
AWS
re:Inforce
JULY 26 – 27, 2022 | BOSTON, MA

IAM202

Getting more out of your service control policies, featuring Morgan Stanley

Swara Gandhi
Solutions Architect
Amazon Web Services

Itay Cohai
Executive Director
Morgan Stanley

Zulfi Haider
Vice President
Morgan Stanley

Agenda

Overview of service control policies (SCPs)

Using SCPs effectively

Getting the most out of your SCPs

Our top 5 recommended SCPs

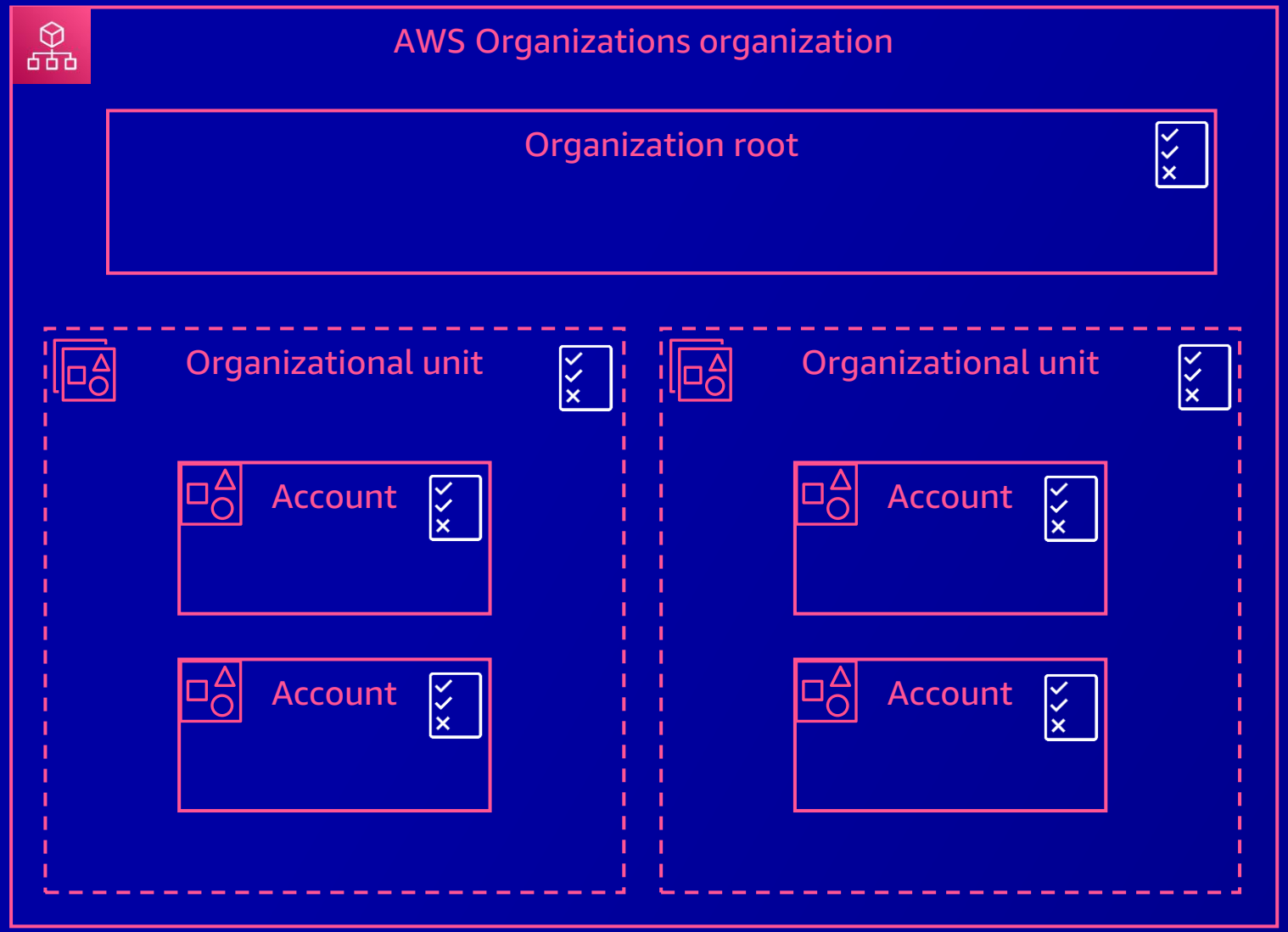
Morgan Stanley AWS organization/SCP evolution

