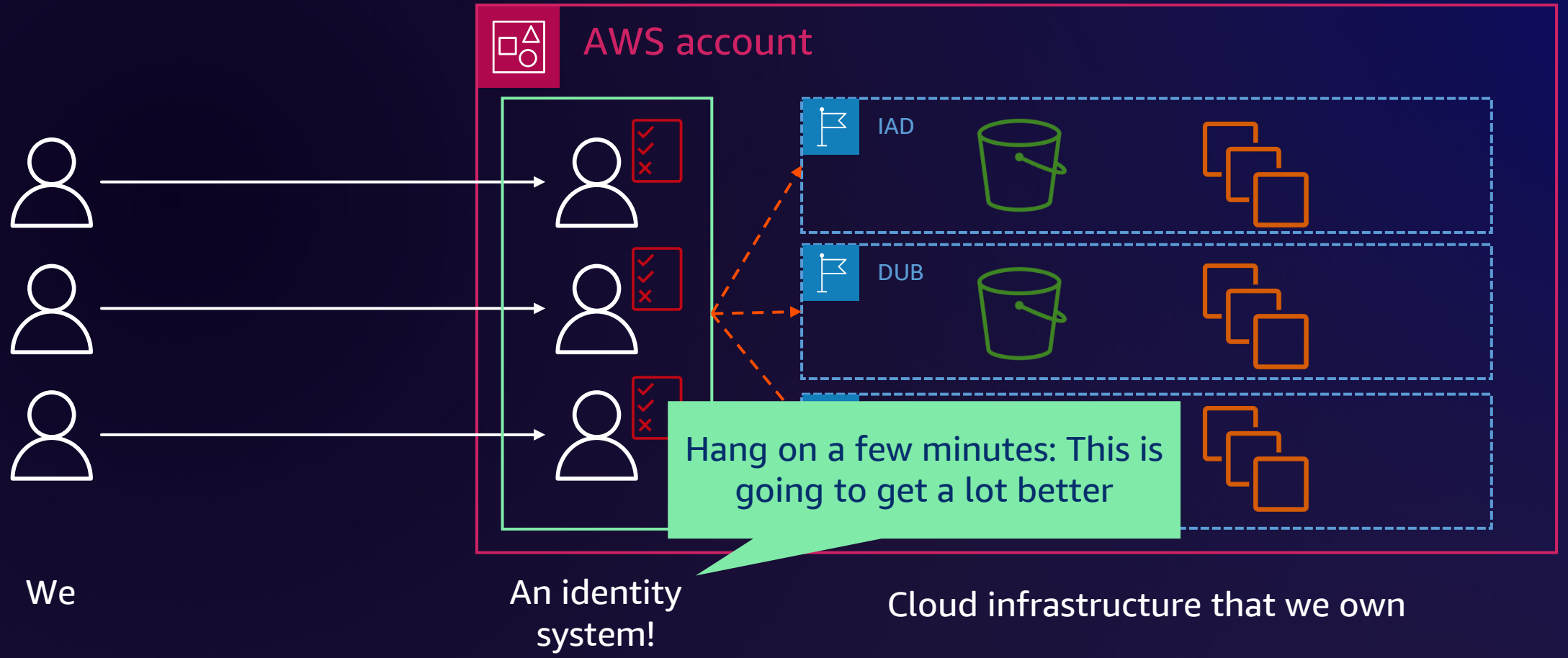


Cloud infrastructure that I own

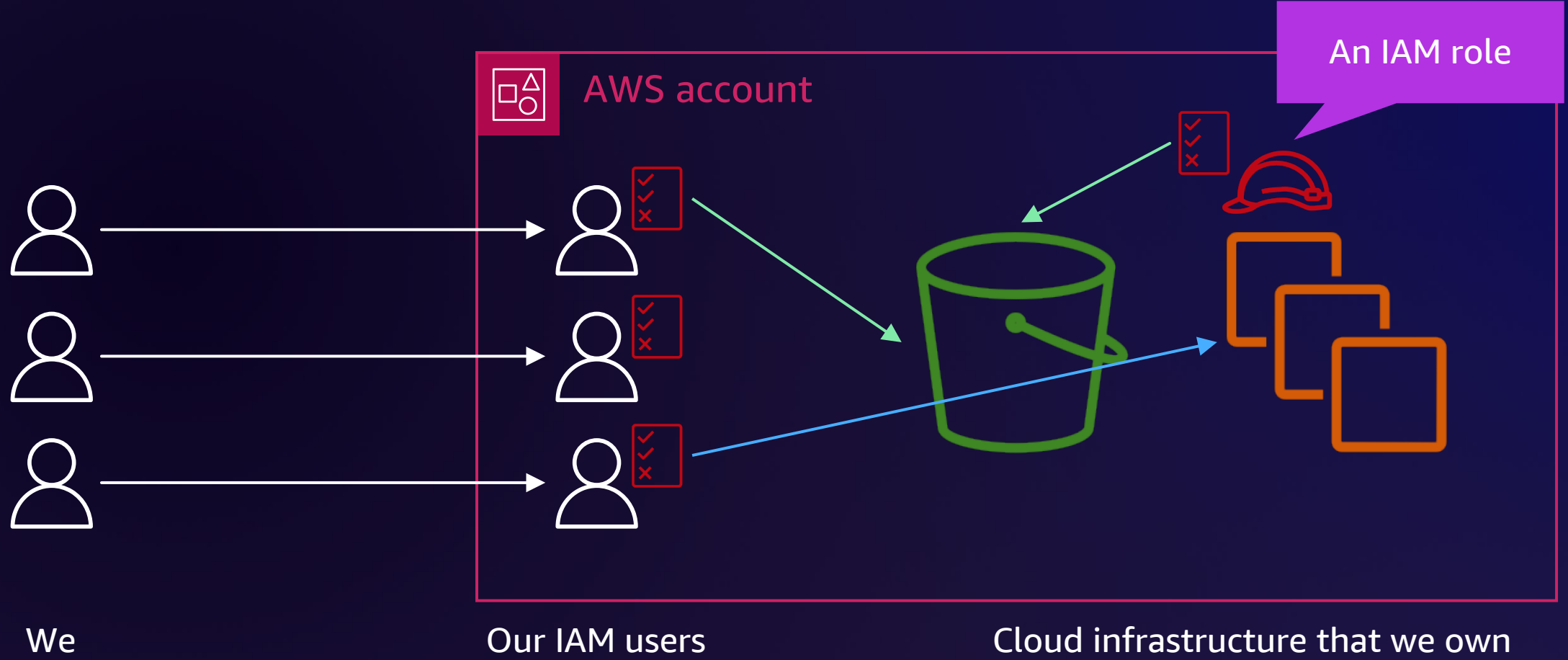
IAM circa 2011: We and our stuff



IAM circa 2011: An identity system for the cloud



IAM circa 2012: Identities for compute



Terminology: IAM principal

IAM principal: Entity registered in the IAM system

Appropriate for nearly all modern use cases

IAM roles: IAM principals that authenticate with temporary credentials

Use cases

- AWS compute environments (EC2 instances, Lambda functions, etc.)
- Non-AWS compute environments (IAM Roles Anywhere)
- Credentials for federated identities (e.g., users in a directory)

IAM users: IAM principals with long-term credentials

Principals and principles: Spell-check

Principal (n.)

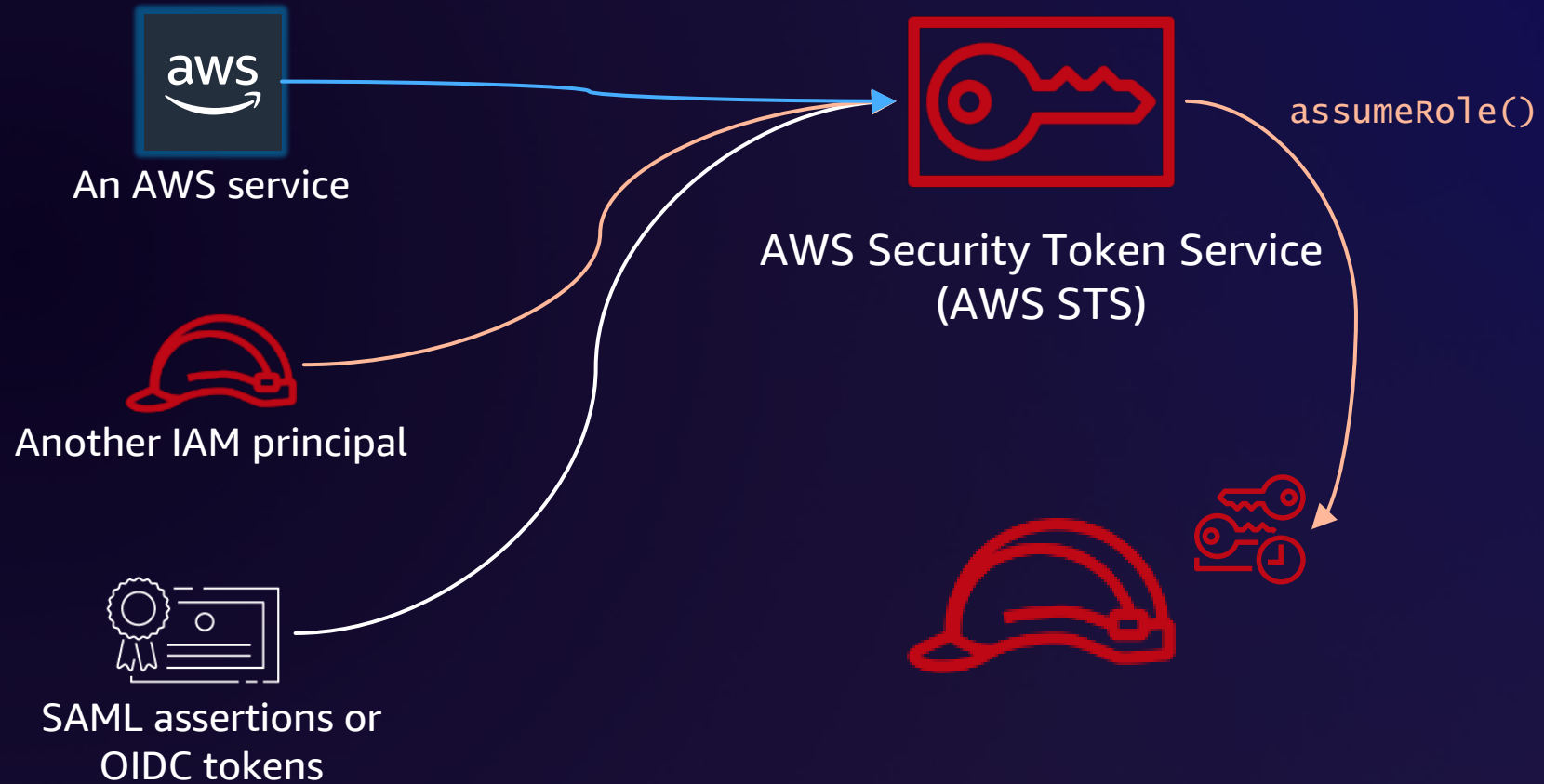


Principle (n.)



Principal principles: Getting IAM role credentials

IAM role credentials are temporary



How AWS services assume IAM roles

IAM > Roles > Create role

Step 1
Select trusted entity

Step 2
Add permissions

Step 3
Name, review, and create

Select trusted entity [Info](#)

Trusted entity type

- AWS service**
Allow AWS services like EC2, Lambda, or others to perform actions in this account.
- AWS account**
Allow entities in other AWS accounts belonging to you or a 3rd party to perform actions in this account.
- SAML 2.0 federation**
Allow 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.

Use case

Allow an AWS service like EC2, Lambda, or others to perform actions in this account.

Common use cases

- EC2**
Allows EC2 instances to call AWS services on your behalf.
- Lambda**
Allows Lambda functions to call AWS services on your behalf.



AWS Lambda
service



Lambda function
runtime environment

How AWS services assume IAM roles

```
// Role Trust Policy, a.k.a. Assume-Role Policy.  
//  
// This policy is attached to the IAM role itself and  
// says who can obtain its credentials (the AWS Lambda  
// service in this case)  
  
{  
  "version": "2012-10-17",  
  "statement": [  
    {  
      "effect": "Allow",  
      "principal": {  
        "service": "lambda.amazonaws.com"  
      },  
      "action": "sts:AssumeRole"  
    }  
  ]  
}
```

Service principal:

Identifies an AWS service acting
on your resources

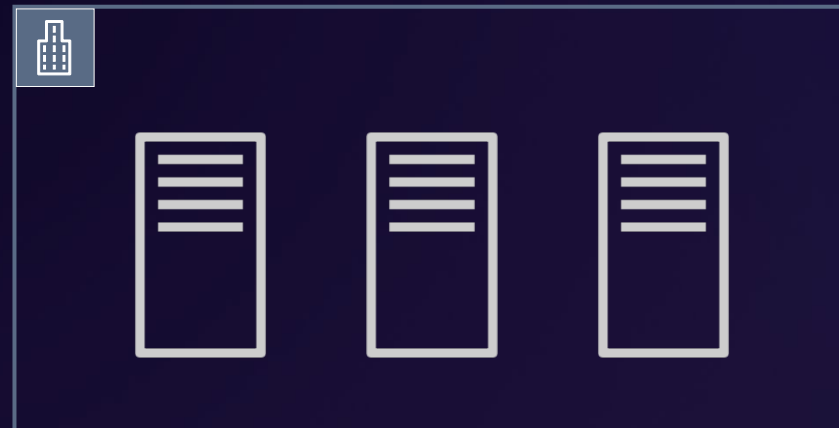


AWS Lambda
service



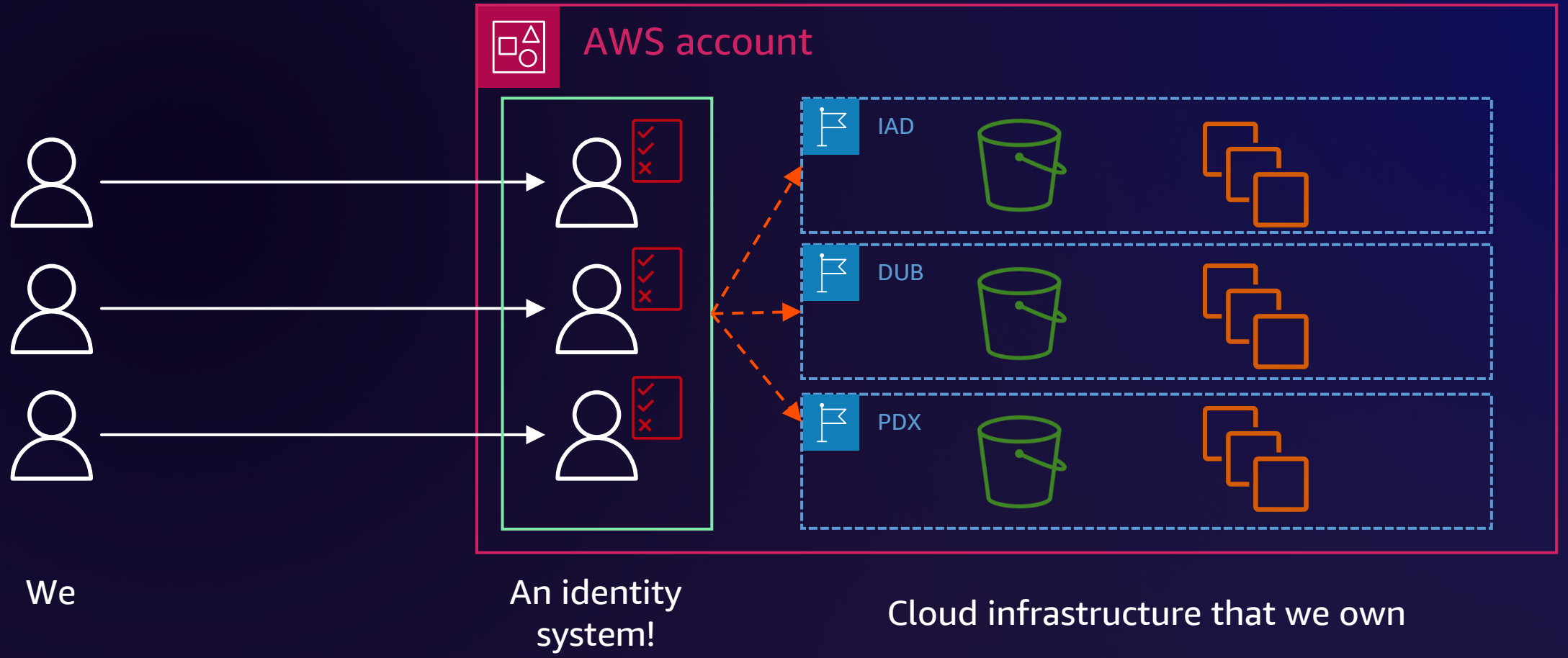
Lambda function
runtime environment

IAM Roles Anywhere: IAM roles for workloads outside AWS



IAM Roles Anywhere: IAM roles for workloads outside AWS

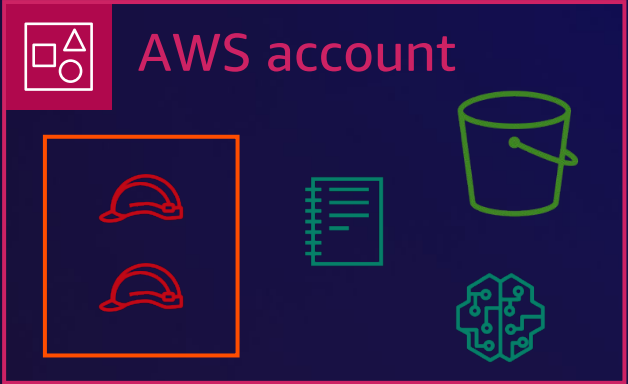






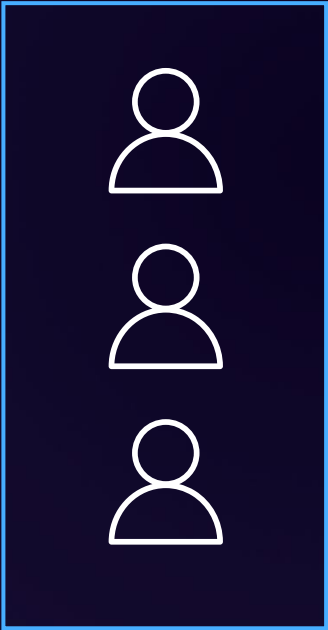
We

AWS IAM as the identity system inside a single AWS account

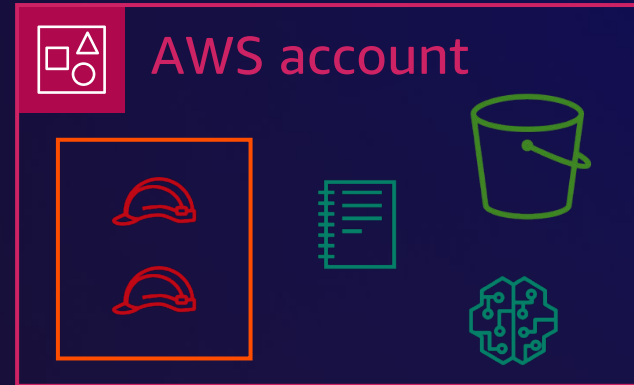


Cloud infrastructure that we own

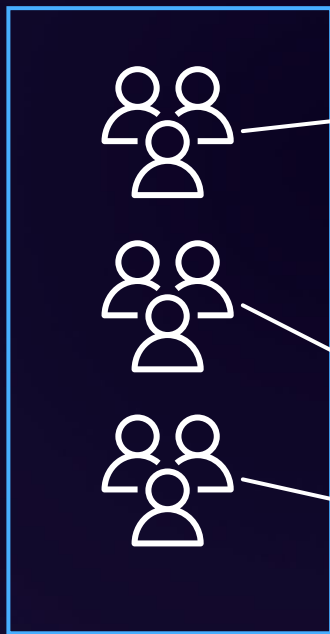




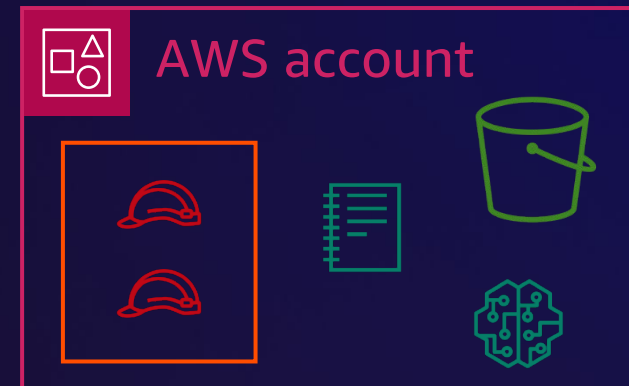
We:
Corp identity
provider



Cloud infrastructure that we own

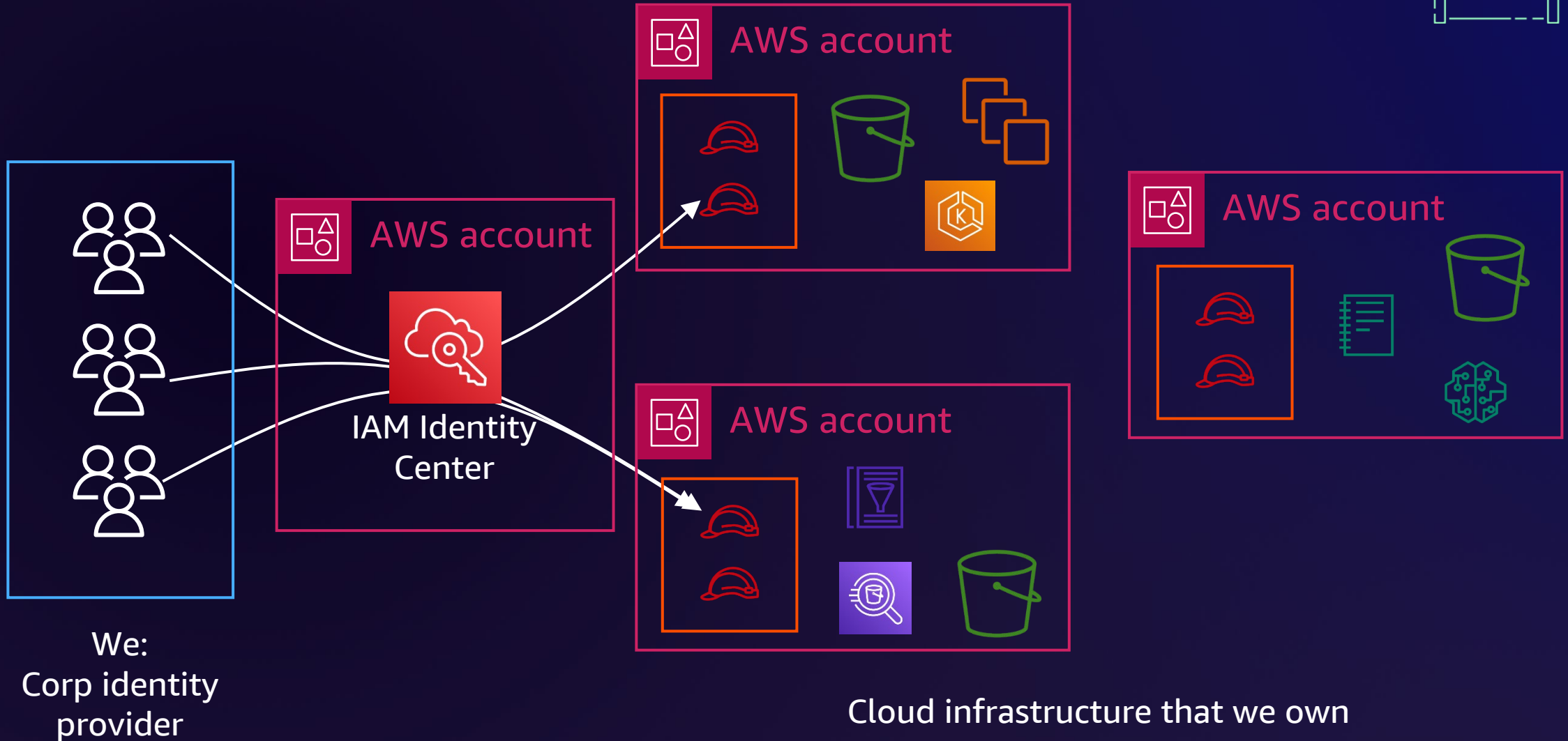


We:
Corp identity
provider

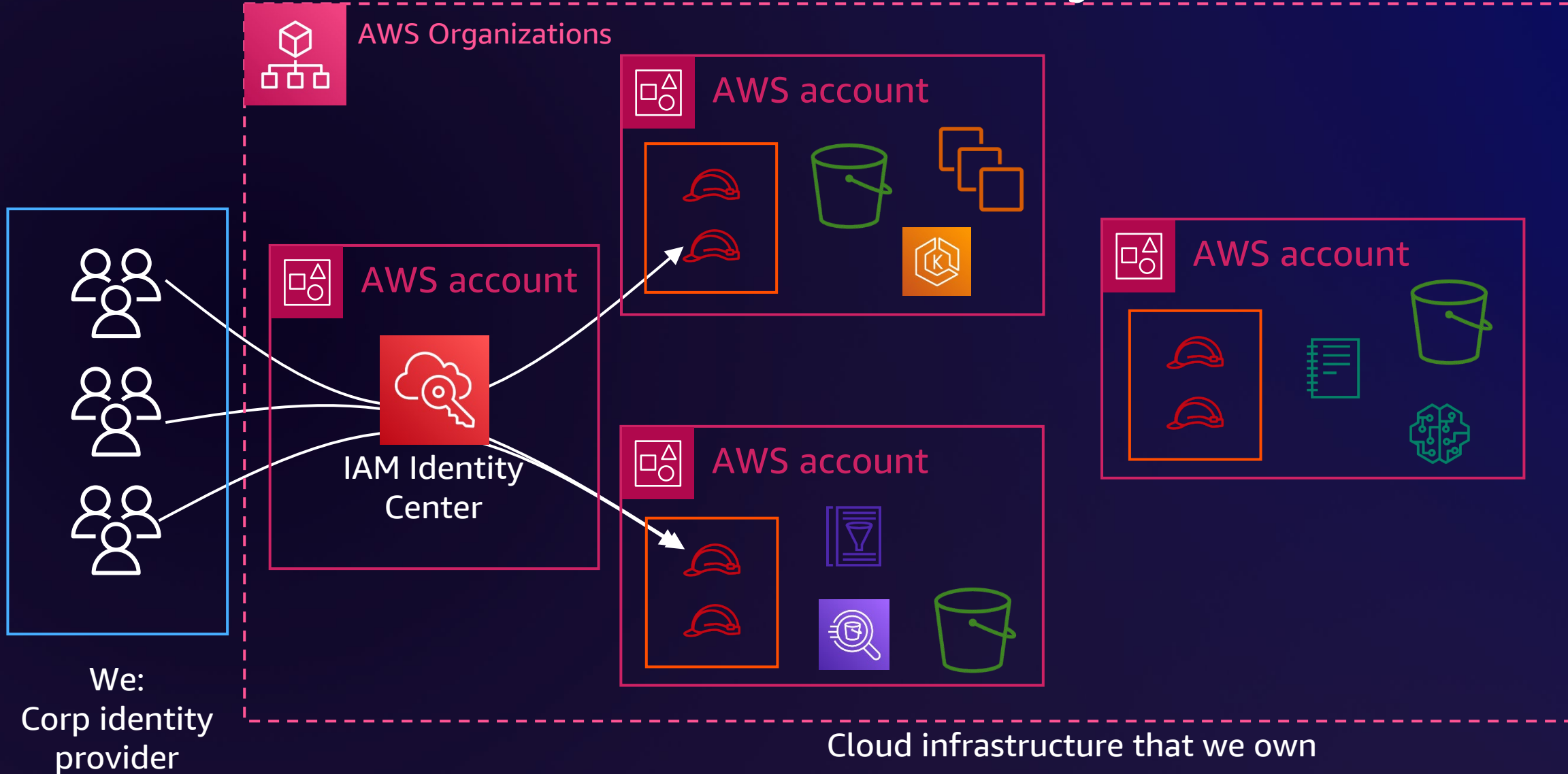


Cloud infrastructure that we own

User access to IAM roles: IAM Identity Center



User access to IAM roles: IAM Identity Center

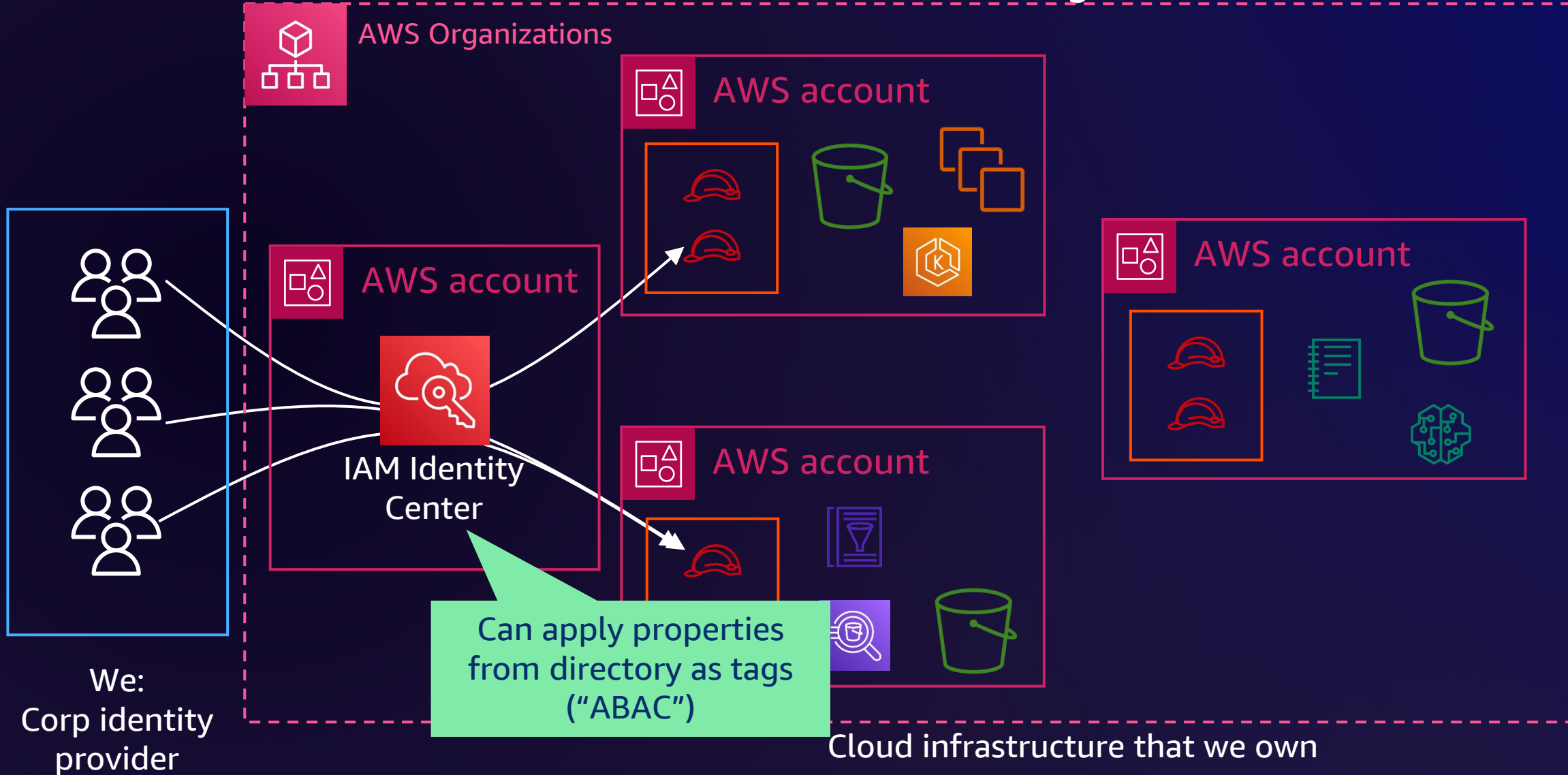


We:
Corp identity
provider

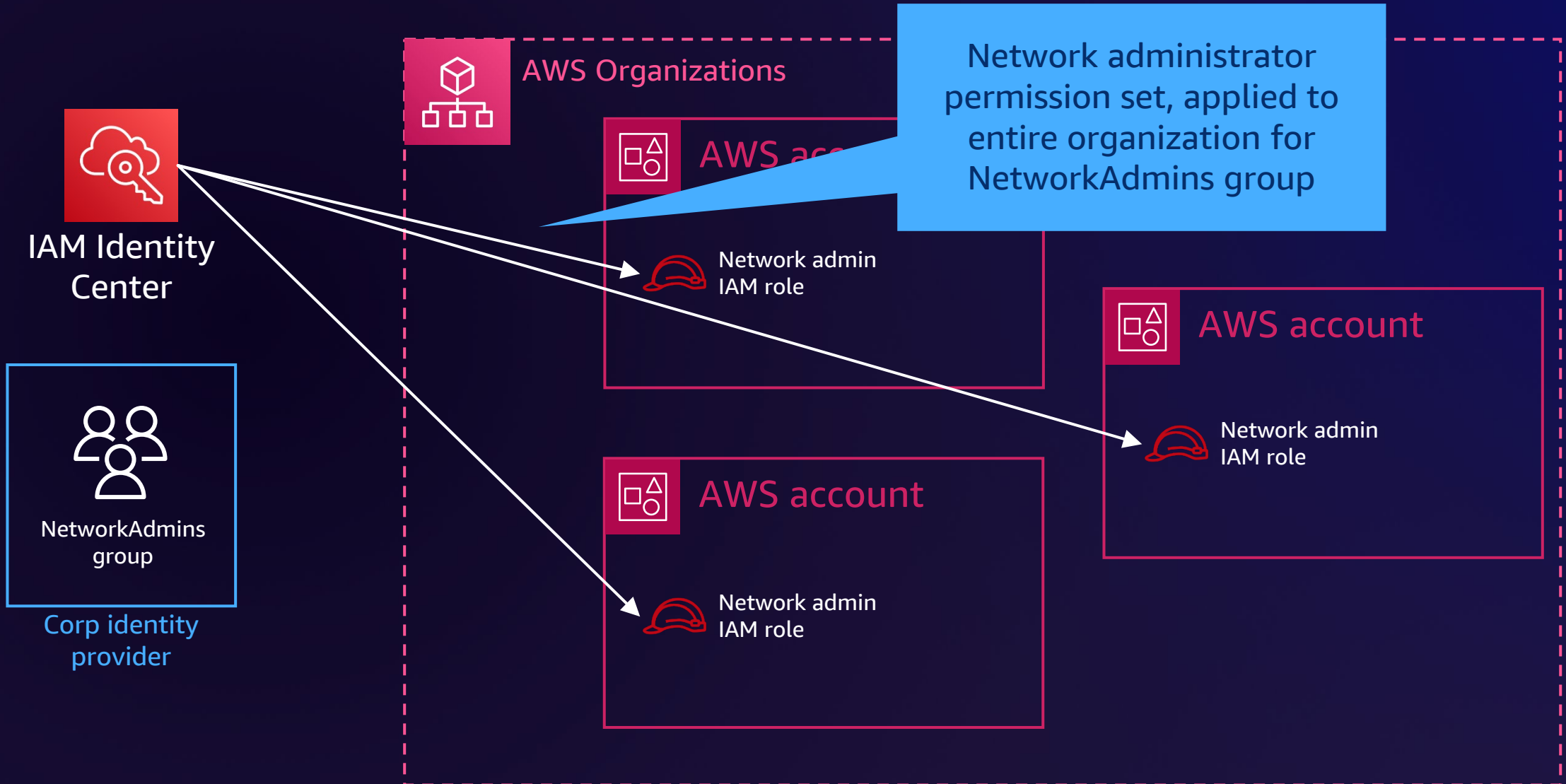
Cloud infrastructure that we own



User access to IAM roles: IAM Identity Center



IAM Identity Center permission sets



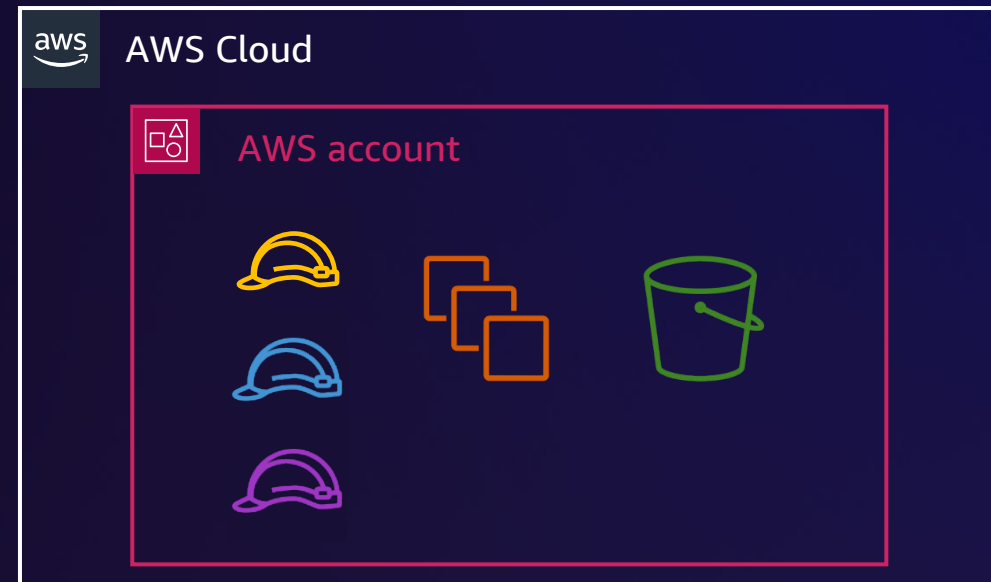
User access to AWS via IAM Identity Center

The image displays the AWS IAM Identity Center user interface. On the left, a sign-in form is visible with the AWS logo at the top. The form includes a 'Sign in' header, a 'Username:' field with the value 'bluejay@example.co', and a 'Password' field with masked characters. Below the password field is a 'Show password' checkbox and a blue button. In the center, a card shows an orange cube icon and the text 'AWS Account (1)'. To the right, a dark header bar displays the user's name 'DataScientist/bluejay' with a dropdown arrow. The main area shows the 'Console Home' page with a search bar and a 'Build a solution' section. This section contains ten quick-start options, each with an icon, a title, and a duration: 'Launch a virtual