SUMMIT

Atlanta

SEC301

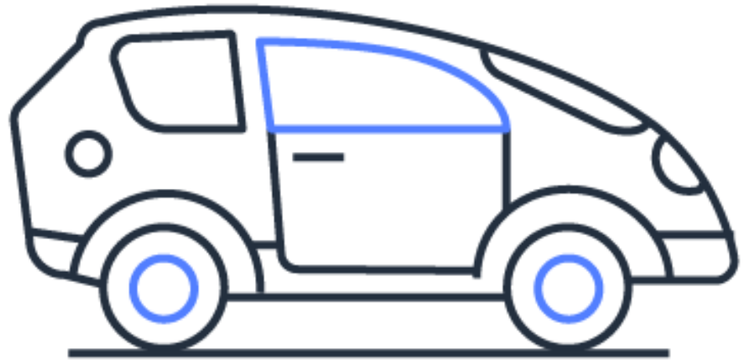
Permissions Boundary Workshop

aws SUMMIT

Letting your teenager drive

- Car keys give a lot of power: drive fast, drive anywhere, etc.
- You can give your teenager rules: don’t speed, don’t go beyond 20 mile range, etc.
- You can only verify that they followed your rules (check odometer, see if they got a speeding ticket or got into an accident.)

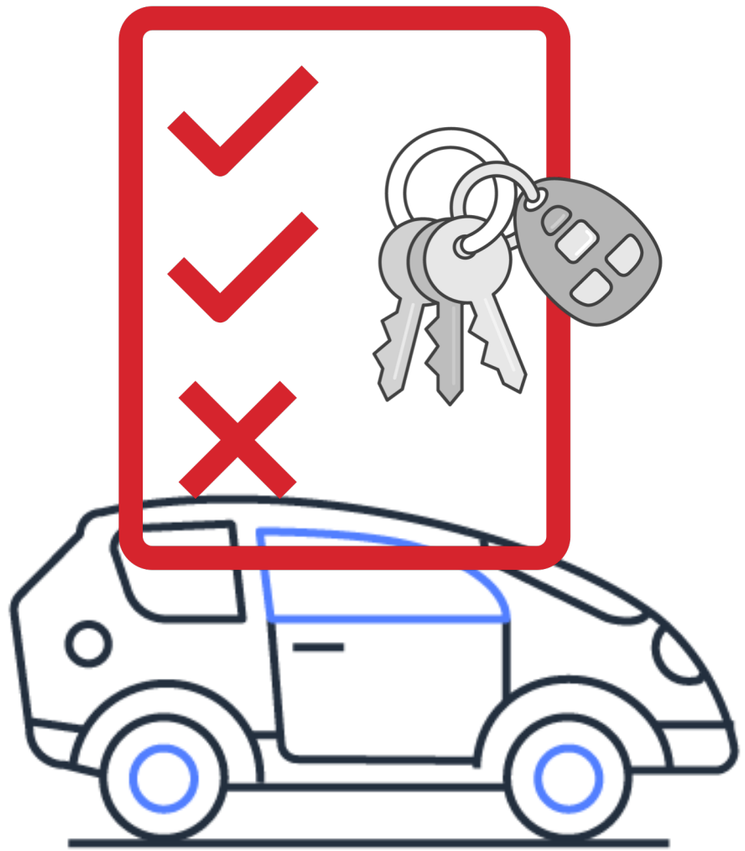
Once they have the car keys they can drive however they want.



Letting your teenager drive

- Some car brands have programmable keys so you can restrict certain parameters.
- Permissions in the car (drive fast, blast the radio or even spin the tires) is the intersection between their desire and the settings you program.

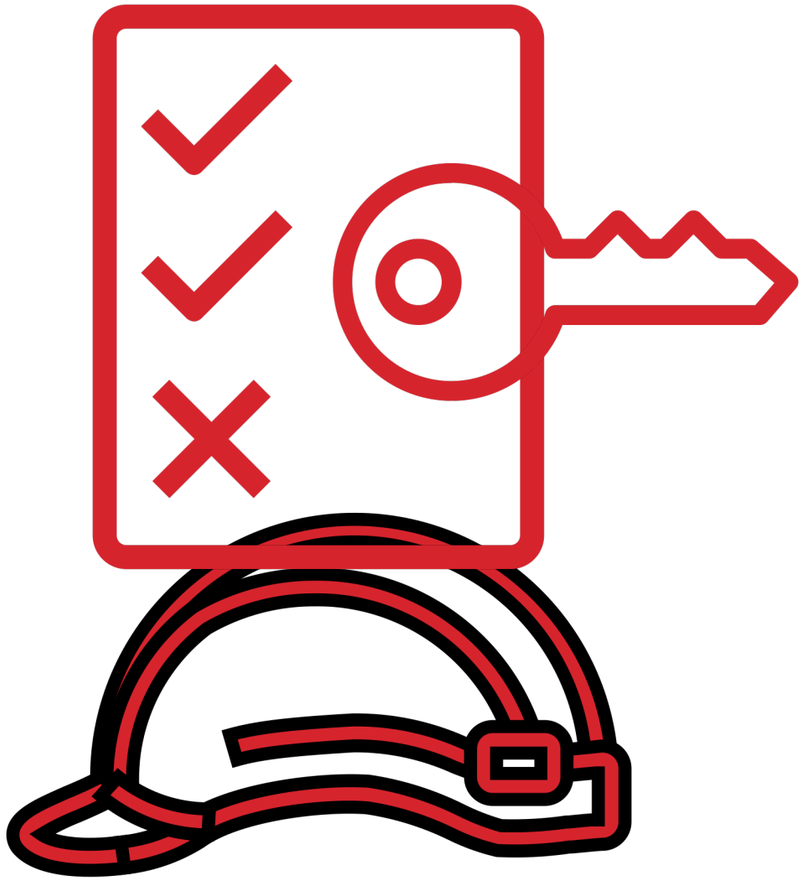
Programming sets maximum of their driving ability.