Lessons learned

Use cases

Overview of SCPs

Set preventive guardrails with SCPs

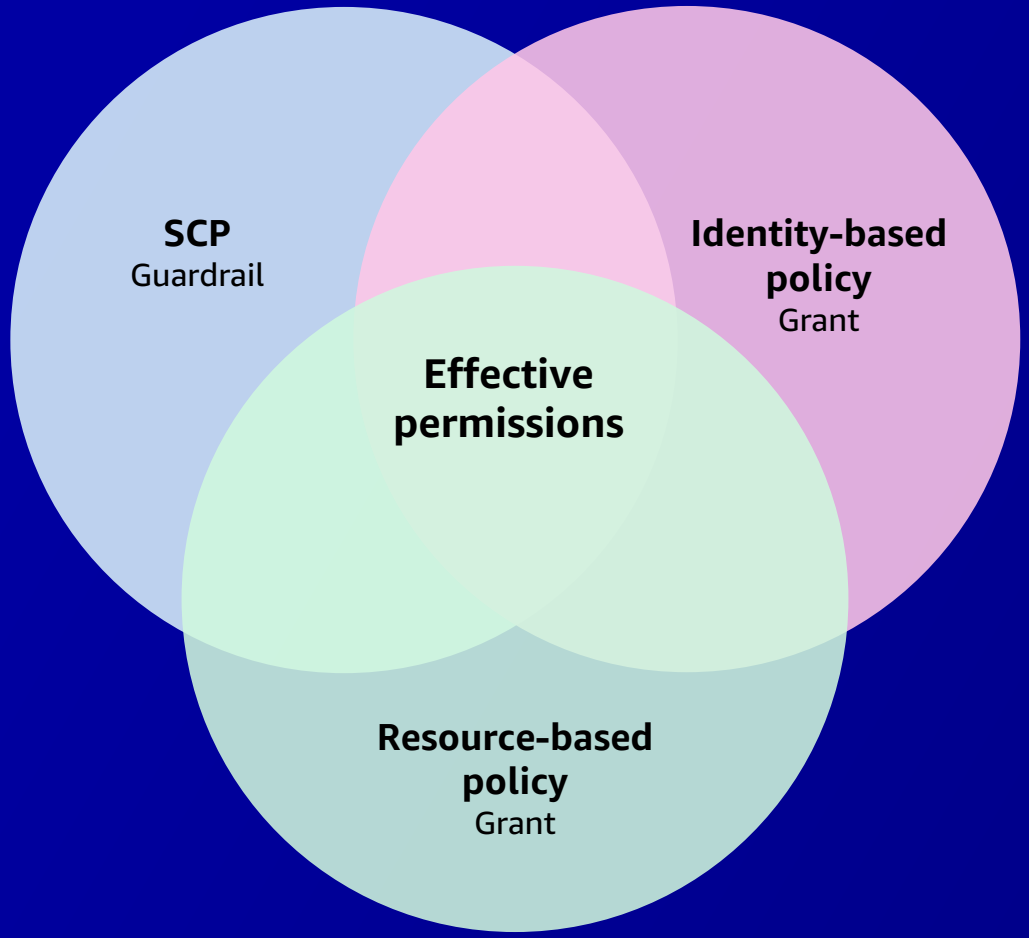


SCPs

- Act as preventive guardrails for principals in your organization (including root user)
- Are used to enforce security invariants

= SCP

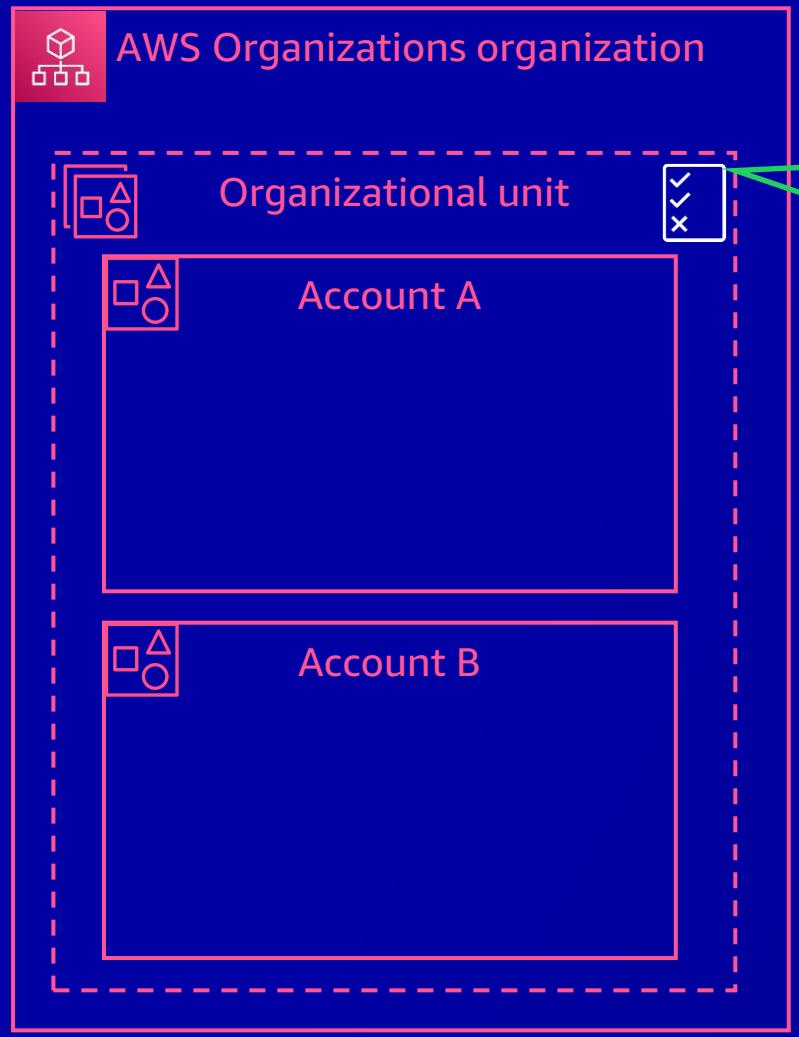
SCPs define maximum allowable permissions



