



AWS
re:Invent

SEC 341 - R

Set permission guardrails for multiple accounts in AWS Organizations

Michael Switzer

Senior Product Manager
AWS Identity

Toby Latin-Stoermer

Senior Software Development Manager
AWS Identity

Agenda

Service control policy overview

Example 1: Restrict access to AWS services

Example 2: Control the AWS regions your accounts operate in

Example 3: Disable root access in an AWS account

Q&A

Related breakouts

SEC209 Getting started with AWS Identity

SEC316 Access control confidence: Grant the right access to the right things

SEC326 AWS identity-dynamic permissions using employee attributes

SEC402 Permissions Boundaries and Delegation

MGT307 Governance at scale: AWS Control Tower, AWS Organizations, and more

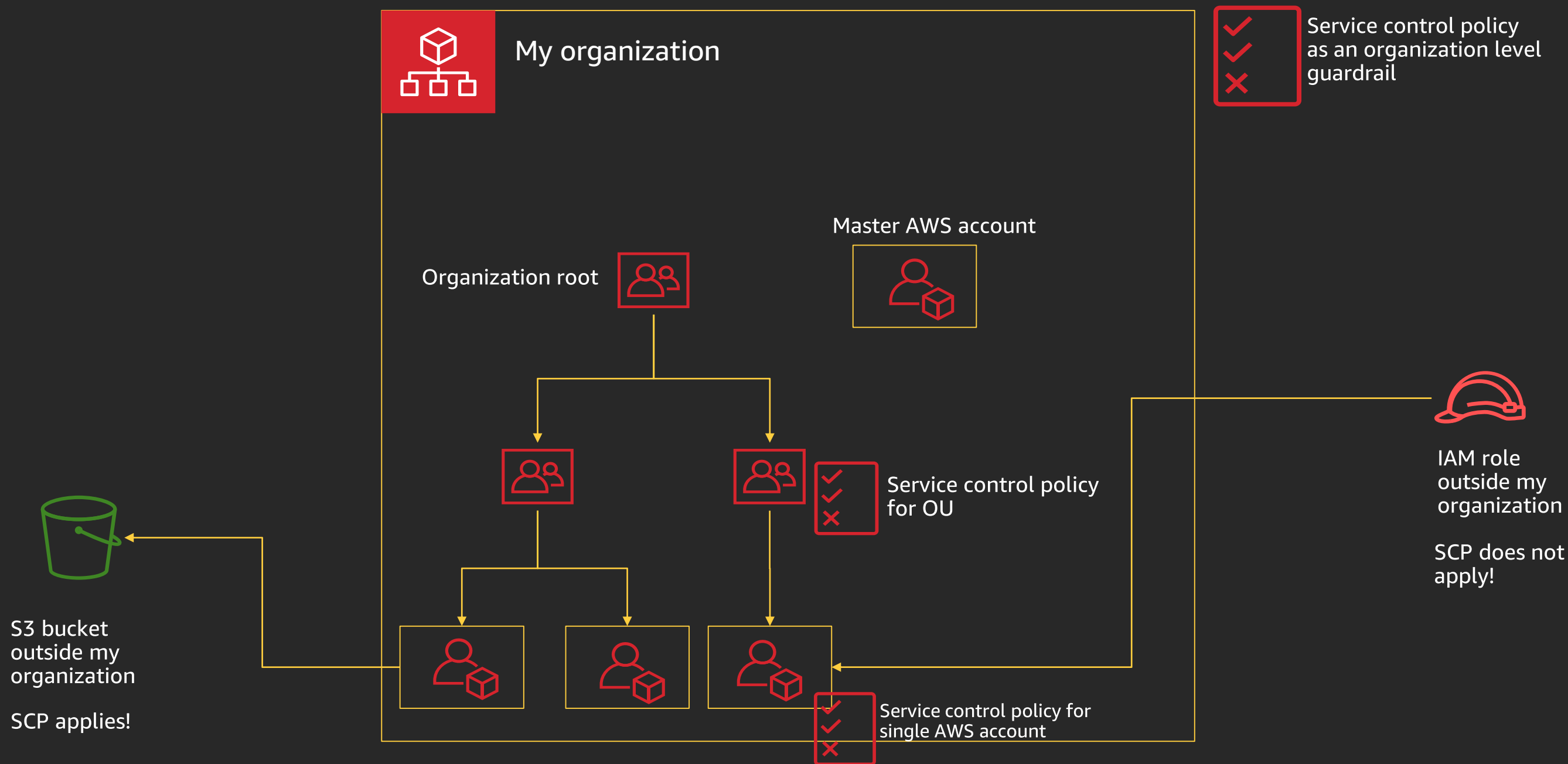
MGT313 Architect governance at enterprise scale with Goldman Sachs

Overview: Service control policies

Service control policies overview

- Organization-scoped access control policy
 - Define the maximum available permissions for IAM identities in your account(s)
 - SCPs use the IAM policy language
- SCPs make it easy to:
 - Govern permissions with total authority over your AWS accounts
 - Apply guardrails to prevent unintended access or meet compliance standards
 - Safely delegate permissions management tasks to your developers