Letting your developer create roles

- Permission to create users or roles is like giving car keys to developers.
- Developer attaches policies (what they want a role to be able to do) but you can also require a permissions boundary (like the programming on the car key).
- Effective permission of the role is the intersection of the two.

Permissions boundary sets maximum permissions of the role.



Agenda

Basics
Demo
Mechanism
Resource Restrictions
Hands On

Basics

aws SUMMIT

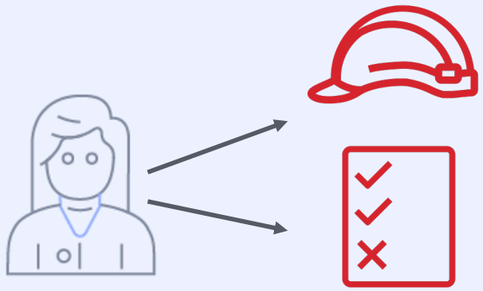
What are permissions boundaries?

Controls the maximum permissions that a user and/or role, created by a delegated admin, can have.

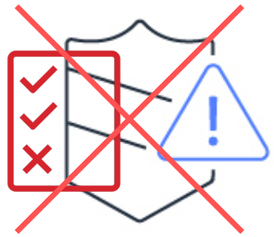
Used to delegate the permission to create users and roles by preventing privilege escalation or unnecessarily broad permissions.

The challenge…

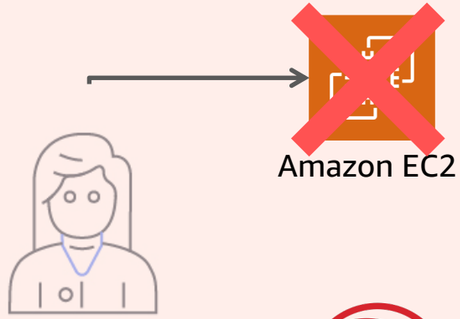
You grant Alice permission to create policies and roles



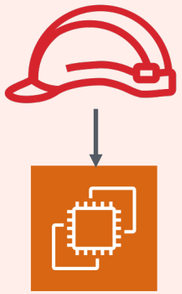
There is no way to restrict the permissions of the policies she creates



Even if Alice's permissions don't allow actions like ec2:RunInstances…



she could create a role that does

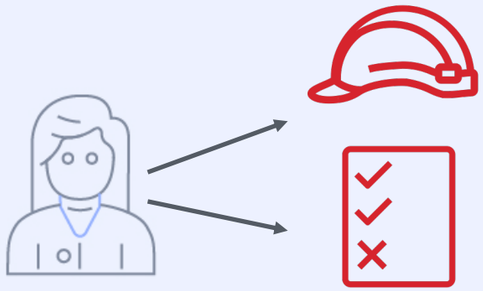


Or she may give a Lambda function more permissions then needed

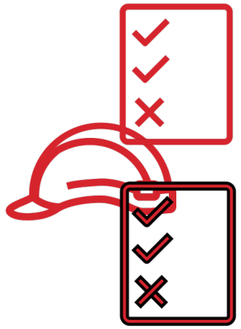
```
"Effect": "Allow",
"Action": “*”,
"Resource": “*”
```

. . .the solution.

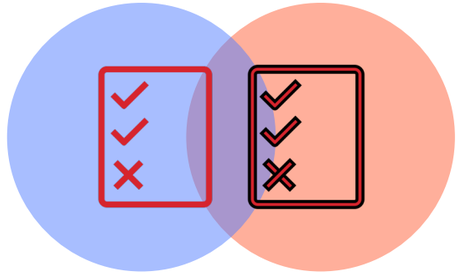
Go ahead and grant Alice the permission to create policies and roles



Require that another policy (permissions boundary) is also attached to the role



The effective permission of the role is the intersection of the two policies



In this way you can set the maximum permission of the roles Alice creates

Use cases

- Developers that need to create roles for Lambda functions
- Application owners that need to create roles for EC2 instances
- Admins that need to be able to create users for particular use cases
- Any others?

It's just a condition …

```
"Condition": {"StringEquals":
      {"iam:PermissionsBoundary":
      "arn:aws:iam::ACCOUNT_ID:policy/permissionboundary"
      }
}
```

. . . applied to principal actions

```
"Effect": "Allow",
"Action": ["iam:CreateRole"],
"Resource": ["arn:aws:iam::ACCOUNT_ID:role/path/*"],
"Condition": {"StringEquals":
      {"iam:PermissionsBoundary":
      "arn:aws:iam::ACCOUNT_ID:policy/permissionboundary"
      }
}
```

End user experience

Create role for a Lambda function

 # Step 1: Create role and attach permissions boundary

```
$ aws iam create-role –role-name Some_Role –path /Some_Path/
–assume-role-policy-document file://Some_Trust_Policy.json
–permissions-boundary arn:aws:iam::<ACCOUNT_NUMBER>:policy/Some_Path/Permissions_Boundary
```