* SCPs only affect principals within your AWS Organizations organization

SCPs apply to your principals

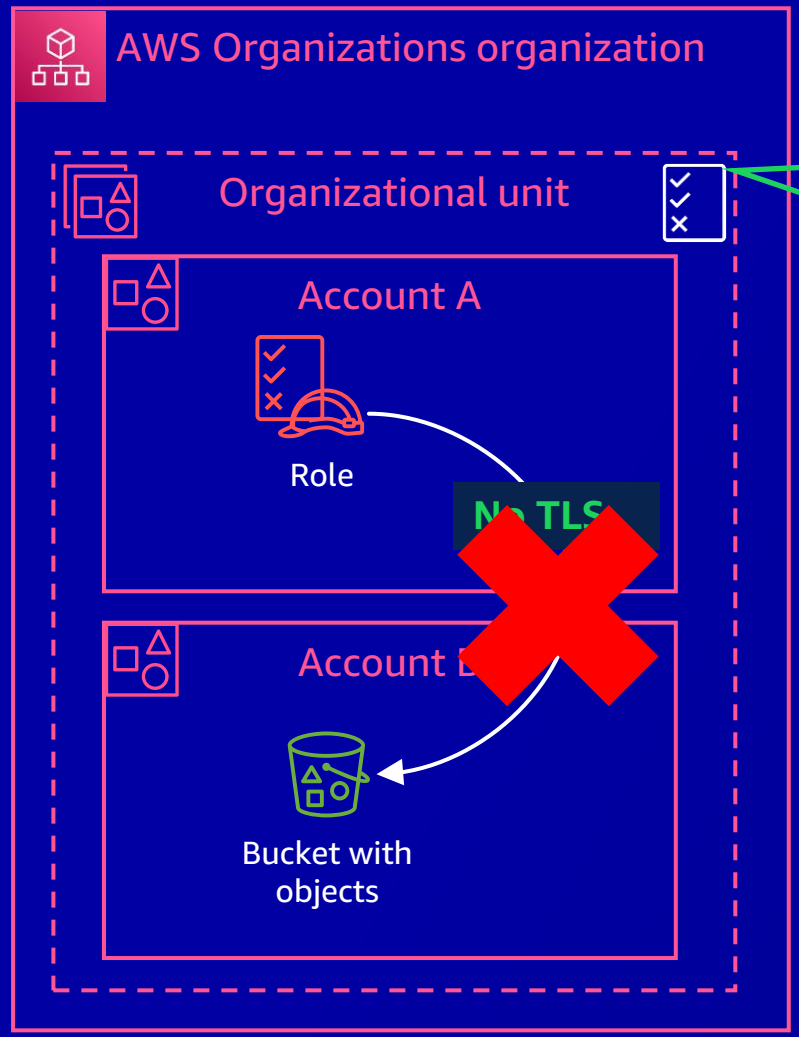
. . . AND ONLY YOUR PRINCIPALS




```
"Effect": ”Deny",
"Action": "s3:*",
"Resource": "*",
"Condition" : {
    "Bool" : {
        "aws:SecureTransport": "false"
    }
}
```