machine' (2 mins), 'Register a domain' (3 mins), 'Start a development project' (5 mins), 'Build a web app' (5 mins), 'Connect an IoT device' (5 mins), 'Deploy a serverless microservice' (2 mins), 'Build using virtual servers' (2 mins), 'Start migrating to AWS' (2 mins), 'Host a static web app' (2 mins), and 'Build SQL Server on AWS' (2 mins). A 'Reset to default layout' button and an 'Add widgets' button are also present.



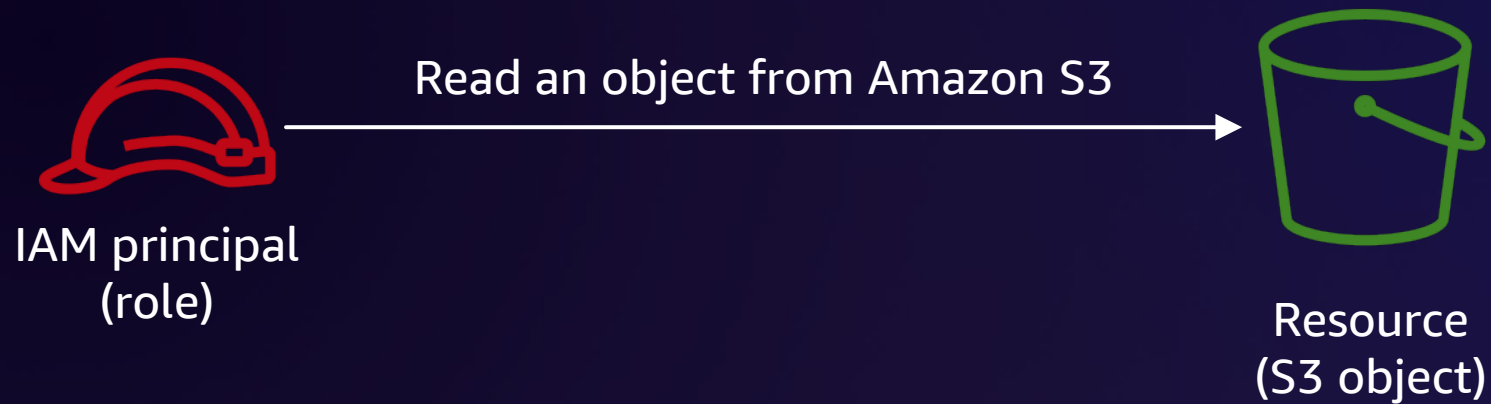
Principles of principals: Summary

- IAM roles for AWS compute environments
- IAM roles for non-AWS compute environments via IAM Roles Anywhere
- IAM roles for federated access by cloud operators (e.g., through IAM Identity Center)



Fundamentals of IAM: Authentication

Authentication example: A request to Amazon S3



Authentication example: A request to Amazon S3



IAM principal
(Lambda function role)

```
// AWS Javascript v3 SDK example
import {S3Client, GetObjectCommand} from "@aws-sdk/client-s3";

const s3Client = new S3Client({});

const obj = await s3Client.send(new GetObjectCommand({
  Bucket: 'example-bucket',
  Key: 'path/to/obj'
}));

// Object contents will be in obj.Body
```



Resource
(S3 object)

Authentication example: A request to Amazon S3



IAM principal
(Lambda function role)

```
// AWS Javascript V3 SDK example
import {S3Client, GetObjectCommand} from "@aws-sdk/client-s3";

const s3Client = new S3Client({});

const obj = await s3Client.send(new GetObjectCommand({
  Bucket: 'example-bucket',
  Key: 'path/to/obj'
}));

// Object contents will be in obj.Body
```

The AWS SDK typically knows where to find the right credentials, though you can also configure them manually here

Resource
(S3 object)

Authentication example: A request to Amazon S3



```
# AWS command-line interface (CLI)  
> aws s3api get-object --bucket example-bucket --key path/to/obj /tmp/obj.out
```



IAM
Identity Center

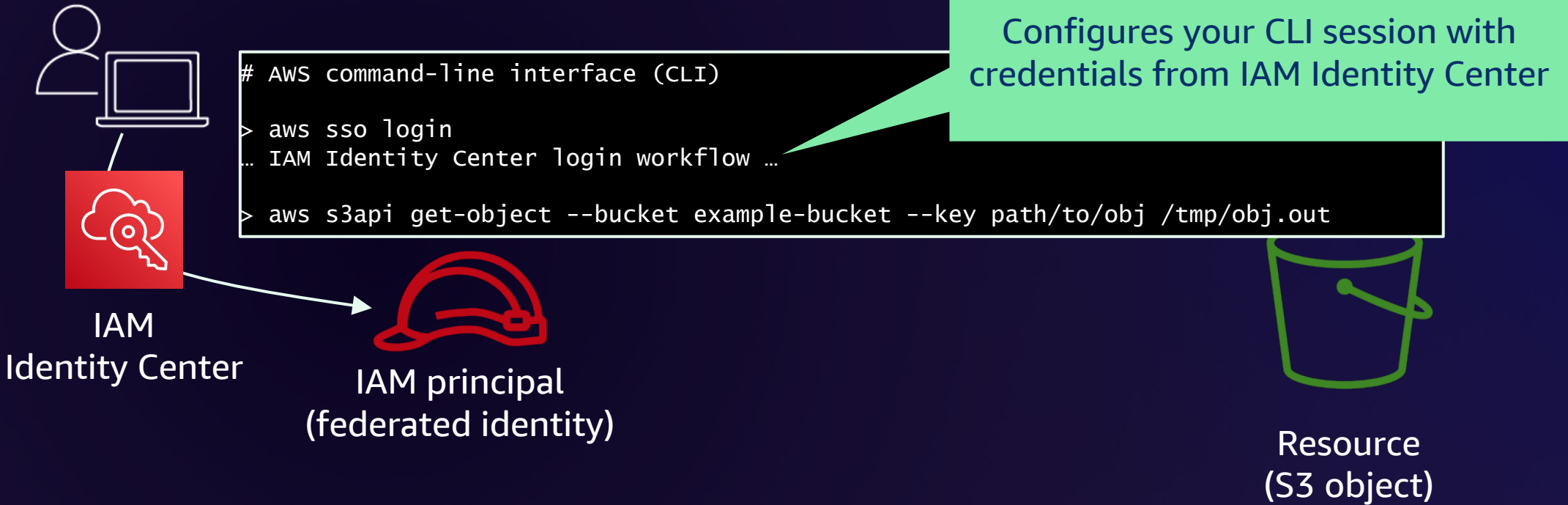


IAM principal
(federated identity)