 # Step 2: Create identity-based policy

No change

 # Step 3: Attach identity-based policy

No change

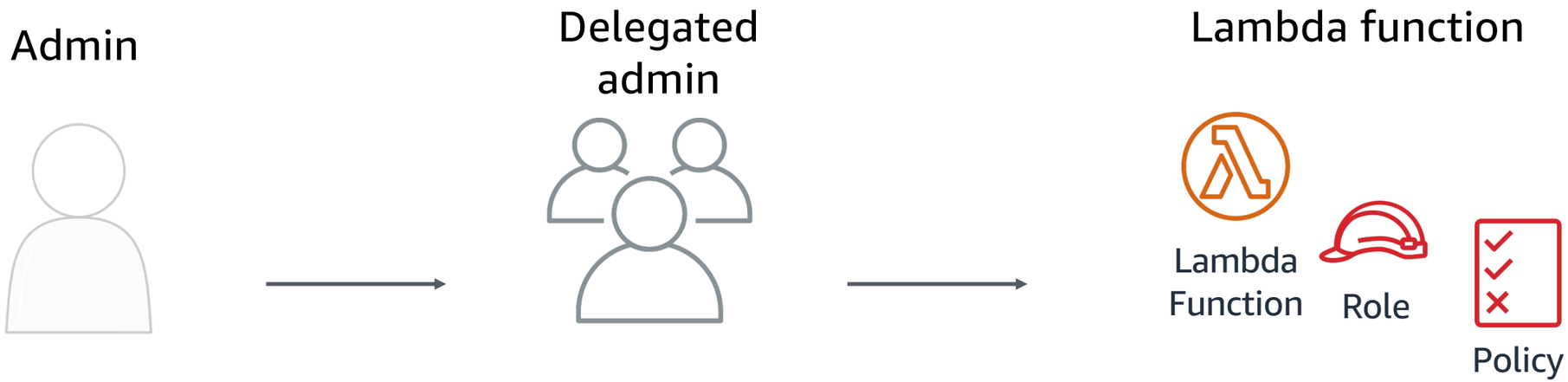
Demo

aws SUMMIT

Demo

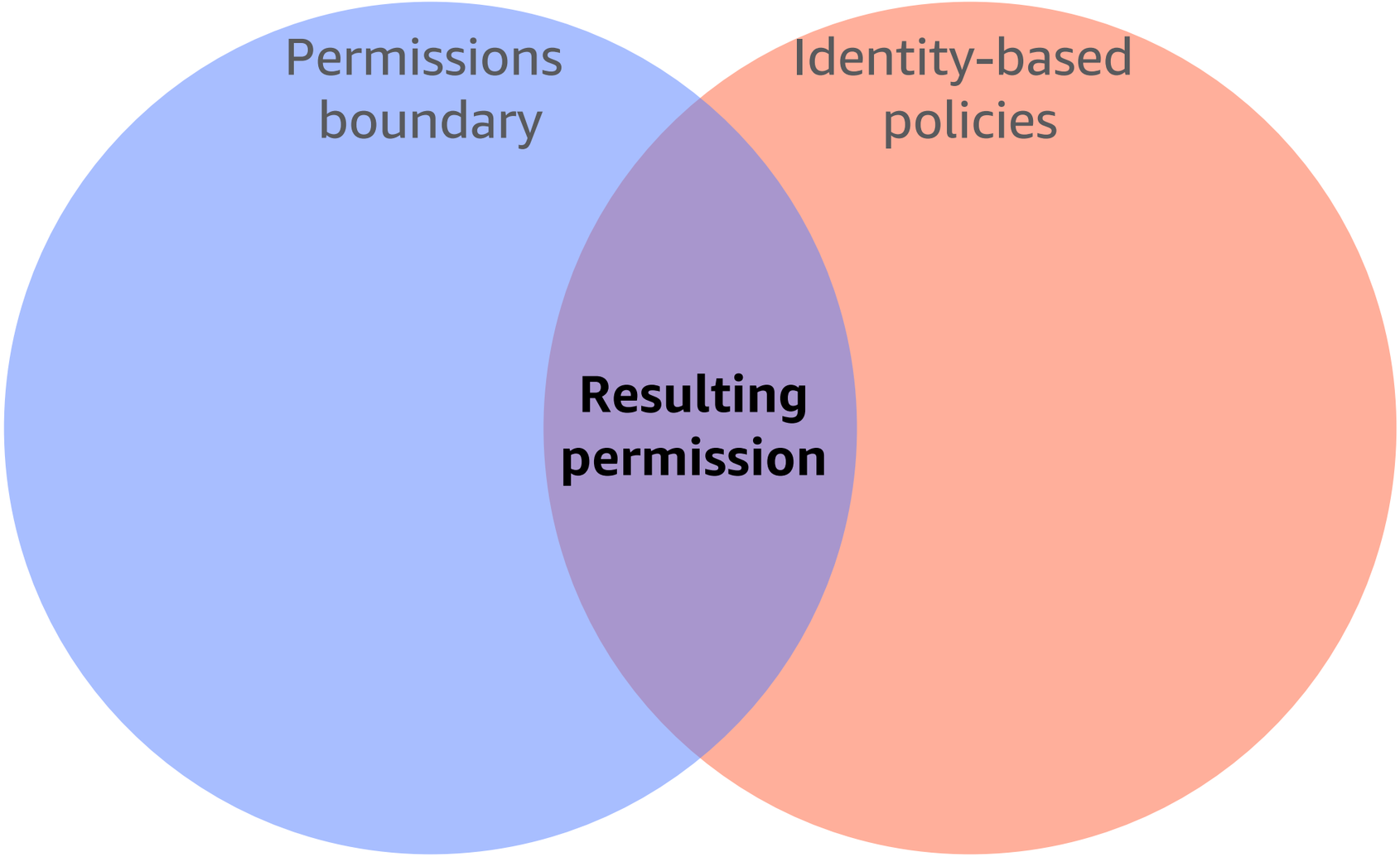
- **Admin requirements**:
  - Roles must not allow privilege escalation or grant unneeded permissions
  - Don’t get in the way of the developer