SCPs apply to your principals

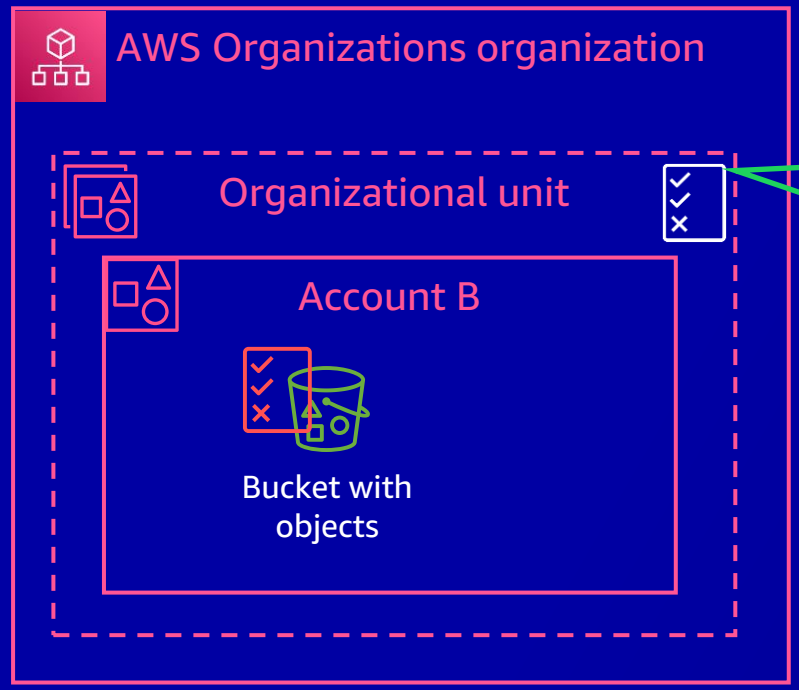
. . . AND ONLY YOUR PRINCIPALS




```
"Effect": ”Deny",
"Action": "s3:*",
"Resource": "*",
"Condition" : {
    "Bool" : {
        "aws:SecureTransport": "false"
    }
}
```

SCPs apply to your principals

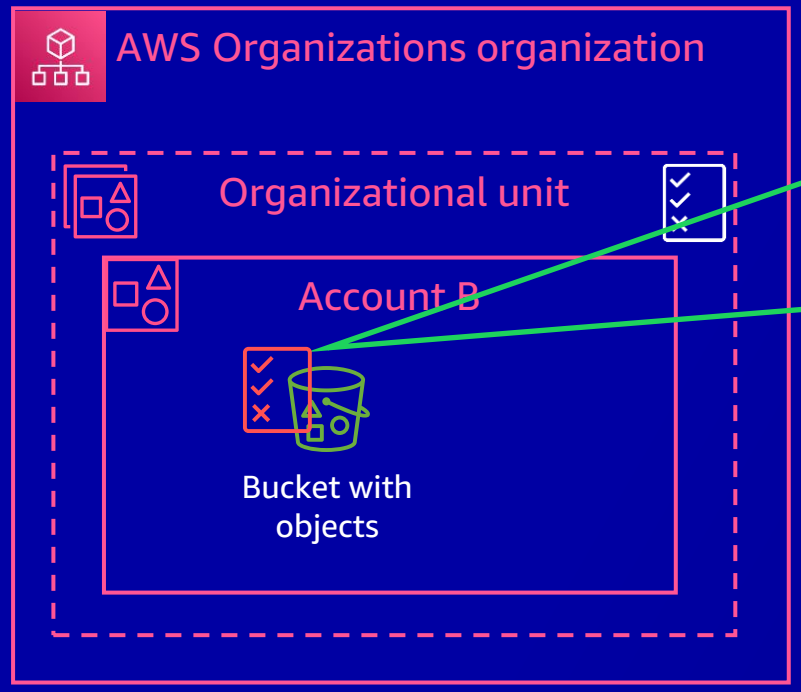
. . . AND ONLY YOUR PRINCIPALS




```
"Effect": ”Deny",
"Action": "s3:*",
"Resource": "*",
"Condition" : {
    "Bool" : {
        "aws:SecureTransport": "false"
    }
}
```