Service Control Policies Overview



Service control policies overview

- Policies with allow effect . . .
 - Supported elements: Effect, Action
 - Use to establish baseline services & actions allowed for your account(s)
- Policies with deny effect . . .
 - Supported elements: Effect, Action, **NotAction**, **Resource**, **Condition**
 - Use to apply guardrails and restrict certain activities

Policy examples

Prerequisites

1. Create an organization with at least one member account
2. Enable “all features” mode in AWS Organizations
3. Enable the “Service Control Policy” policy type

Example 1: Restrict access to AWS services

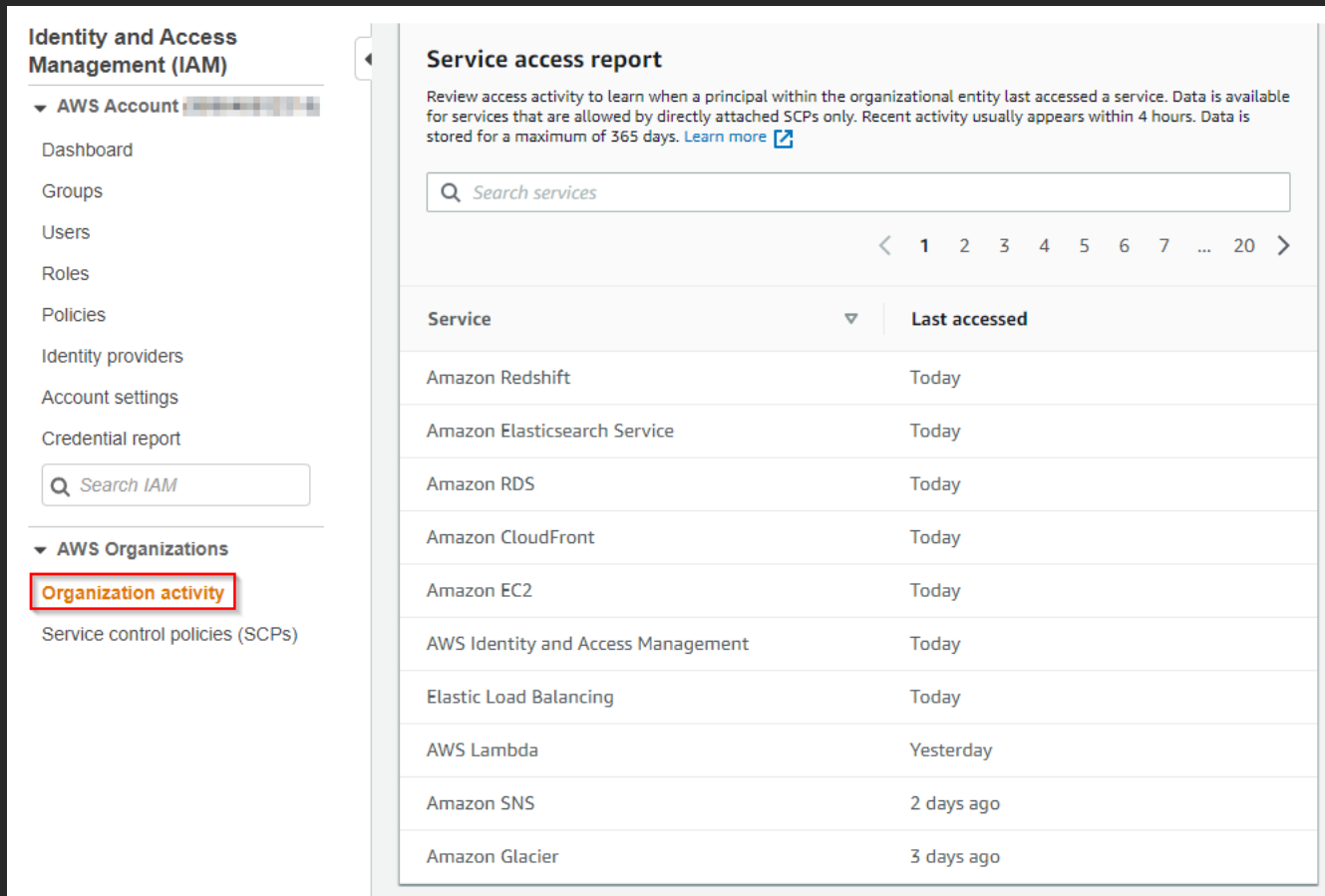
Situation: Following the IAM best practice of granting least privilege, I want to prevent my accounts from using any AWS services that my team doesn't use today.

Approach: Identify the services my team is using with IAM access advisor. Apply an SCP to enable those services and restrict access to all other activity.

Example 1: Restrict access to AWS services

Step 1: Use IAM access advisor to identify the services I am using in my organization

AWS Console:



The screenshot shows the AWS IAM console interface. On the left, the 'Identity and Access Management (IAM)' sidebar is visible, with 'AWS Organizations' expanded and 'Organization activity' highlighted. The main content area displays the 'Service access report' for the selected organization. It includes a search bar, a pagination control showing 20 items, and a table of services accessed.