- **Delegated admin (developer) requirements**:
  - IAM policy to allow access to S3 bucket
  - IAM role to attach to a Lambda function
  - Lambda function that reads from an S3 bucket



Mechanism

Policy evaluation – Venn diagrams

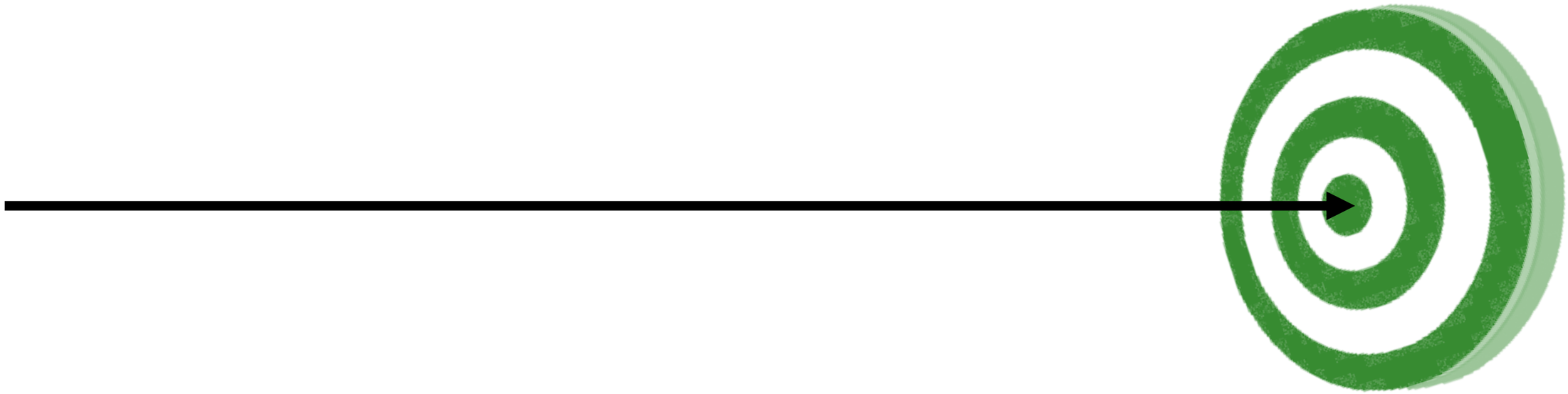


Policy evaluation – the archery analogy

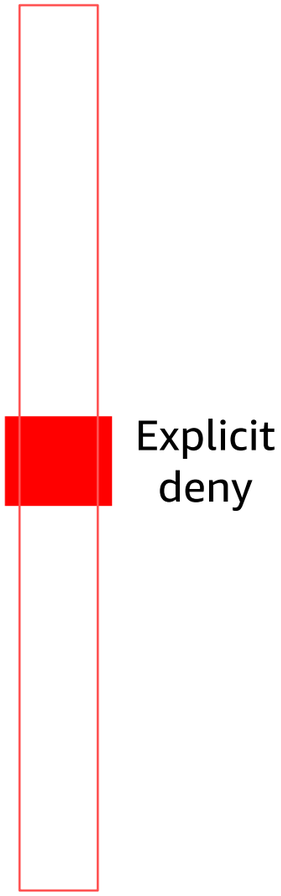
API
Request

API request trying to hit the target

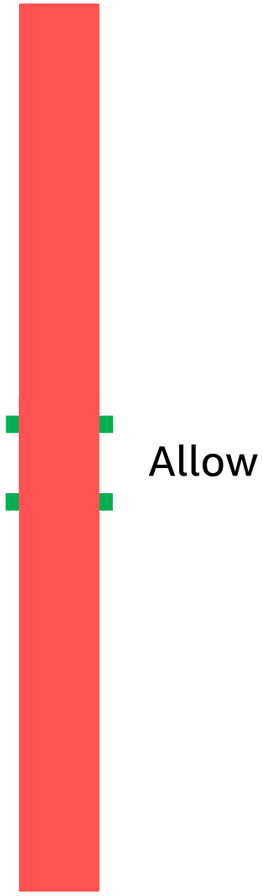
API
Request



Two types of obstacles



Explicit deny



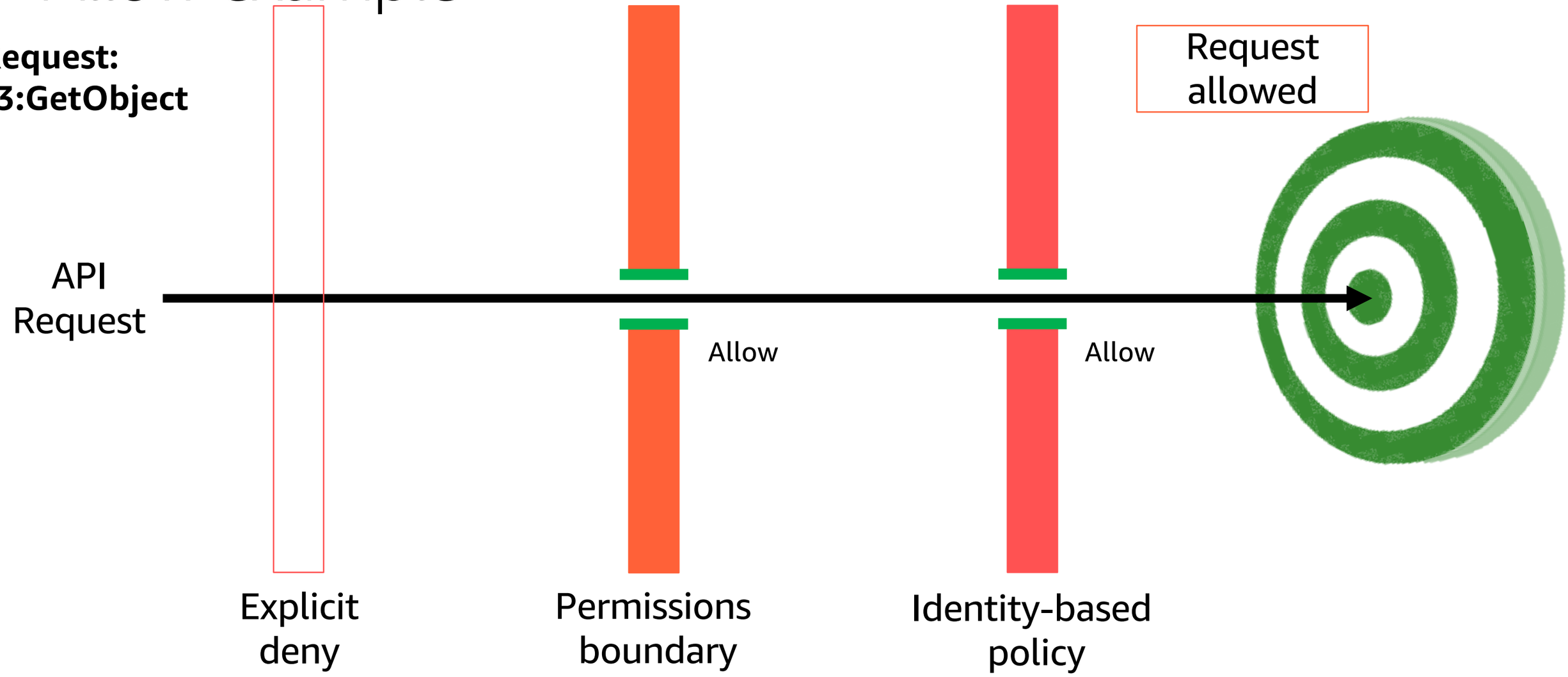
Allow

Explicit deny