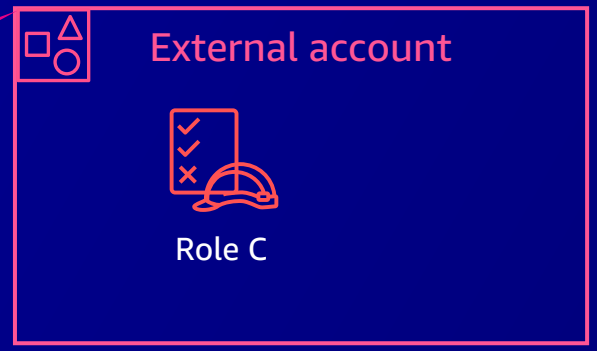
SCPs apply to your principals

. . . AND ONLY YOUR PRINCIPALS



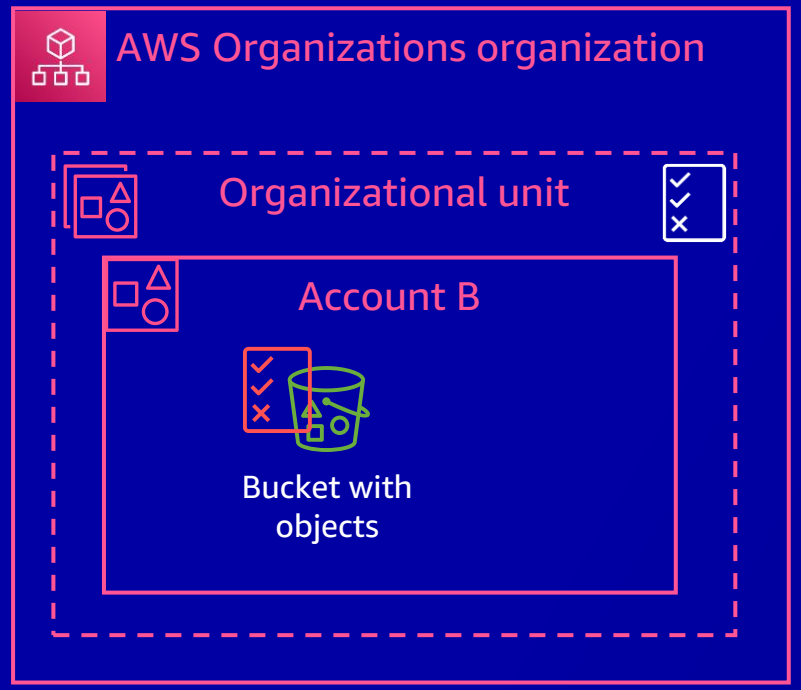

```
"Effect": ”Allow",
”Principal": {
    "AWS": ”arn:aws:iam:ExternalAccount:role/RoleC
    "Action": "s3:*",
    "Resource": [
        "arn:aws:s3:::Workload-B-Bucket/*",
        "arn:aws:s3:::Workload-B-Bucket”
    ]
}
```

Account outside of your organization

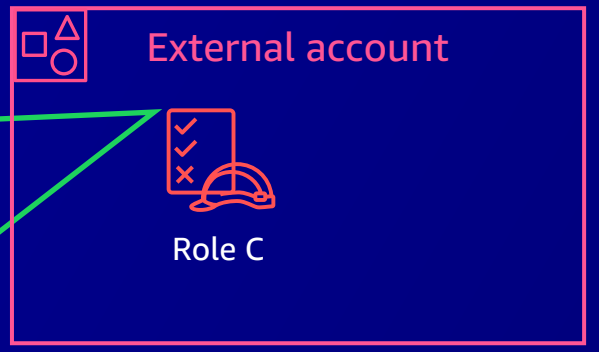


SCPs apply to your principals

. . . AND ONLY YOUR PRINCIPALS

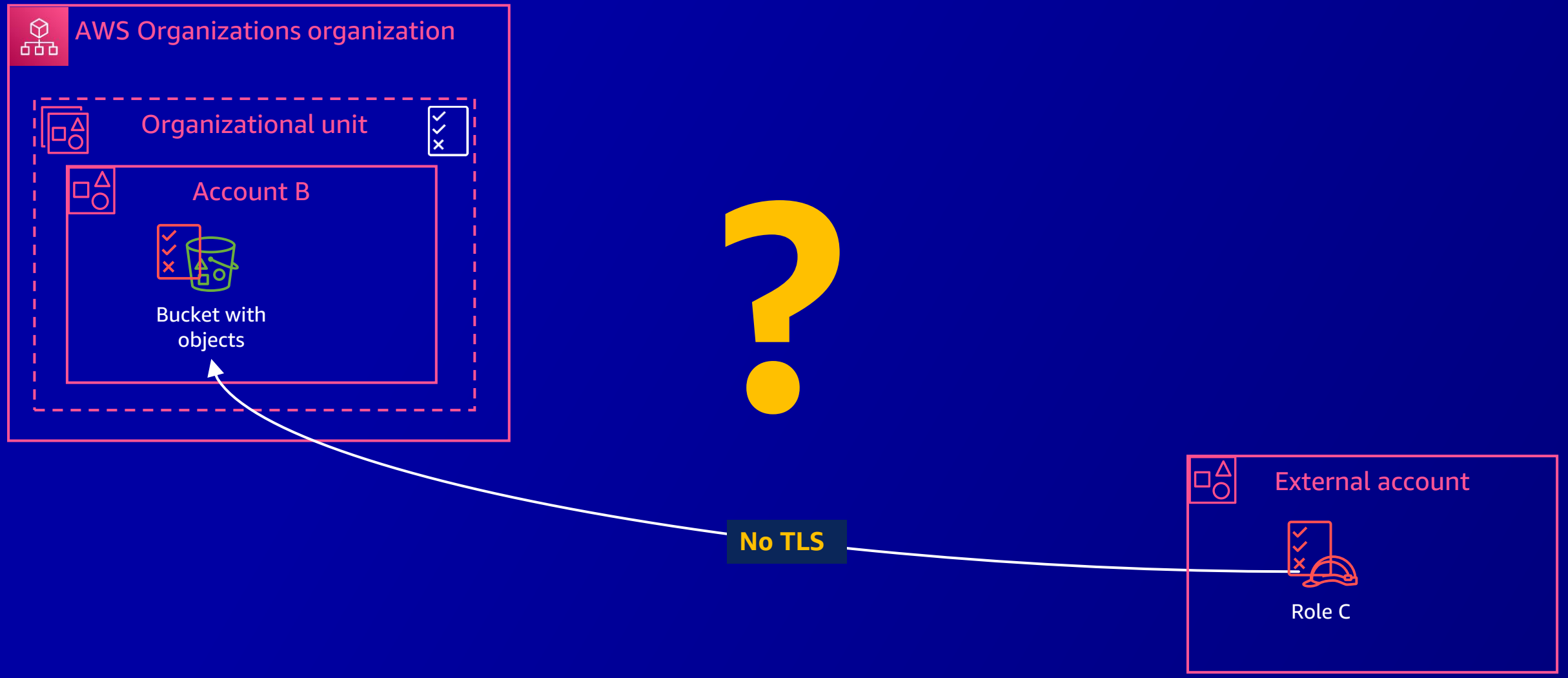



```
"Effect": ”Allow",
"Action": "s3:*",
"Resource": [
    "arn:aws:s3:::Workload-B-Bucket/*",
    "arn:aws:s3:::Workload-B-Bucket”
]
```