Resource
(S3 object)

Authentication example: A request to Amazon S3



Authentication example: A request to Amazon S3

```
> aws s3api get-object --bucket example-bucket --key path/to/obj --debug /tmp/obj.out  
... Lots of debug output...
```



IAM principal



Resource
(S3 object)

Authentication example: A request to Amazon S3

```
> aws s3api get-object --bucket example-bucket --key path/to/obj --debug /tmp/obj.out
...
2023-04-20 13:57:04,144 - MainThread - botocore.auth - DEBUG - CanonicalRequest:
GET
/path/to/obj
host:example-bucket.s3.amazonaws.com
...
```



IAM principal



Resource
(S3 object)

Authentication example: A request to Amazon S3

```
> aws s3api get-object --bucket example-bucket --key path/to/obj --debug /tmp/obj.out
...
2023-04-20 13:57:04,144 - MainThread - botocore.auth - DEBUG - CanonicalRequest:
GET
/path/to/obj
host:example-bucket.s3.amazonaws.com
...
```

```
> dig example-bucket.s3.amazonaws.com
```

```
;; ANSWER SECTION:
```

```
example-bucket.s3.amazonaws.com. 42821 IN CNAME s3-1-w.amazonaws.com.
```

```
s3-1-w.amazonaws.com. 60 IN CNAME s3-w.us-east-1.amazonaws.com.
```

```
s3-w.us-east-1.amazonaws.com. 2 IN A 52.216.208.41
```

```
... more records...
```

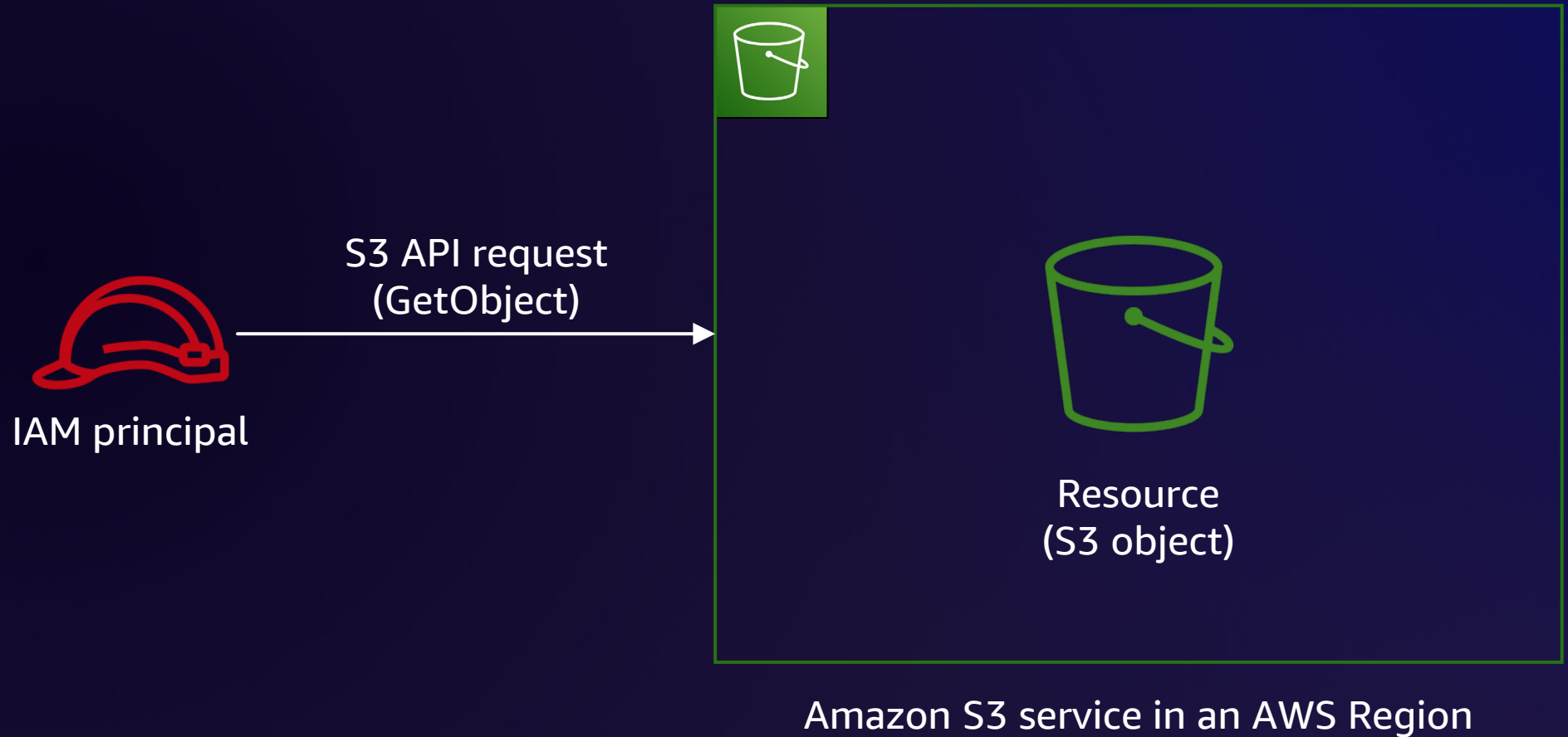


IAM principal

Reso
(S3 o

Amazon S3 in us-east-1

Authentication example: A request to Amazon S3



Authentication example: A request to Amazon S3

```
> aws s3api get-object --bucket example-bucket --key path/to/obj /tmp/obj.out
...
2023-04-20 13:57:04,144 - MainThread - botocore.auth - DEBUG - CanonicalRequest:
GET
/path/to/obj

host:example-bucket.s3.amazonaws.com
...
2023-04-20 13:57:04,145 - MainThread - botocore.endpoint - DEBUG - Sending http request:
<AWSPreparedRequest stream_output=True, method=GET, url=https://example-
bucket/path/to/obj, headers={ ... , 'Authorization': 'AWS4-HMAC-SHA256
Credential=ASIAEXAMPLE/20230420/us-east-1/s3/aws4_request, SignedHeaders=host;x-amz-
content-sha256;x-amz-date;x-amz-security-token, Signature=a76b58...8', 'X-Amz-Security-
Token': 'a-base64-value', ... }>
```

IAM principal

Resource
(S3 object)

Amazon S3 service in an AWS Region

Authentication example: A request to Amazon S3

Access key ID:
Identifies the requester

IAM principal

```
> aws s3api get-object --bucket example-bucket --key path/to/obj /tmp/obj.out
...
2023-04-20 13:57:04,144 - MainThread - botocore.auth - DEBUG - CanonicalRequest:
GET
/path/to/obj
host:example-bucket.s3.amazonaws.com
...
2023-04-20 13:57:04,145 - MainThread - botocore.endpoint - DEBUG - Sending http request:
<AWSPreparedRequest stream_output=True, method=GET, url=https://example-
bucket/path/to/obj, headers={ ... , 'Authorization': 'AWS4-HMAC-SHA256
Credential=ASIAEXAMPLE/20230420/us-east-1/s3/aws4_request, SignedHeaders=host;x-amz-
content-sha256;x-amz-date;x-amz-security-token, Signature=a76b58...8', 'X-Amz-Security-
Token': 'a-base64-value', ... }>
```

Resource
(S3 object)

Amazon S3 service in an AWS Region

Authentication example: A request to Amazon S3

Access key ID:
Identifies the requester



IAM principal

```
> aws s3api get-object --bucket example-bucket --key path/to/obj /tmp/obj.out
...
2023-04-20 13:57:04,144 - MainThread - botocore.auth - DEBUG - CanonicalRequest:
GET
/path/to/obj
host:example-bucket.s3.amazonaws.com
...
2023-04-20 13:57:04,145 - MainThread - botocore.endpoint - DEBUG - Sending http request:
<AWSPreparedRequest stream_output=True, method=GET, url=https://example-
bucket/path/to/obj, headers={ ... , 'Authorization': 'AWS4-HMAC-SHA256
Credential=ASIAEXAMPLE/20230420/us-east-1/s3/aws4_request, SignedHeaders=host;x-amz-
content-sha256;x-amz-date;x-amz-security-token, Signature=a76b58...8', 'X-Amz-Security-
Token': 'a-base64-value', ... }>
```

Signature of request:
Produced client-side (CLI)
with session credentials

Resource
(S3 object)

Amazon S3 service in an AWS Region

Authentication example: A request to Amazon S3

Access key ID:
Identifies the requester

```
> aws s3api get-object --bucket example-bucket --key path/to/obj /tmp/obj.out
...
2023-04-20 13:57:04,144 - MainThread - botocore.auth - DEBUG - CanonicalRequest:
GET
/path/to/obj
host:example-bucket.s3.amazonaws.com
...
2023-04-20 13:57:04,145 - MainThread - botocore.endpoint - DEBUG - Sending http request:
<AWSPreparedRequest stream_output=True, method=GET, url=https://example-
bucket/path/to/obj, headers={ ... , 'Authorization': 'AWS4-HMAC-SHA256
Credential=ASIAEXAMPLE/20230420/us-east-1/s3/aws4_request, SignedHeaders=host;x-amz-
content-sha256;x-amz-date;x-amz-security-token, Signature=a76b58...8', 'X-Amz-Security-
Token': 'a-base64-value', ... }>
```

Signature of request:
Produced client-side (CLI)
with session credentials



IAM principal

Resource
(S3 object)

Token associated with the
current role session

Amazon S3 service in an AWS Region



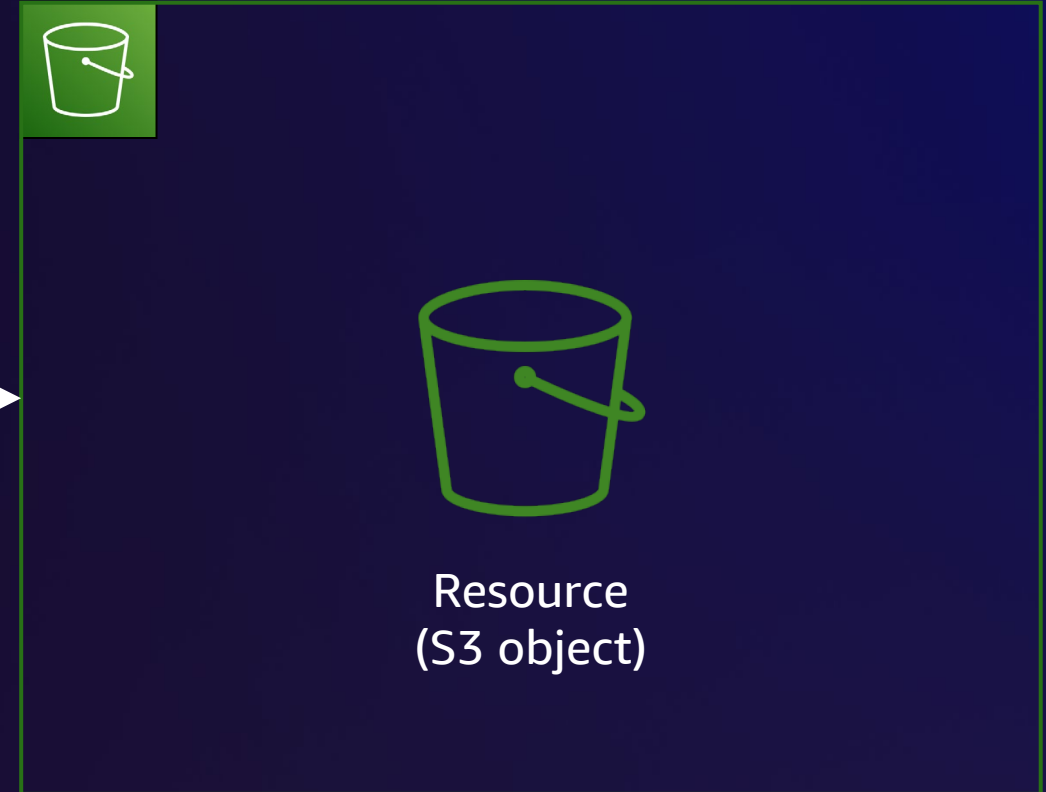
Authentication example: A request to Amazon S3

```
> aws s3api get-object --bucket example-bucket \  
  --key path/to/obj \  
  /tmp/obj.out
```



IAM principal

Amazon S3 API
request (GetObject)



Amazon S3 service in an AWS Region

Authentication example: A request to Amazon S3



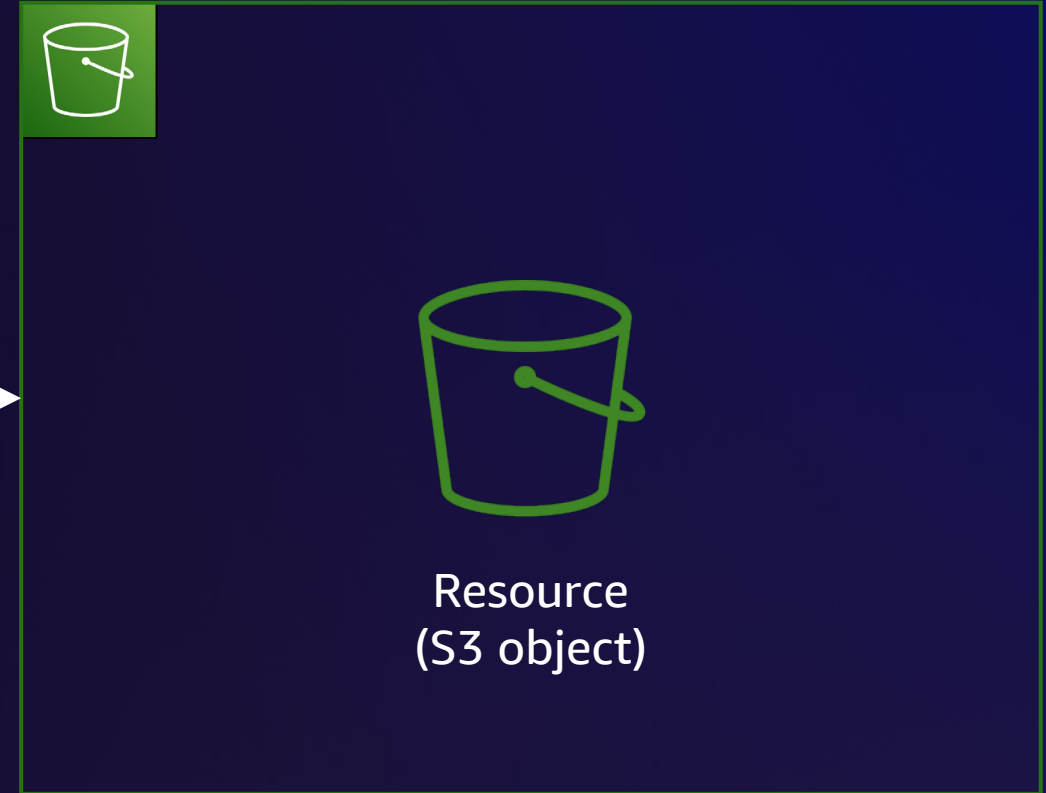
IAM data plane
in the region

```
> aws s3api get-object --bucket example-bucket \  
  --key path/to/obj \  
  /tmp/obj.out
```