Everything else

Allow example

Request:
s3:GetObject



Effective permissions – scenario 1

Request: s3:GetObject / bucket name: example1

Permissions boundary

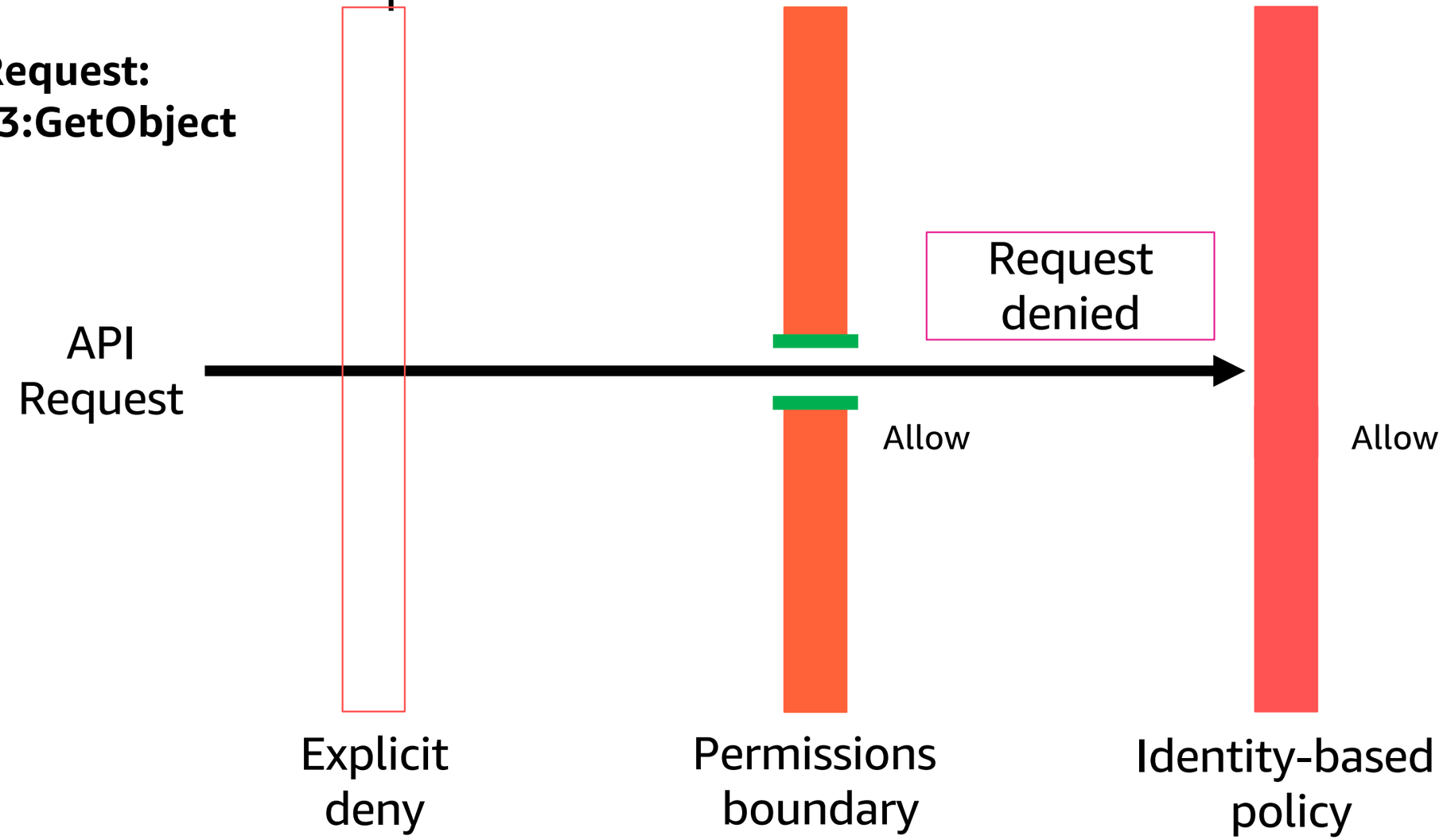
```
{
  "Version": "2012-10-17",
  "Statement": [
     {
       "Effect": "Allow",
       "Action": [
         "logs:CreateLogGroup",
         "logs:CreateLogStream”,
         "logs:PutLogEvents”
       ],
     "Resource": "arn:aws:logs:*:*:*"
     },
     {
         "Effect": "Allow",
      "Action": ["s3:GetObject"],
      "Resource”:"arn:aws:s3:::example1/*"
     }
}
```