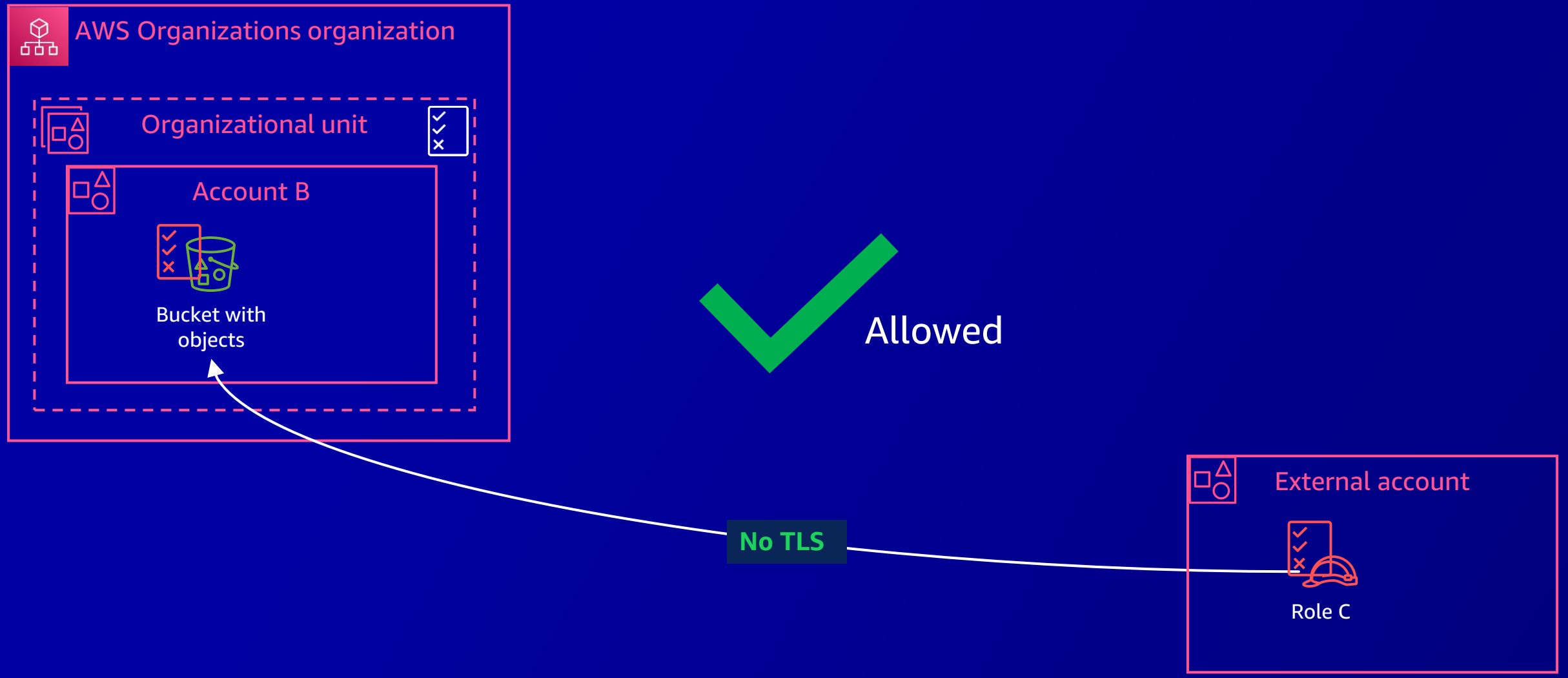
SCPs apply to your principals

. . . AND ONLY YOUR PRINCIPALS

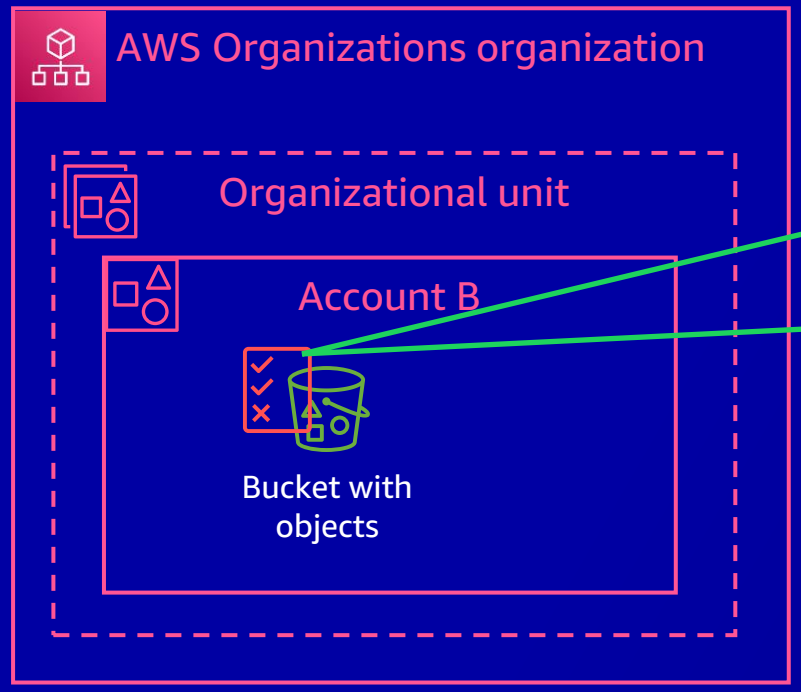


SCPs apply to your principals

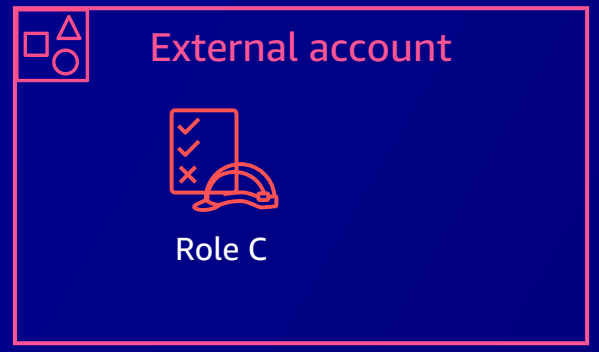