IAM principal

SigV4-signed
request



Amazon S3 service in an AWS Region

Authentication example: A request to Amazon S3



IAM data plane
in the region

```
> aws s3api get-object --bucket example-bucket \  
  --key path/to/obj \  
  /tmp/obj.out
```



IAM principal



Amazon S3 service in an AWS Region

Successful authentication → Identity properties



Principal context

- `aws:PrincipalAccount`
- `aws:PrincipalOrgId`
- `aws:PrincipalIsAwsService`
- `aws:PrincipalTag/$TAG_NAME`
- Lots of others

Amazon S3 service in an AWS Region

Successful authentication → Identity policies



Principal context

- `aws:PrincipalAccount`
- `aws:PrincipalOrgId`
- `aws:PrincipalIsAwsService`
- `aws:PrincipalTag/$TAG_NAME`
- Lots of others



Amazon S3 service in an AWS Region

Authenticating a request: Summary

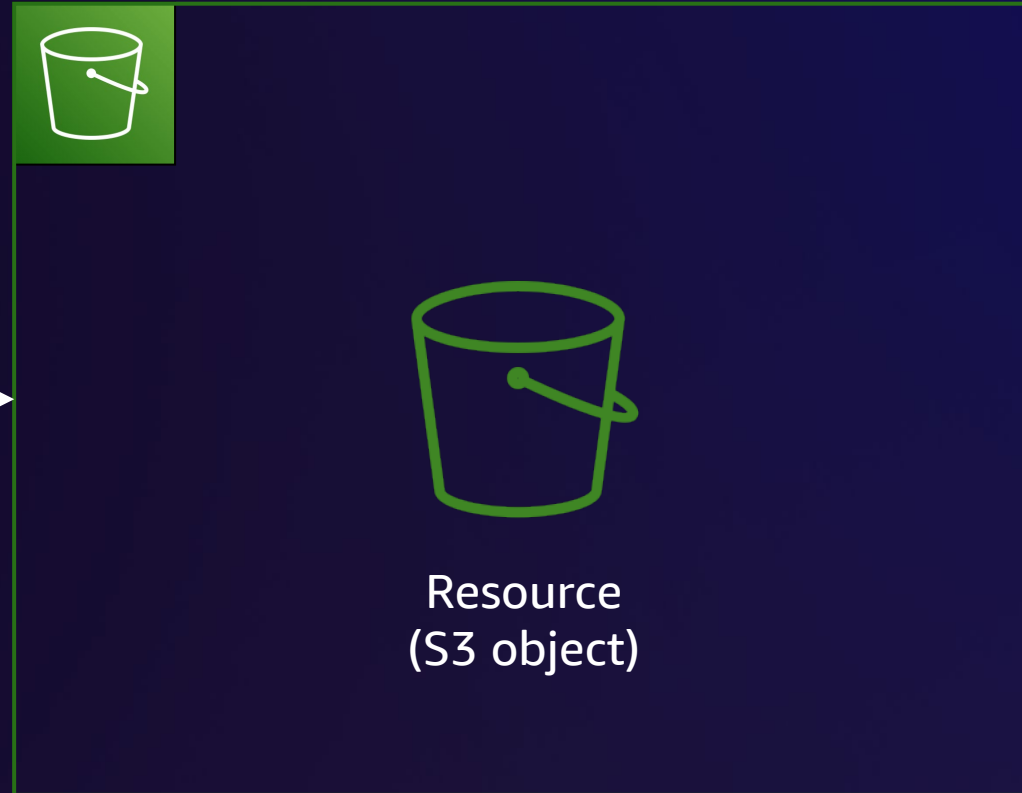


IAM data plane
in the region



IAM principal

SigV4-signed
request



Amazon S3 service in an AWS Region

Fundamentals of IAM: Authorization

IAM's little secret: Authorization is string-matching

Request properties

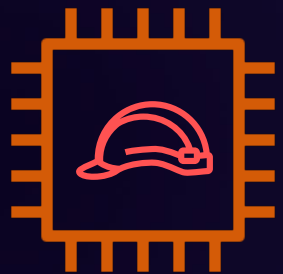
- `s3:GetObject`
- `aws:PrincipalXXX`
- `arn:aws:s3:::example-bucket/path/to/obj`
- `s3:[stuff]`
- `aws:SourceIp/Vpc`



Policy document[s]

Workloads tend to need specific IAM permissions

```
{  
  "Effect": "Allow",  
  "Action": "s3:GetObject",  
  "Resource": "arn:aws:s3:::example-bucket/*"  
}
```



Application running on EC2 instance

Read objects



S3 bucket

Write table items

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



DynamoDB table

Writing custom IAM policies

Create policy

A policy defines the AWS permissions that you can assign to a user, group, or role. You can create and edit a policy in the

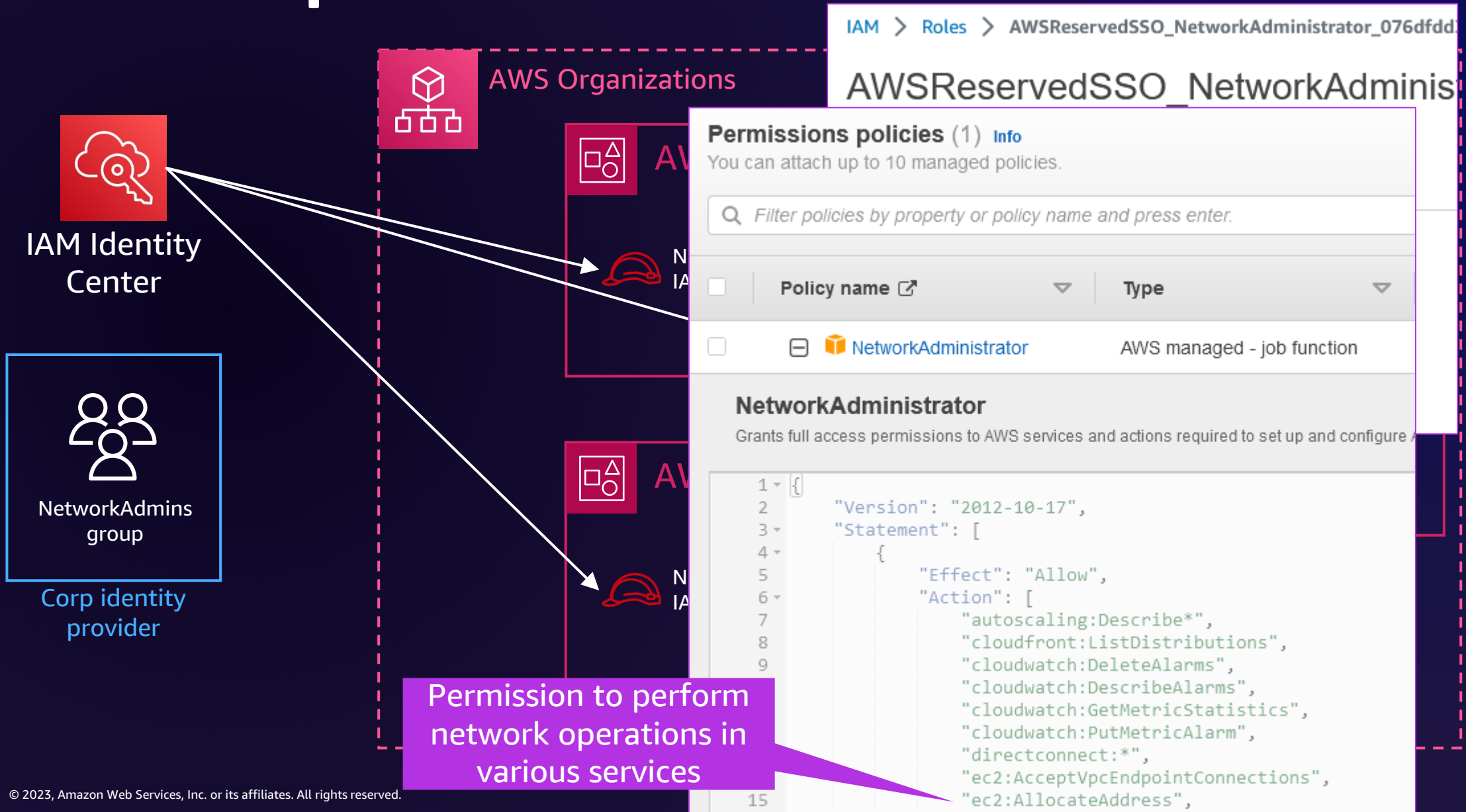
Visual editor JSON

```
1 {
2   "Version": "2012-10-17",
3   "Statement": [
4     {
5       "Effect": "Allow",
6       "Action": "s3:GetObject",
7       "Resource": "arn:aws:s3:::example-bucket/path/*"
8     },
9     {
10      "Effect": "Allow",
11      "Action": "kms:Decrypt",
12      "Resource": "arn:aws:kms:us-east-1:111122223333:key/01234567-89ab-cdef"
13    }
14  ]
15 }
```


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

Feedback on your IAM policy

Policies for operator access: Permission sets



IAM's little secret: Authorization is string-matching



Amazon S3

Request properties

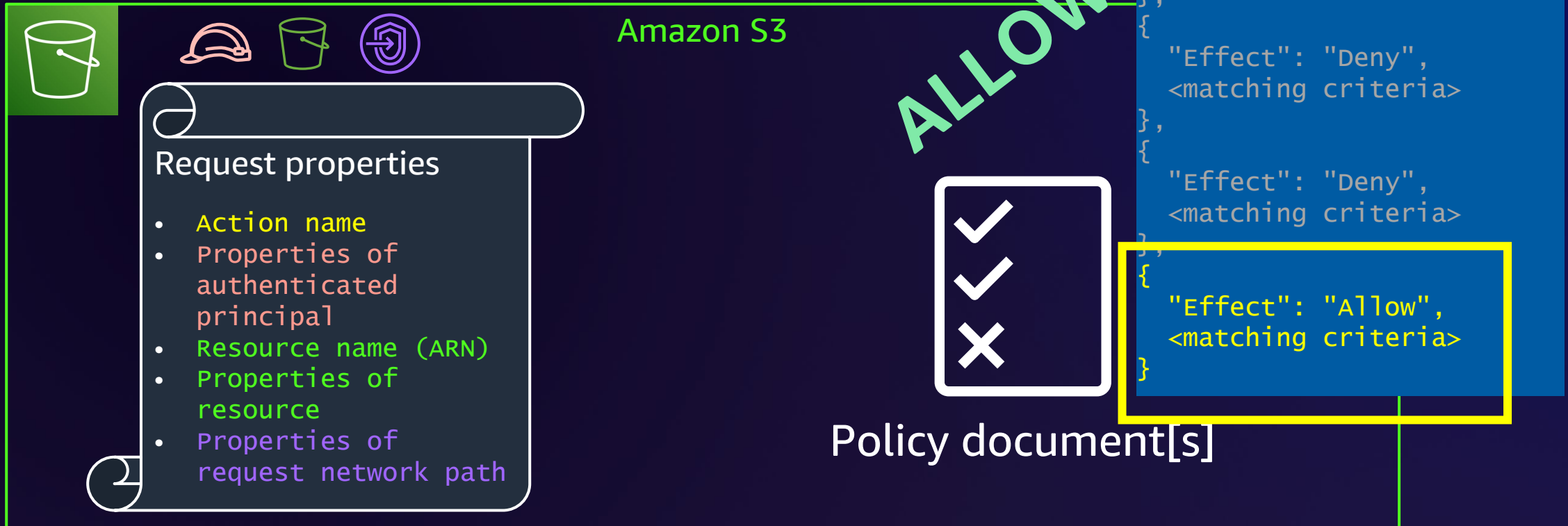
- `s3:GetObject`
- `aws:PrincipalXXX`
- `arn:aws:s3:::example-bucket/path/to/obj`
- `s3:[stuff]`
- `aws:SourceIp/Vpc`



Policy document[s]

```
{
  "Effect": "Allow",
  <matching criteria>
},
{
  "Effect": "Deny",
  <matching criteria>
},
{
  "Effect": "Deny",
  <matching criteria>
},
{
  "Effect": "Allow",
  <matching criteria>
}
```

IAM's little secret: Authorization is string-matching



IAM's little secret: Authorization is string-matching



IAM's little secret: Authorization is string-matching



Amazon S3

Request properties

- Action name
- Properties of authenticated principal
- Resource name (ARN)
- Properties of resource
- Properties of request network path

DENY



Policy document[s]

```
{
  "Effect": "Allow",
  <matching criteria>
},
{
  "Effect": "Deny",
  <matching criteria>
},
{
  "Effect": "Deny",
  <matching criteria>
},
{
  "Effect": "Allow",
  <matching criteria>
}
```

How authorization works on AWS



Principal context

- `aws:PrincipalAccount`
- `aws:PrincipalOrgId`
- `aws:PrincipalIsAwsService`
- `aws:PrincipalTag/$TAG_NAME`
- Lots of others

Result of successful authentication:
Properties and policies of the caller

How authorization works on AWS



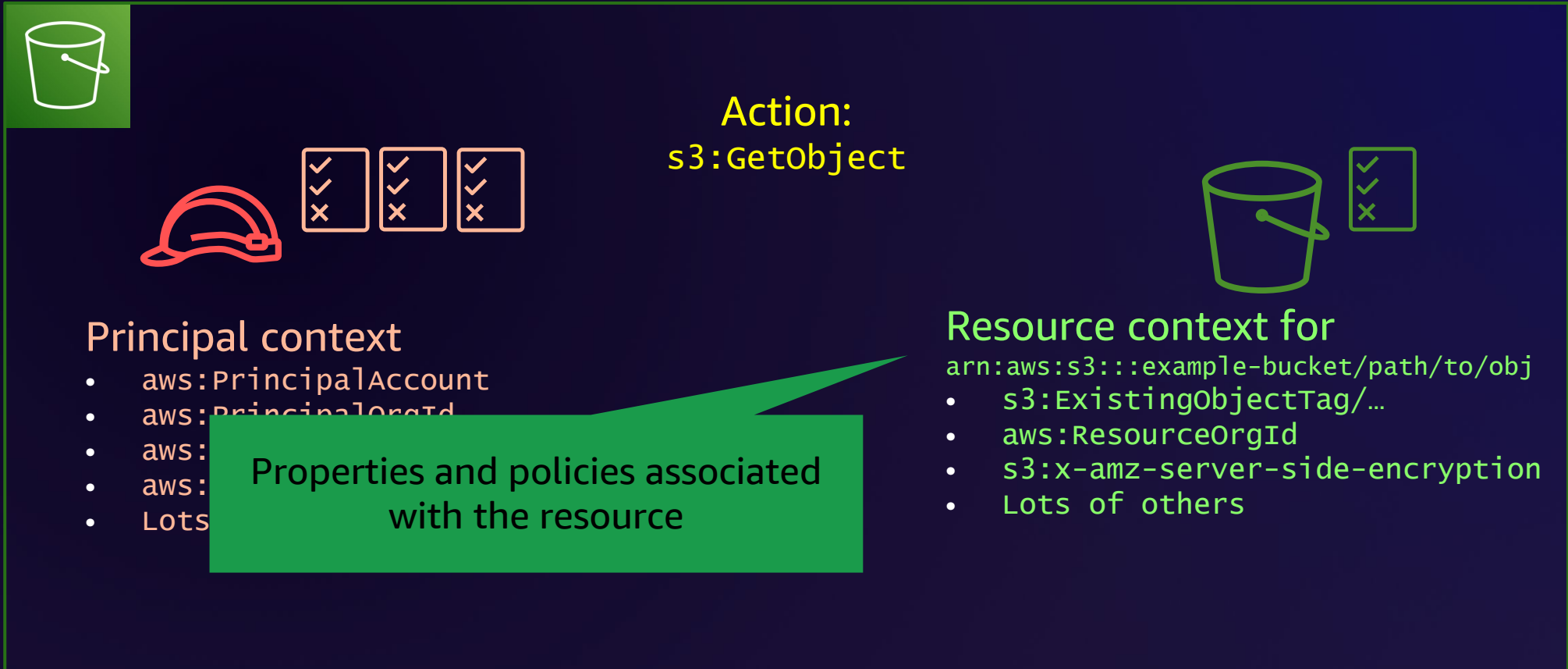
Principal context

- `aws:PrincipalAccount`
- `aws:PrincipalOrgId`
- `aws:PrincipalIsAwsService`
- `aws:PrincipalTag/$TAG_NAME`
- Lots of others