Identity-based policy

```
{
  "Version": "2012-10-17",
  "Statement": [
     {
       "Effect": "Allow",
       "Action": [
         "logs:CreateLogGroup",
         "logs:CreateLogStream",
         "logs:PutLogEvents",
           ],
      "Resource": "*"
      }
    ]
}
```

Effective permissions – scenario 1

Request:
s3:GetObject



Effective permissions – scenario 2

Request: s3:GetObject / bucket name: example1

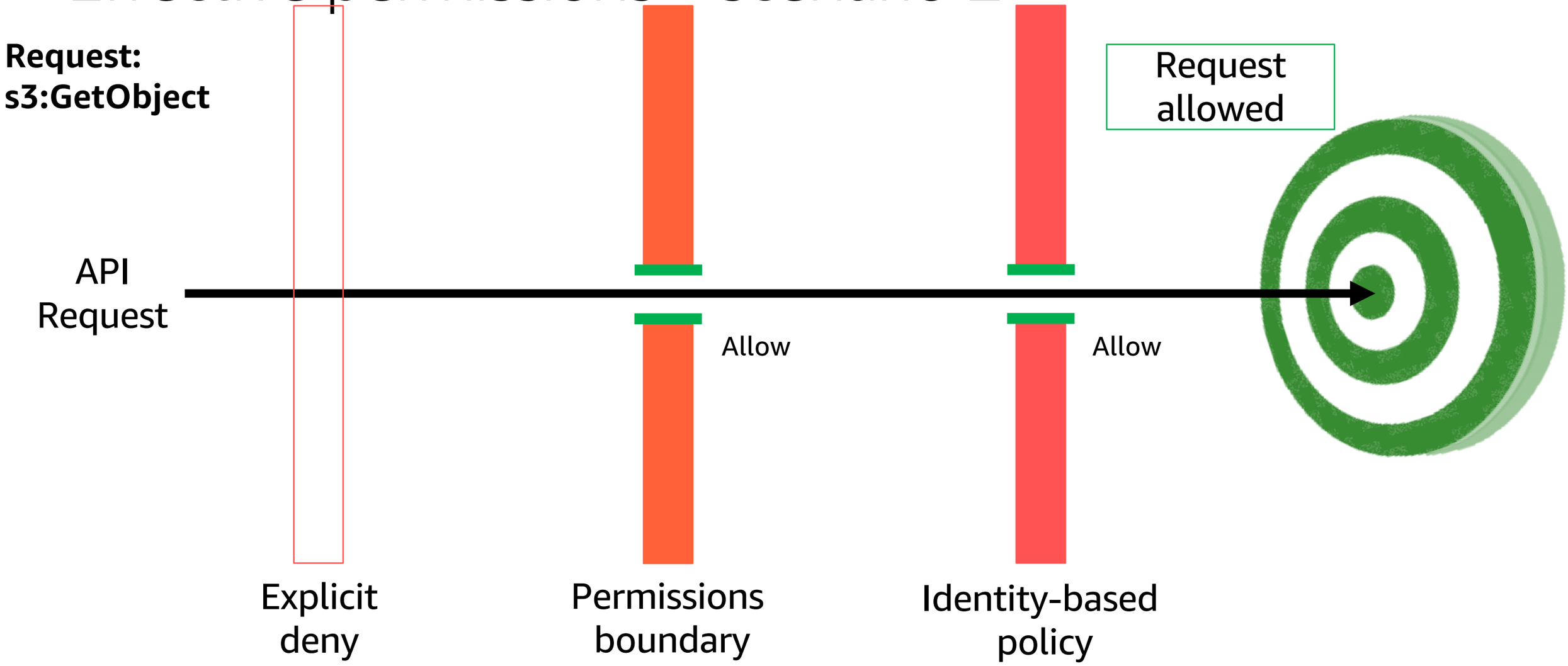
Permissions boundary

```
{
  "Version": "2012-10-17",
  "Statement": [
     {
       "Effect": "Allow",
       "Action": [
         "logs:CreateLogGroup",
         "logs:CreateLogStream”,
         "logs:PutLogEvents”
       ],
     "Resource": "arn:aws:logs:*:*:*"
     },
     {
         "Effect": "Allow",
      "Action": ["s3:GetObject"],
      "Resource”:"arn:aws:s3:::example1/*"
     }
}
```

Identity-based policy

```
{
  "Version": "2012-10-17",
  "Statement": [
     {
       "Effect": "Allow",
       "Action": [
         "logs:CreateLogGroup",
         "logs:CreateLogStream",
         "logs:PutLogEvents",
         "s3:*"
        ],
      "Resource": "*"
      }
    ]
}
```

Effective permissions – scenario 2



Resource Restrictions

aws SUMMIT

Resource Restrictions

Goal: create a “walled garden” for a team of delegated admins to be able to do their job without impacting resources of other teams.