. . . AND ONLY YOUR PRINCIPALS



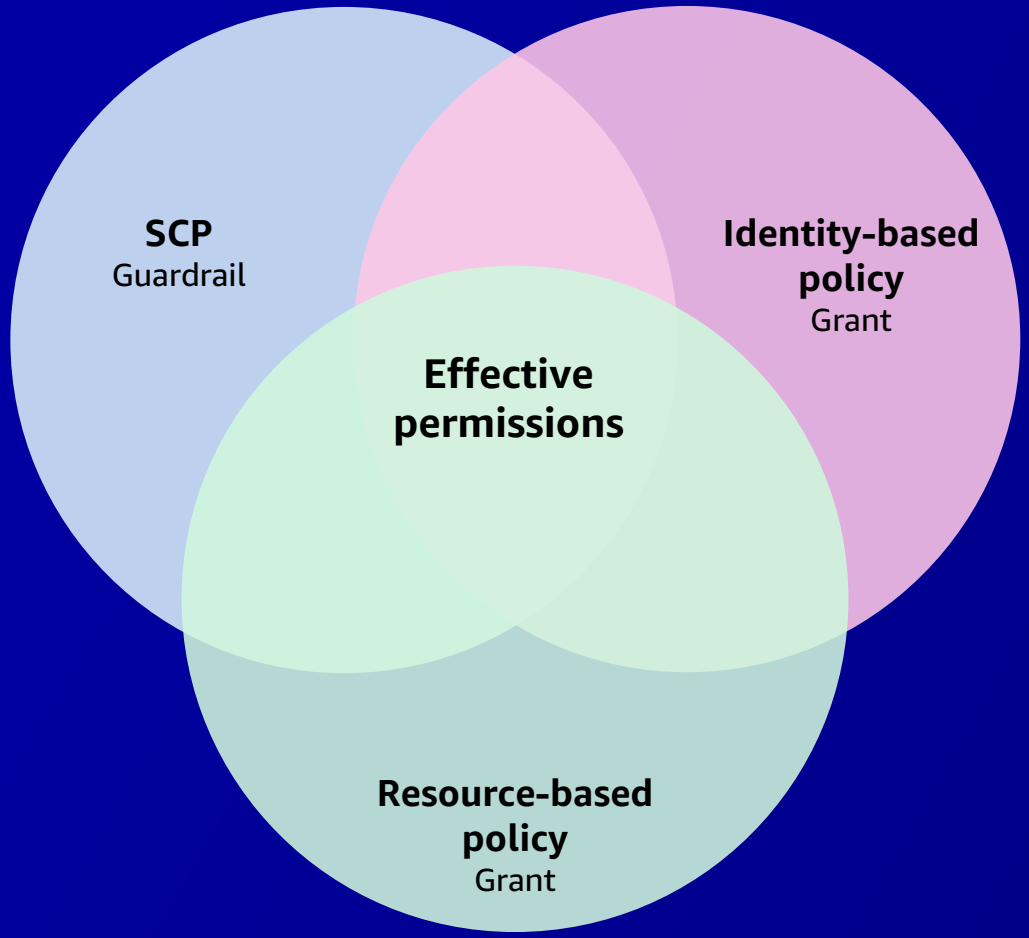
Restrict external principals using resource policy




```
"Effect": "Deny",
"Principal": "*",
"Action": "s3:*",
"Resource": [
    "arn:aws:s3:::Workload-B-Bucket/*",