Action:
`s3:GetObject`

Assigned by service based on API
method invoked

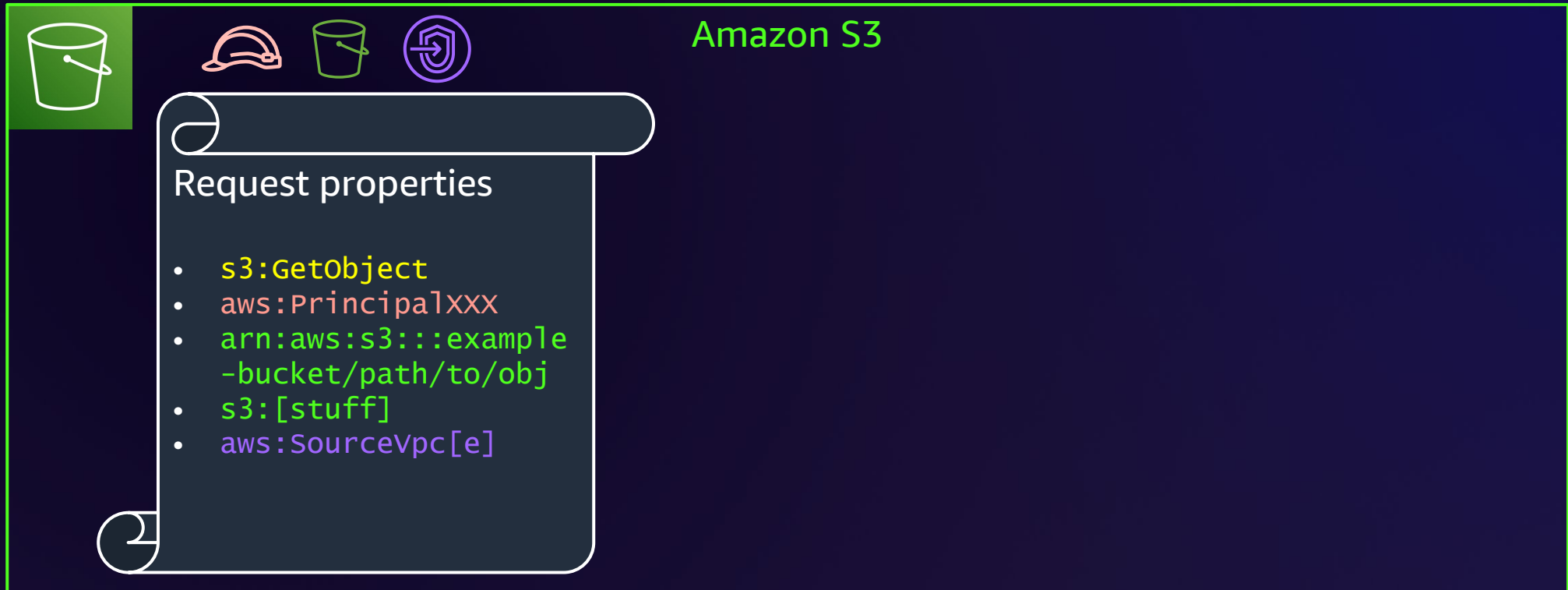
How authorization works on AWS



How authorization works on AWS



The authorization request context



The diagram illustrates the authorization request context for Amazon S3. It features a central scroll titled "Request properties" containing a list of context elements. Above the scroll are icons for a bucket, a hard hat, a bucket with a checkmark, and a shield with a checkmark. The text "Amazon S3" is positioned to the right of these icons.





Amazon S3

Request properties

- `s3:GetObject`
- `aws:PrincipalXXX`
- `arn:aws:s3:::example-bucket/path/to/obj`
- `s3:[stuff]`
- `aws:SourceVpc[e]`

Authorization: Matching a request to policy statements





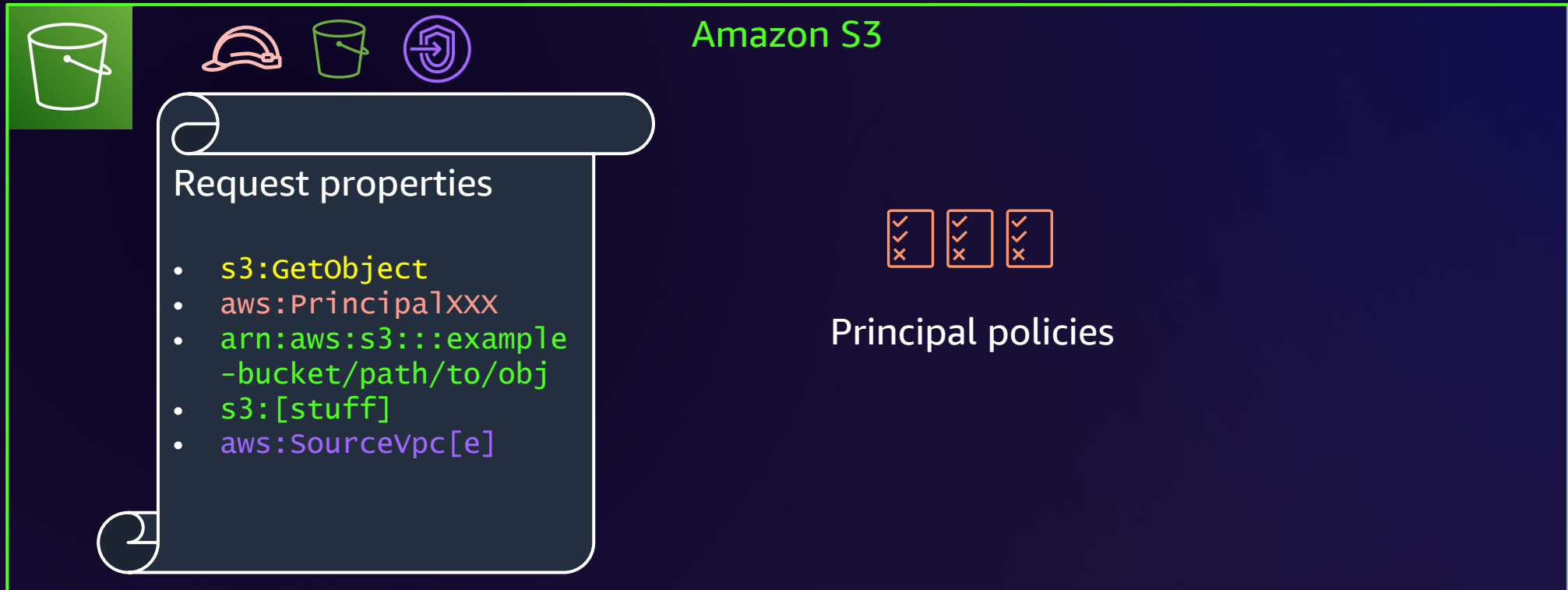
Amazon S3

Request properties

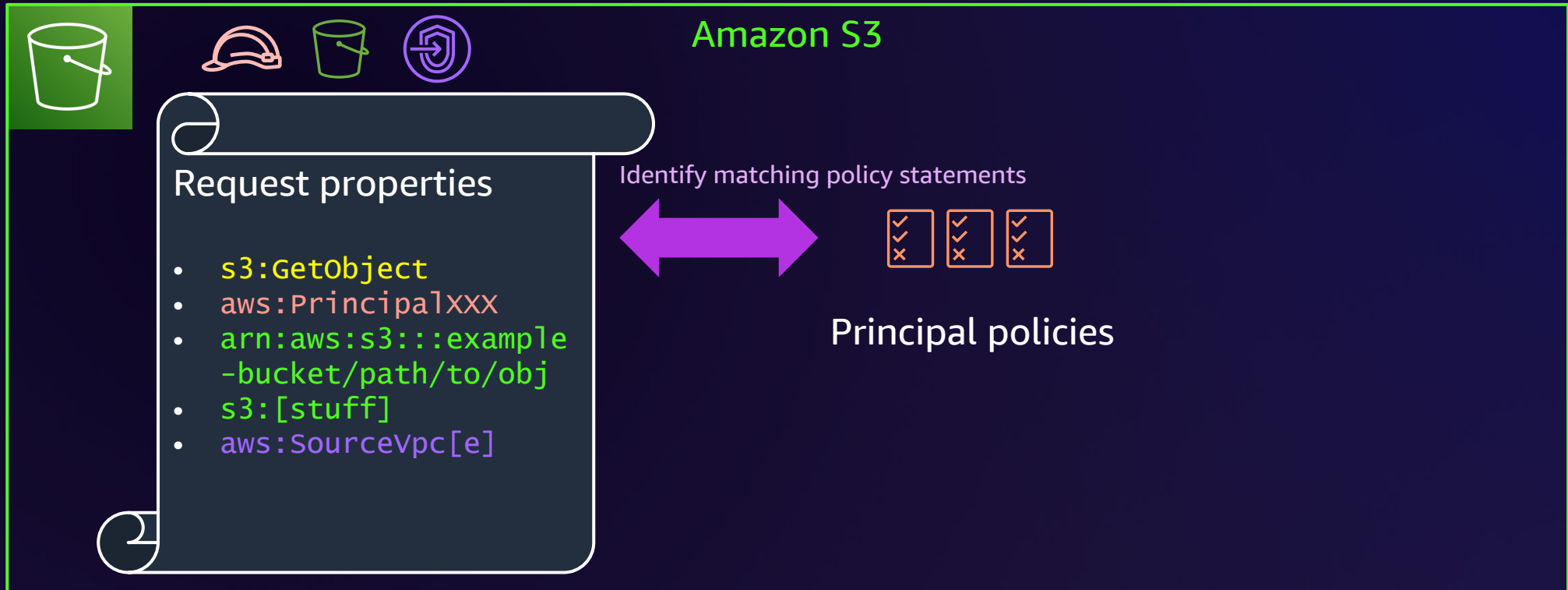
- `s3:GetObject`
- `aws:PrincipalXXX`
- `arn:aws:s3:::example-bucket/path/to/obj`
- `s3:[stuff]`
- `aws:SourceVpc[e]`

Principal policies

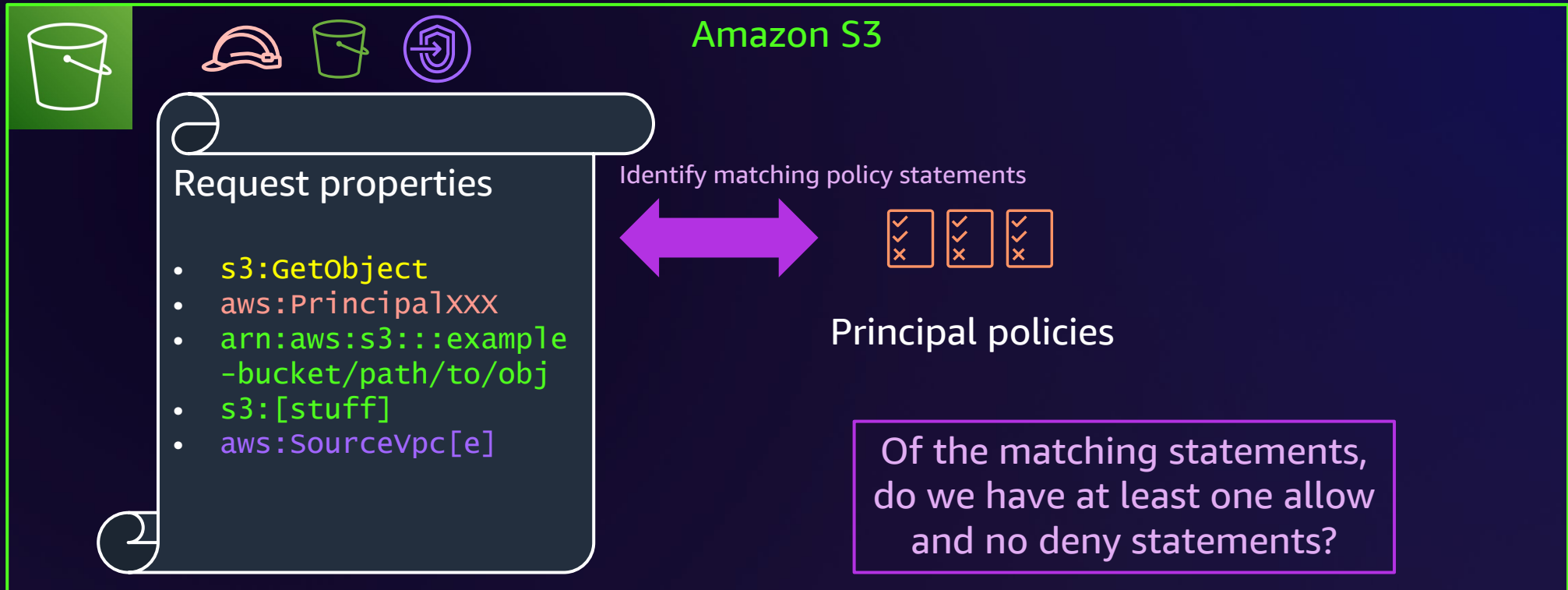
Authorization: Matching a request to policy statements