Paths are preferred but require the CLI.

Naming can also be used.

Resource Restrictions - paths

Role: arn:aws:iam::123456789012:role/**webadmins**

Role with a path: arn:aws:iam::123456789012:role/**namer/webadmins**

Role with paths: arn:aws:iam::123456789012:role/**namer/dept1/webadmins**

Permission:

"Effect": "Allow",

"Action": "iam:DeleteRole”,

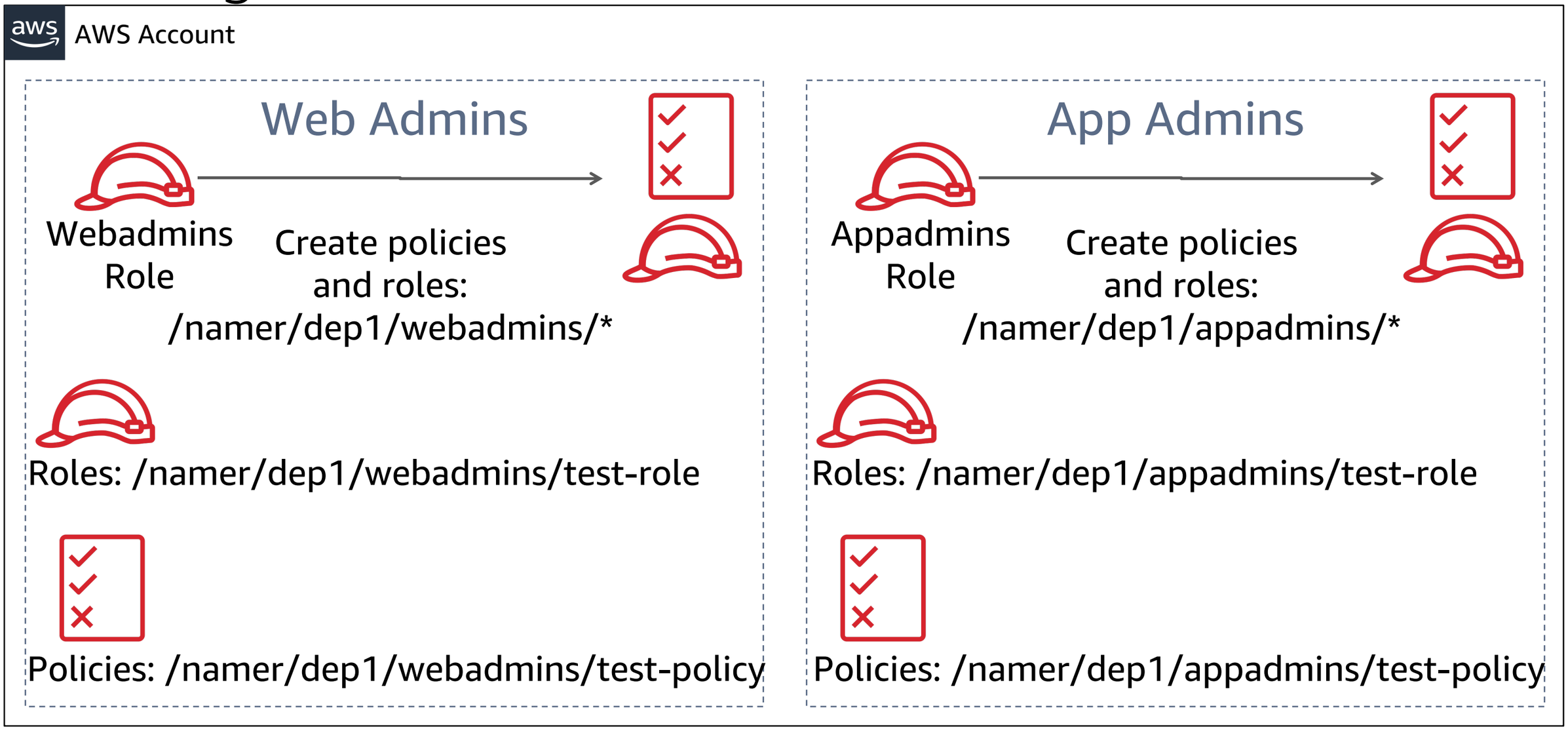
"Resource": "arn:aws:iam::123456789012:**/namer/dept1/***”

or "Resource": "arn:aws:iam::123456789012:**/namer/***”

Command:

```
aws iam create-role --role-name webadmin --path /namer/dept1/ --assume-role-policy-document file://policydoc
```

Pathing Walled Garden



Resource Restrictions - paths

Basic path example:

arn:aws:iam::123456789012:role/**webadmins/????**

Naming example:

arn:aws:iam::123456789012:role/**webadmins***

Support for the permissions boundary condition context key

- AttachRolePolicy
- AttachUserPolicy
- CreateRole
- CreateUser
- DeleteRolePermissionsBoundary
- DeleteUserPermissionsBoundary
- DeleteRolePolicy
- DeleteUserPolicy
- DetachRolePolicy
- DetachUserPolicy
- PutRolePermissionsBoundary
- PutUserPermissionsBoundary
- PutRolePolicy
- PutUserPolicy