Service	Last accessed
Amazon Redshift	Today
Amazon Elasticsearch Service	Today
Amazon RDS	Today
Amazon CloudFront	Today
Amazon EC2	Today
AWS Identity and Access Management	Today
Elastic Load Balancing	Today
AWS Lambda	Yesterday
Amazon SNS	2 days ago
Amazon Glacier	3 days ago

AWS CLI:

```
> aws iam generate-organizations-access-report \
    --entity-path o-orgid/r-rootid
```

Response:

```
{
  "JobId": "2eb6c2e6-0xmp-ec04-1425-c937916a64af"
}
```

```
> aws iam get-organizations-access-report \
    --job-id 2eb6c2e6-0xmp-ec04-1425-c937916a64af
```

Example 1: Restrict access to AWS services

Step 2: Attach an Allow-based SCP to enable the services you identified in access advisor

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "EnableApprovedServices",
      "Effect": "Allow",
      "Action": [
        "iam:*",
        "rds:*",
        "...",
        "glacier:*"
      ],
      "Resource": "*",
    }
  ]
}
```

This policy does not grant any permissions!

Policies with Allow effect are translated to:

"Effect": "Deny",
"NotAction": []

Example 1: Restrict access to AWS services

Step 3: If you have not done already, remove the *AWSFullAccess* policy from your organization root

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

This policy is automatically applied when you enable SCPs, to prevent an unintended service interruption for your organization. It allows all services and actions.

Example 2: Control access to AWS regions

Situation: Due to compliance or regulatory requirements, I want to control which AWS regions my accounts are allowed to operate in.

Approach: Apply an SCP that restricts access to the regions I don't want in my account. Then, restrict ability to enable more regions in my accounts.

Example 2: Control access to AWS regions

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "DenyAllEUOperations",
      "Effect": "Deny",
      "Action": "*",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:RequestedRegion": [
            "eu-central-1",
            "eu-west-1",
            "eu-west-2",
            "eu-west-3",
            "eu-north-1"
          ]
        }
      }
    }
  ]
}
```

Deny all AWS actions . . .

If the request context includes these regions

Example 2: Control access to AWS regions

- **Why not list your allowed regions instead?**
 - Some AWS services and actions are global (regionless)
 - IAM, Amazon Route 53, AWS Global Accelerator
 - If you use a whitelist, you need to exclude these services from your policy
- **With the blacklist policy, how do I catch new regions in my filter?**
 - Starting in 2019, all new AWS regions are opt-in
 - New regions won't be available until you enable them in your account
 - You can use SCPs to control how your accounts opt in to regions!

Example 2 (cont'd): Disable region enablement

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "DenyRegionEnableDisable",
      "Effect": "Deny",
      "Action": [
        "account:EnableRegion",
        "account:DisableRegion"
      ],
      "Resource": "*",
      "Condition": {
        "StringNotEquals": {
          "account:TargetRegion": "ap-east-1"
        }
      }
    }
  ]
}
```

Deny the actions to enable or disable a region

... Unless that region is on my approved list

Example 3: Restrict access to root credentials

Situation: The root user is capable of performing any AWS action. After my initial account configuration, I do not need the root user. I would like to prevent the root user from being used inside my accounts.

Approach: Use an SCP to deny access to AWS actions if the root user is making a request.

Example 3: Restrict access to root credentials

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "DenyRootActivity",
      "Effect": "Deny",
      "Action": "*",
      "Resource": "*",
      "Condition": {
        "StringLike": {
          "aws:PrincipalARN": [
            "arn:aws:iam::*:root"
          ]
        }
      }
    }
  ]
}
```

Deny all AWS actions . . .

If the request is made by the root user

Example 3: Restrict access to root credentials

Important!

Some activities do not have AWS actions associated with them. This policy will not prevent the root credentials from being used to perform these activities. This includes:

- Changing support level
- Account settings
- Account recovery/closure

Next steps and other policies to try

- Secure access to resources inside your Amazon Virtual Private Cloud (VPC)
- Protect tag keys and other attributes used for attribute-based access controls
- Restrict sensitive operations, like changing VPC settings, to specific IAM identities
- Use conditions like `ec2:instancetype` to apply cost-saving controls
- These and many more examples are available in the documentation

Q&A

Learn security with AWS Training and Certification

Resources created by the experts at AWS to help you build and validate cloud security skills



30+ free digital courses cover topics related to cloud security, including Introduction to Amazon GuardDuty and Deep Dive on Container Security



Classroom offerings, like AWS Security Engineering on AWS, feature AWS expert instructors and hands-on activities



Validate expertise with the **AWS Certified Security - Specialty** exam

Visit aws.amazon.com/training/paths-specialty/

Thank you!

Michael Switzer

switzm@amazon.com



Please complete the session
survey in the mobile app.