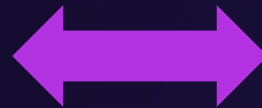
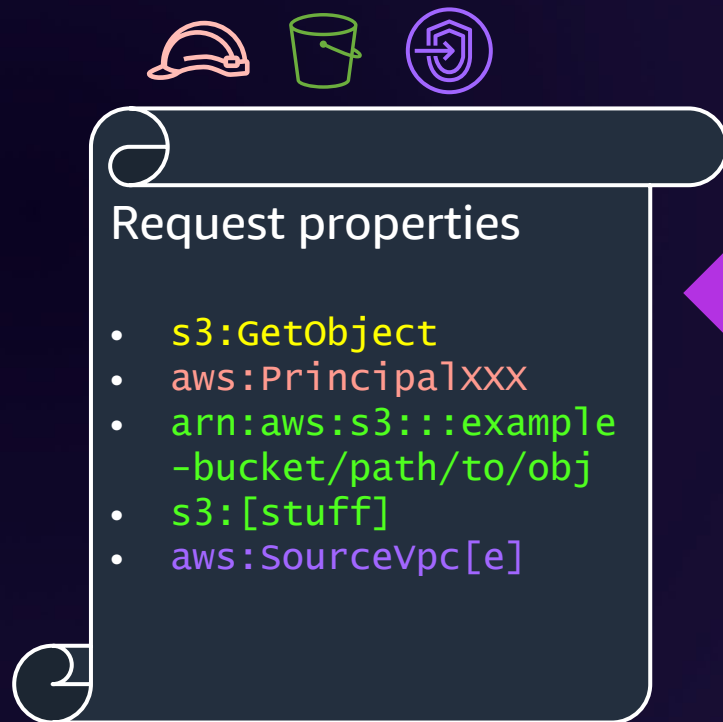
Authorization: Matching a request to policy statements



Authorization: Matching a request to policy statements



Authorizing a request: Summary



Principal policies

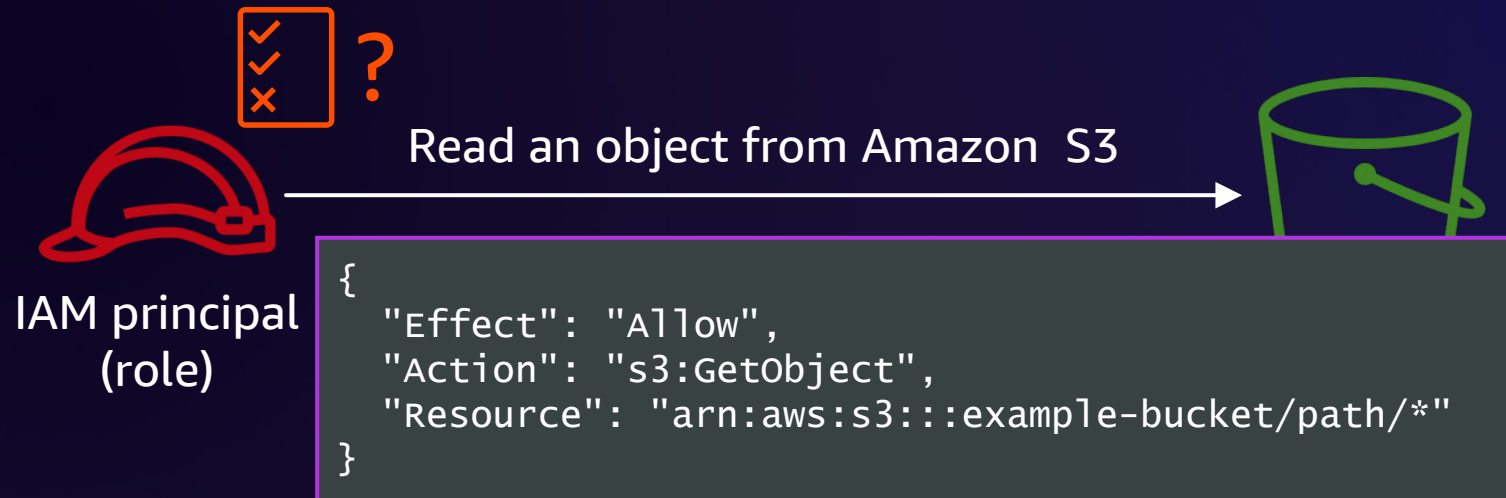
ALLOW?

!DENY?

IAM policy-writing for builders

Example: Read-only access to part of an S3 bucket

```
# AWS command-line interface (CLI)  
> aws s3api get-object --bucket example-bucket --key path/to/obj /tmp/obj.out
```



Bookmark this page: IAM policy documentation

The screenshot shows the AWS documentation page for 'Actions, resources, and condition keys for AWS services'. The page title is 'Actions, resources, and condition keys for AWS services' with a 'PDF' link below it. The breadcrumb trail is 'AWS > Documentation > Service Authorization Reference > Service Authorization Reference'. A left-hand navigation pane is open, showing a search bar and a list of services under 'Reference'. The service 'Actions, resources, and condition keys' is selected and highlighted in blue. A dark blue callout box with white text points to this selected item, stating: 'Full documentation of every IAM action for every AWS service'. The main content area shows the beginning of the article, including the heading 'The actions table' and the start of a paragraph: 'The **Actions** table lists all the actions that you can use in an IAM policy statement's **Action** element as an action in an IAM policy. Some services include permission-only actions that don't directly...'. A QR code is located in the bottom right corner of the screenshot area, enclosed in a blue square.

https://docs.aws.amazon.com/service-authorization/latest/reference/reference_policies_actions-resources-contextkeys.html



IAM policy documentation for Amazon S3

Amazon Route 53 Recovery Readiness					s3:TlsVersion
Amazon Route 53 Resolver					s3:DataAccessPointAccount
Amazon S3	GetObject	Grants permission to retrieve objects from Amazon S3	Read	object*	s3:DataAccessPointArn
Amazon S3 Glacier					s3:AccessPointNetworkOrigin
Amazon S3 Object Lambda					s3:ExistingObjectTag/<key>
Amazon S3 on Outposts					s3:authType
Amazon SageMaker					s3:ResourceAccount
Amazon SageMaker geospatial capabilities					s3:signatureAge
Amazon SageMaker Ground Truth Synthetic					s3:signatureversion
AWS Savings Plans					s3:TlsVersion
AWS Secrets Manager					s3:x-amz-content-sha256
AWS Security Hub					

Action:
s3:GetObject

Resource:
object

Conditions specific to
the s3:GetObject
IAM action



IAM policy documentation for Amazon S3

Resource types defined by Amazon S3

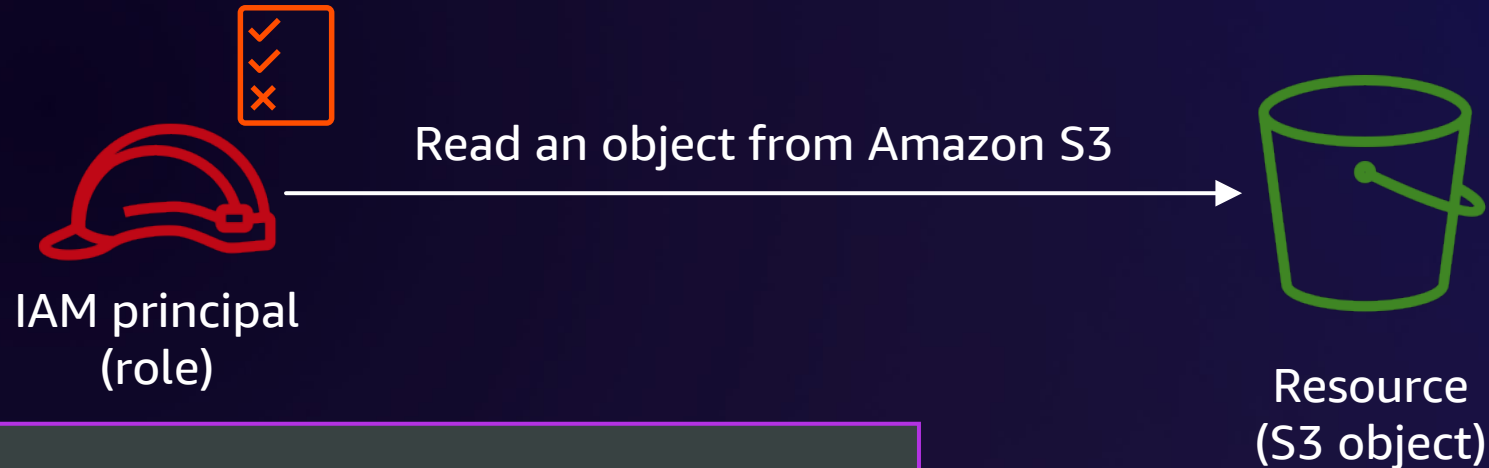
The following resource types are defined by this service and can be used in the `Resource` element of IAM permission policy statements. Each action that can be specified with that action. A resource type can also define which condition keys you can include in a policy. These keys are displayed in the columns in the following table, see [Resource types table](#).

Resource types	ARN
accesspoint	<code>arn:\${Partition}:s3:\${Region}:\${Account}:accesspoint/\${AccessPointName}</code>
bucket	<code>arn:\${Partition}:s3:::\${BucketName}</code>
object	<code>arn:\${Partition}:s3:::\${BucketName}/\${ObjectName}</code>
job	<code>arn:\${Partition}:s3:\${Region}:\${Account}:job/\${JobId}</code>
storagelensconfiguration	<code>arn:\${Partition}:s3:\${Region}:\${Account}:storagelensconfiguration/\${StorageLensConfigurationName}</code>
objectlambdaaccesspoint	<code>arn:\${Partition}:s3:\${Region}:\${Account}:objectlambdaaccesspoint/\${AccessPointName}</code>

Amazon Resource Name (ARN) for an S3 object:
`arn:aws:s3:::example-bucket/path/to/obj`

Example: Read-only access to part of an S3 bucket

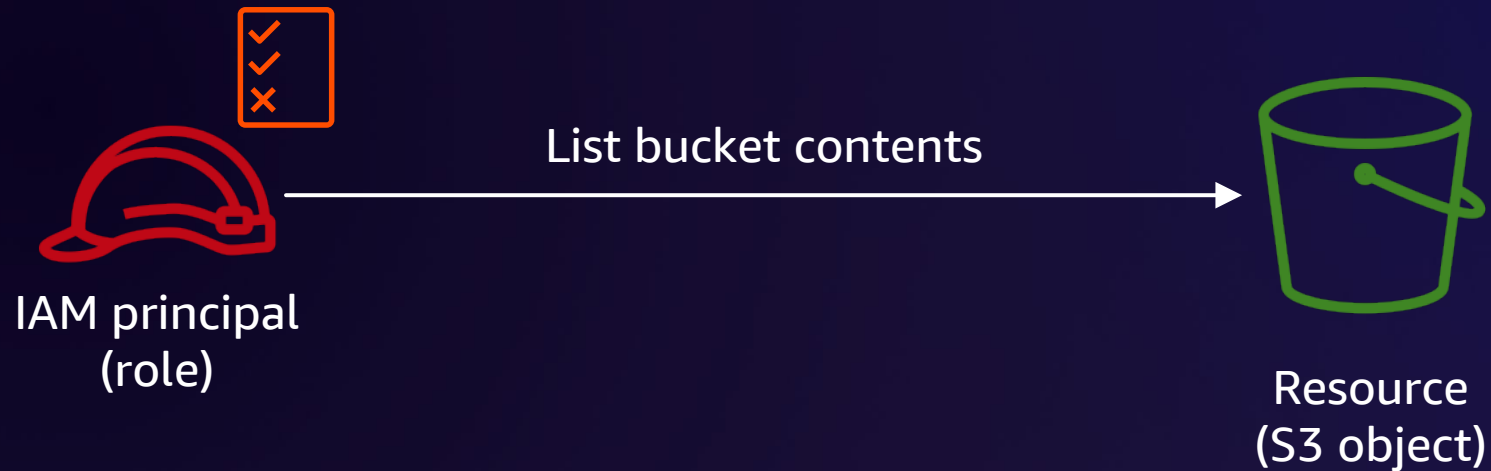
```
# AWS command-line interface (CLI)  
> aws s3api get-object --bucket example-bucket --key path/to/obj /tmp/obj.out
```



```
{  
  "Effect": "Allow",  
  "Action": "s3:GetObject",  
  "Resource": "arn:aws:s3:::example-bucket/path/*"  
}
```

Example: Read-only access to Amazon S3, including list

```
# AWS command-line interface (CLI)  
> aws s3api list-objects-v2 --bucket example-bucket
```



IAM policy documentation for Amazon S3

<p>ListBucket</p>	<p>Grants permission to list some or all of the objects in an Amazon S3 bucket (up to 1000)</p>	<p>List</p>	<p>bucket*</p>	<ul style="list-style-type: none">s3:DataAccessPointAccounts3:DataAccessPointArns3:AccessPointNetworkOrigins3:authTypes3:delimiters3:max-keyss3:prefixs3:ResourceAccounts3:signatureAges3:signatureversions3:TlsVersions3:x-amz-content-sha256
-------------------	-------------------------------------------------------------------------------------------------	-------------	----------------	-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------



Example: Read-only access to Amazon S3, including list

```
// Statements from IAM policy
...
{
  "Effect": "Allow",
  "Action": "s3:GetObject",
  "Resource": "arn:aws:s3:::example-bucket/path/*"
},
{
  "Effect": "Allow",
  "Action": "s3:ListBucket",
  "Resource": "arn:aws:s3:::example-bucket"
}
```

ListObjects response shows all objects in the bucket

```
# AWS command-line interface (CLI)
> aws s3api list-objects-v2 --bucket example-bucket \
  --query 'Contents[].{Key:Key}' --output text

other-path/object1.txt
other-path/object2.txt
path/hello.jpg
yet-another-path/mydata.cs
```

What happens here?

Example: Read-only access to Amazon S3, including list

```
// Statements from IAM policy
...
{
  "Effect": "Allow",
  "Action": "s3:GetObject",
  "Resource": "arn:aws:s3:::example-bucket/path/*"
},
{
  "Effect": "Allow",
  "Action": "s3:ListBucket",
  "Resource": "arn:aws:s3:::example-bucket",
  "Condition": {
    "StringLike": {
      "s3:prefix": "path/*"
    }
  }
}
```

Example: Read-only access to Amazon S3, including list

```
// Statements from IAM policy
...
{
  "Effect": "Allow",
  "Action": "s3:GetObject",
  "Resource": "arn:aws:s3:::example-bucket/path/*"
},
{
  "Effect": "Allow",
  "Action": "s3:ListBucket",
  "Resource": "arn:aws:s3:::example-bucket",
  "Condition": {
    "StringLike": {
      "s3:prefix": "path/*"
    }
  }
}
```

No prefix was specified, so the allow statement does not match

```
# AWS command-line interface (CLI)
> aws s3api list-objects-v2 --bucket example-bucket \
  --query 'Contents[].{Key:Key}' --output text

An error occurred (AccessDenied) when calling the
ListObjectsV2 operation: Access Denied
```

What happens here?

Example: Read-only access to Amazon S3, including list

```
// Statements from IAM policy
...
{
  "Effect": "Allow",
  "Action": "s3:GetObject",
  "Resource": "arn:aws:s3:::example-bucket/path/*"
},
{
  "Effect": "Allow",
  "Action": "s3:ListBucket",
  "Resource": "arn:aws:s3:::example-bucket",
  "Condition": {
    "StringLike": {
      "s3:prefix": "path/*"
    }
  }
}
```

The request is specifying a prefix that matches the condition in the allow statement

```
# AWS command-line interface (CLI)
> aws s3api list-objects-v2 --bucket example-bucket \
  --prefix path/
  --query 'Contents[].{Key:Key}' --output text
path/hello.jpg
```

What happens here?