Q&A

aws SUMMIT

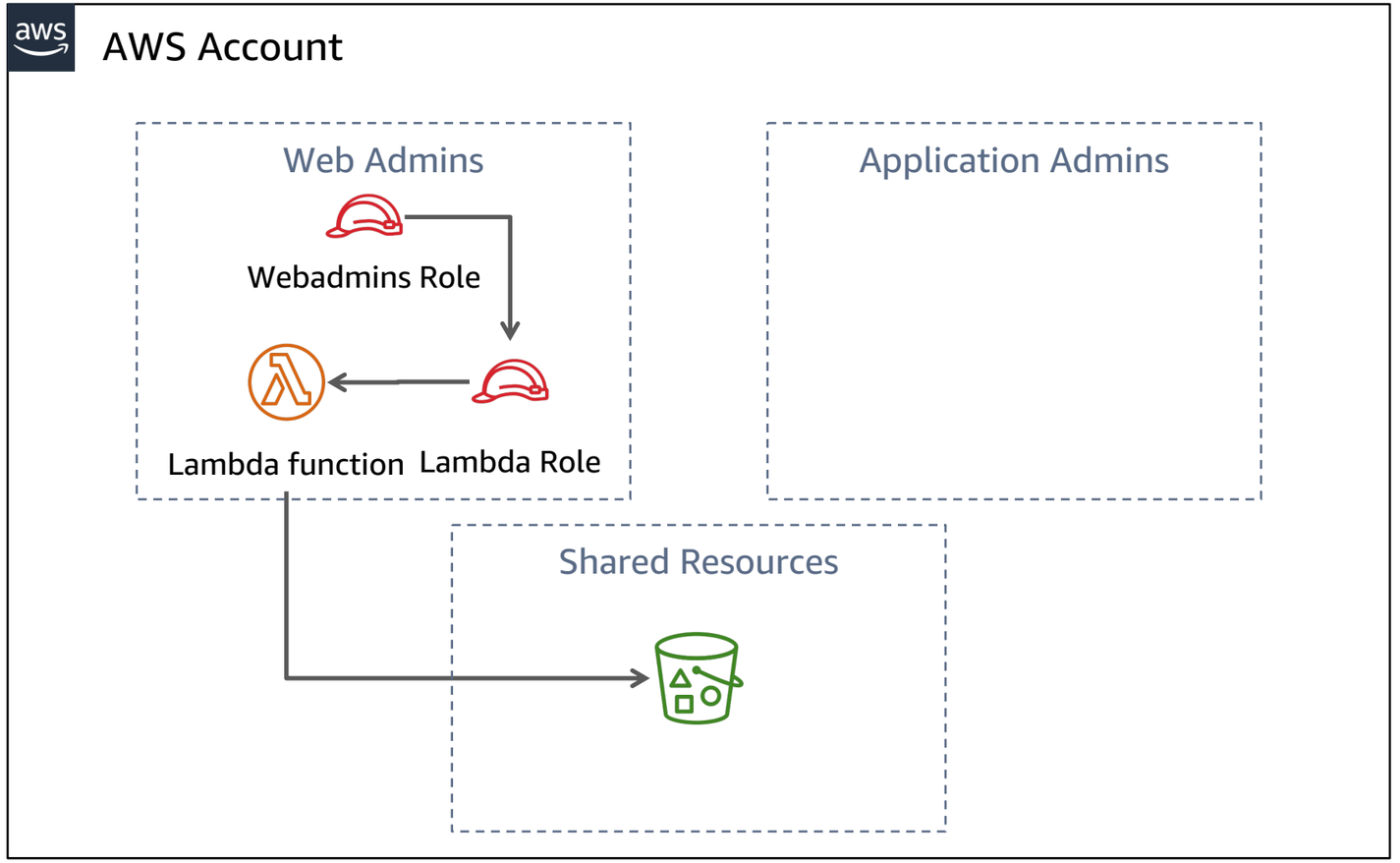
End of presentation questions

- What is the condition context key used for permissions boundaries?
- What are some of the advantages of using pathing for policies, users and roles?
- What are some permissions boundaries use cases?

Hands On

aws SUMMIT

Workshop setup



Permissions boundaries – workflow reminder

Admins
(Build phase)

Create delegated admins

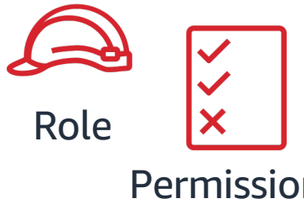
Requirement: users and roles created by delegated admins must have a permissions boundary

Delegated admins
(Verify phase)

Create “bound” users & roles

Ability: can create users and roles that have permissions boundaries attached

“Bound” IAM
roles



Role restricted by permissions boundaries

Result: Permissions boundary restrict the permissions of the role

Restricted
resources

Permissions for resources restricted

Permissions of the roles attached to resources like Lambda functions are limited by the permissions boundary

Permissions Boundaries Workshop

Build Phase (60 min)

Use: US East (Ohio)
us-east-2

https://bit.ly/2CMjqmh

Read through the **Overview**, then click on **Build Phase** and run the CloudFormation template:
https://identity-round-robin.awssecworkshops.com/permission-boundaries-advanced/

aws SUMMIT

Permissions Boundaries Workshop

Verify Phase (15 min)

Use: US East (Ohio)
us-east-2

https://bit.ly/2CMjqmh

Click on **Verify Phase:**
https://identity-round-robin.awssecworkshops.com/permission-boundaries-advanced/

aws SUMMIT

Final Q&A

aws SUMMIT

End of workshop questions

- What is the risk of implementing permissions boundaries without resource restrictions?
- What do you attach the permissions boundary to?
- How does a permissions boundary differ from an IAM policy?