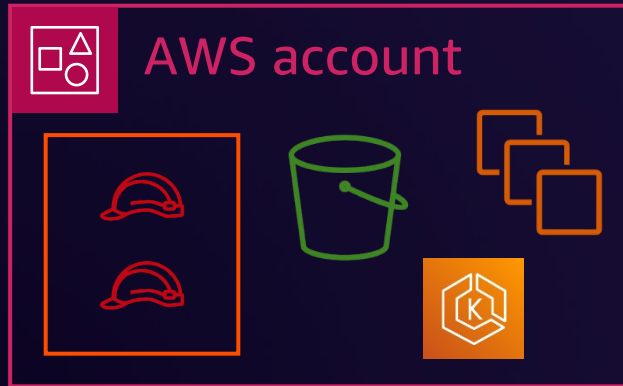
Example: Invoke a Lambda function



```
// Statements from IAM policy
...
{
  "Effect": "Allow",
  "Action": "lambda:InvokeFunction",
  "Resource": "arn:aws:lambda:us-east-2:111122223333:function:MyFunction"
};
```

ARN of a Lambda function

Authorizing requests made by other AWS accounts

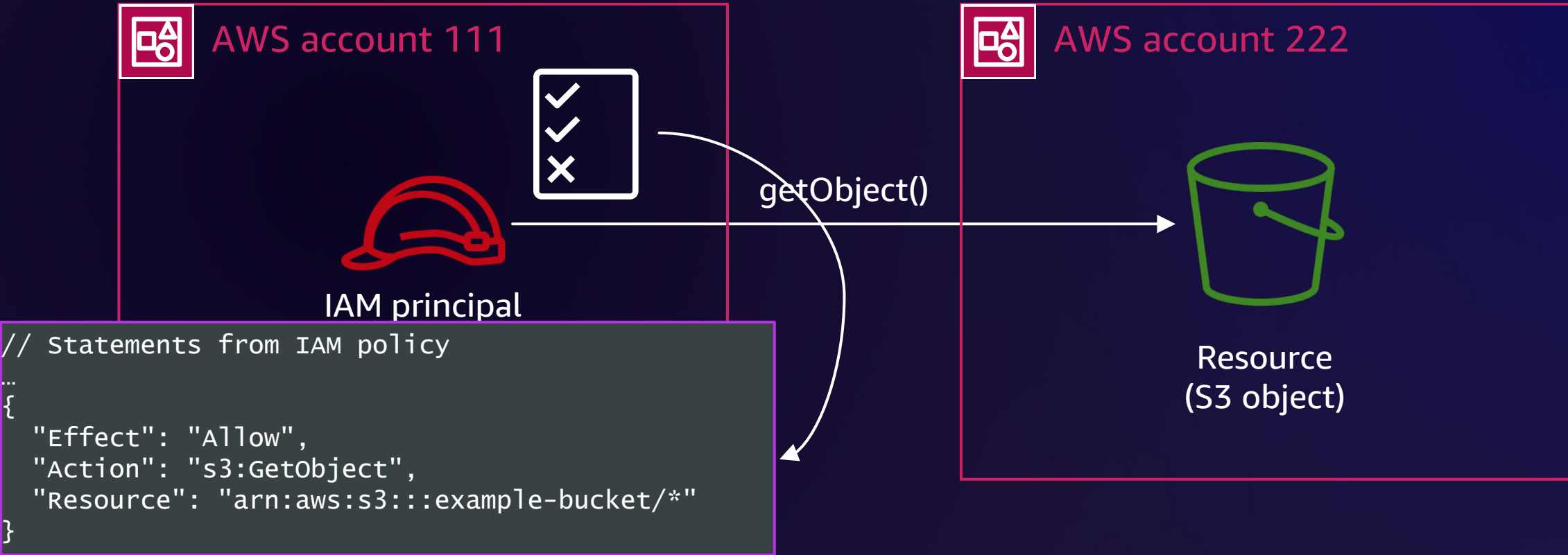


Cloud infrastructure that we own

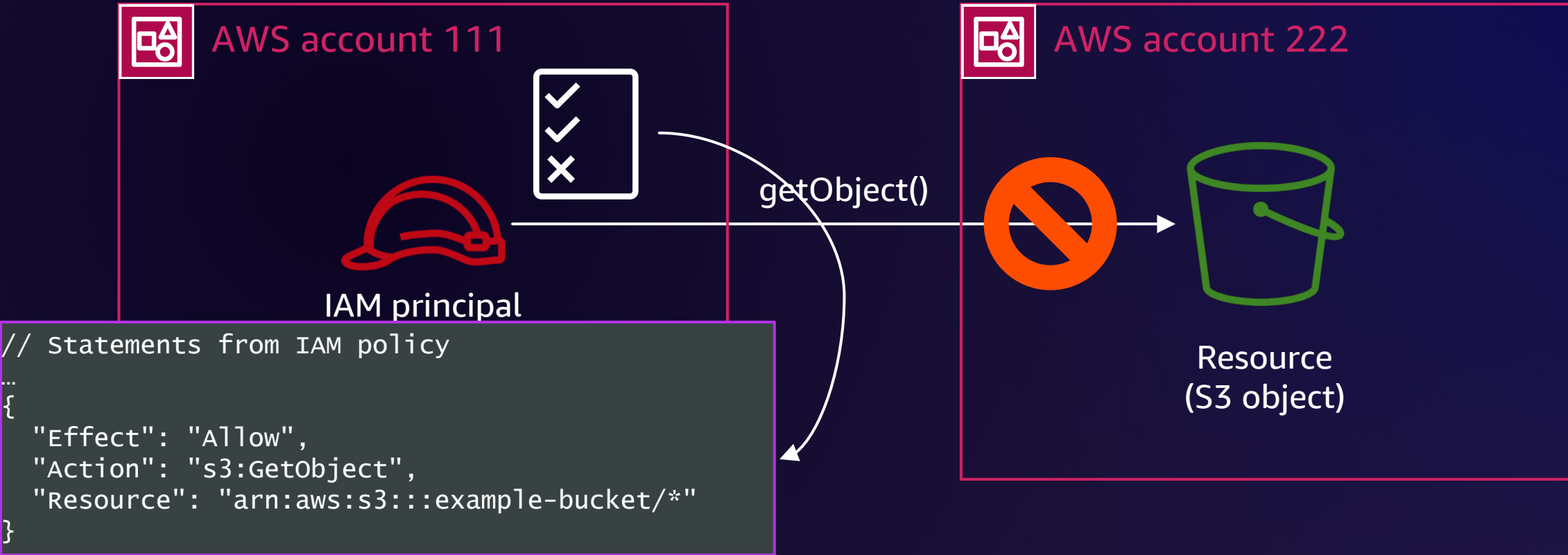
Reading an object from an Amazon S3 bucket in another account



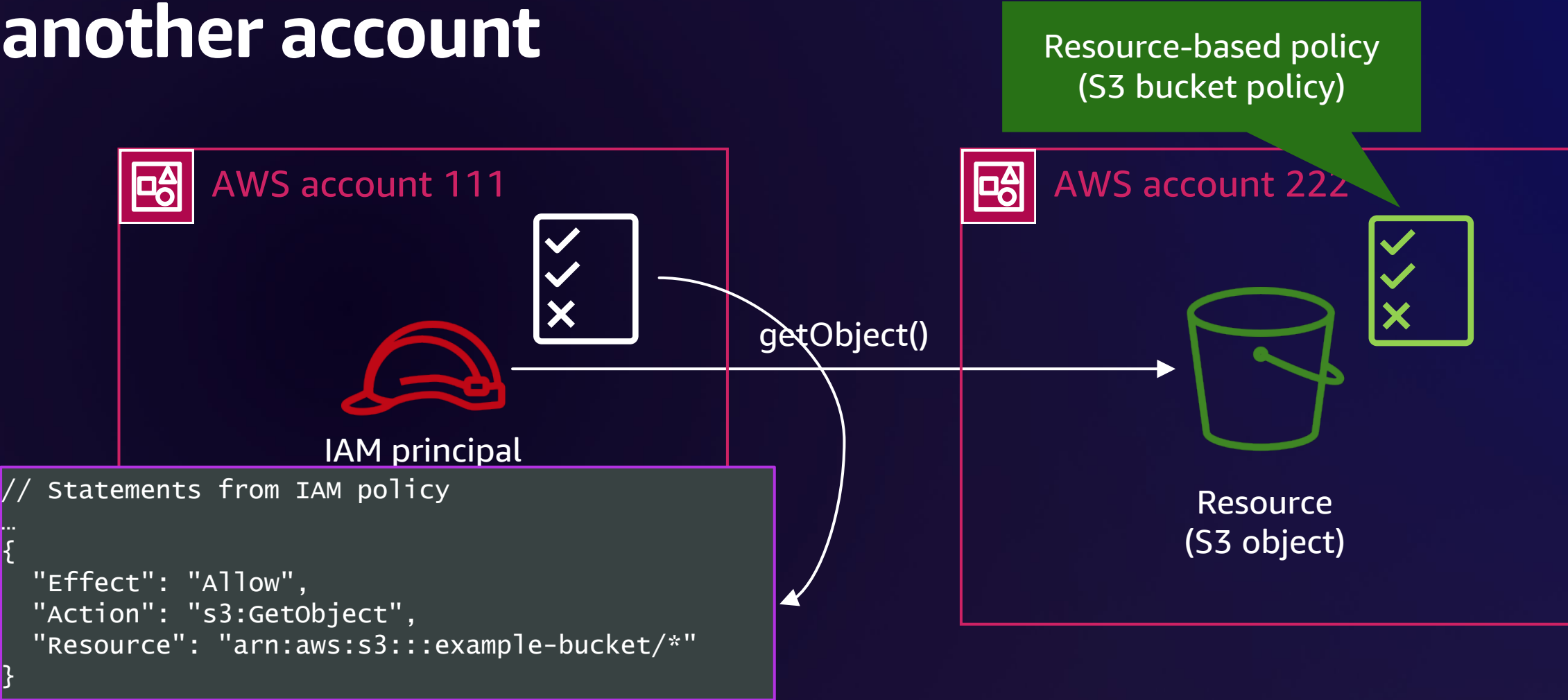
Reading an object from an Amazon S3 bucket in another account



Reading an object from an Amazon S3 bucket in another account

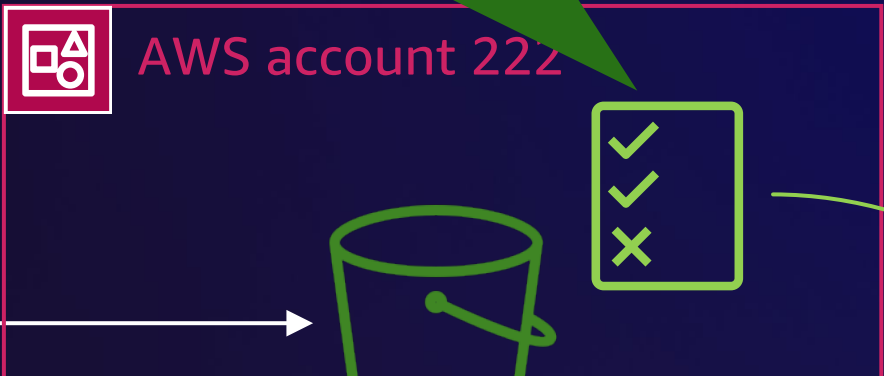


Reading an object from an Amazon S3 bucket in another account



Reading an object from an Amazon S3 bucket in another account

Resource-based policy (S3 bucket policy)



getObject()

```
// Statements from IAM policy
...
{
  "Effect": "Allow",
  "Action": "s3:GetObject",
  "Resource": "arn:aws:s3:::example-bucket/*"
}
```





```
{
  "Effect": "Allow",
  "Principal": {
    "AWS": "111"
  },
  "Action": "s3:GetObject",
  "Resource": "arn:aws:s3:::example-bucket/*"
}
```



Cross-account authorization





Amazon S3


Request properties

- `s3:GetObject`
- `aws:PrincipalXXX`
- `arn:aws:s3:::example-bucket/path/to/obj`
- `s3:[stuff]`
- `aws:SourceVpc[e]`

Principal policies: 

Resource policy: 

Issued by caller account Issued by bucket-owner account



Each authorization must explicitly allow the access

Resource-based policy example: Amazon S3 cross-account

```
// Necessary, but not sufficient
```

```
{  
  "Effect": "Allow",  
  "Action": "s3:GetObject",  
  "Resource": "arn:aws:s3:::example-bucket/path/*"  
}
```



Principal policy:

Issued by caller account
111

```
// Necessary, but not sufficient
```

```
{  
  "Effect": "Allow",  
  "Principal": {  
    "AWS": "111"  
  },  
  "Action": "s3:GetObject",  
  "Resource": "arn:aws:s3:::example-bucket/*"  
}
```







Resource policy:

Issued by bucket-owner account
222

Same-account authorization with resource policy





Amazon S3

Request properties

- `s3:GetObject`
- `aws:PrincipalXXX`
- `arn:aws:s3:::example-bucket/path/to/obj`
- `s3:[stuff]`
- `aws:SourceVpc[e]`

Principal policies Resource policy

Issued by caller account:
Single authorization

Resource-based policy example: Assume-role policies on IAM accounts themselves



AWS Lambda
service



Lambda function
runtime environment

```
// Role Trust Policy, a.k.a. Assume-Role Policy.  
//  
// This policy is attached to the IAM Role itself and  
// says who can obtain its credentials (the AWS Lambda  
// service in this case)
```

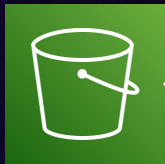
```
{  
  "Version": "2012-10-17",  
  "Statement": [  
    {  
      "Effect": "Allow",  
      "Principal": {  
        "Service": "lambda.amazonaws.com"  
      },  
      "Action": "sts:AssumeRole"  
    }  
  ]  
}
```

Service principal:

Identifies an AWS service acting
on your resources

What AWS Lambda can do:
Obtain credentials for this role

Resource-based policy example: Amazon S3 permission to send events to an Amazon SQS queue



Amazon S3, sending event notifications to an Amazon SQS queue



Amazon SQS queue

```
{
  "Effect": "Allow",
  "Principal": {
    "Service": "s3.amazonaws.com"
  },
  "Action": "sqs:SendMessage",
  "Resource": "arn:aws:sqs:us-east-2:111122223333:my-queue",
  "Condition": {
    "ArnEquals": {
      "aws:SourceAccount": "111122223333",
      "aws:SourceArn": "arn:aws:s3:::example-bucket"
    }
  }
}
```



The Amazon S3 service can send messages to this Amazon SQS queue, as long as those messages correspond to this bucket and account

What we didn't cover

Concepts to learn about next

- AWS Organizations and service control policies
- VPC endpoints and VPC endpoint policies
- Building a data perimeter on AWS

Learning more about IAM

- Data perimeters

<https://docs.aws.amazon.com/whitepapers/latest/building-a-data-perimeter-on-aws/building-a-data-perimeter-on-aws.html>

- Build some things on AWS ← This technique works best

AWS IAM recognized by Gartner Peer Insights



Thank you!

Becky Weiss

becky@amazon.com



Please complete
the session survey
in the mobile app