    "arn:aws:s3:::Workload-B-Bucket"
     ],
"Condition": {
     "Bool": {
        "aws:SecureTransport": "false"
    }
}
```

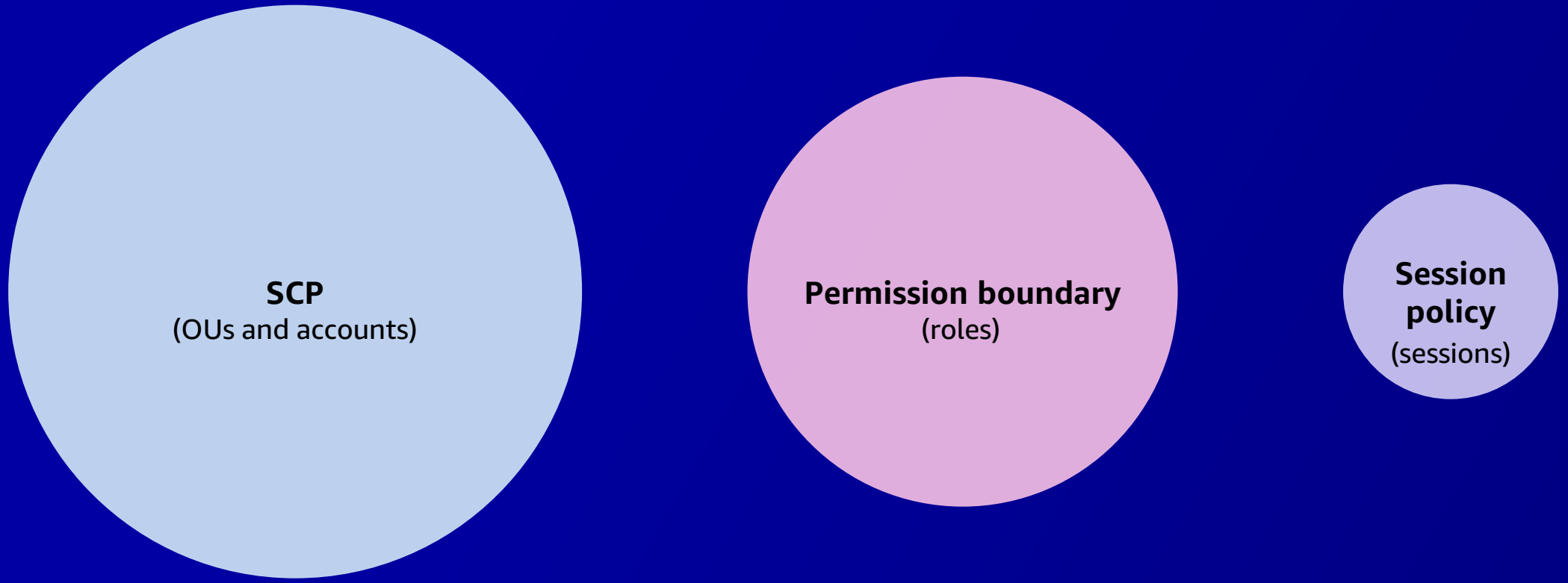


SCPs affect principals in your organization

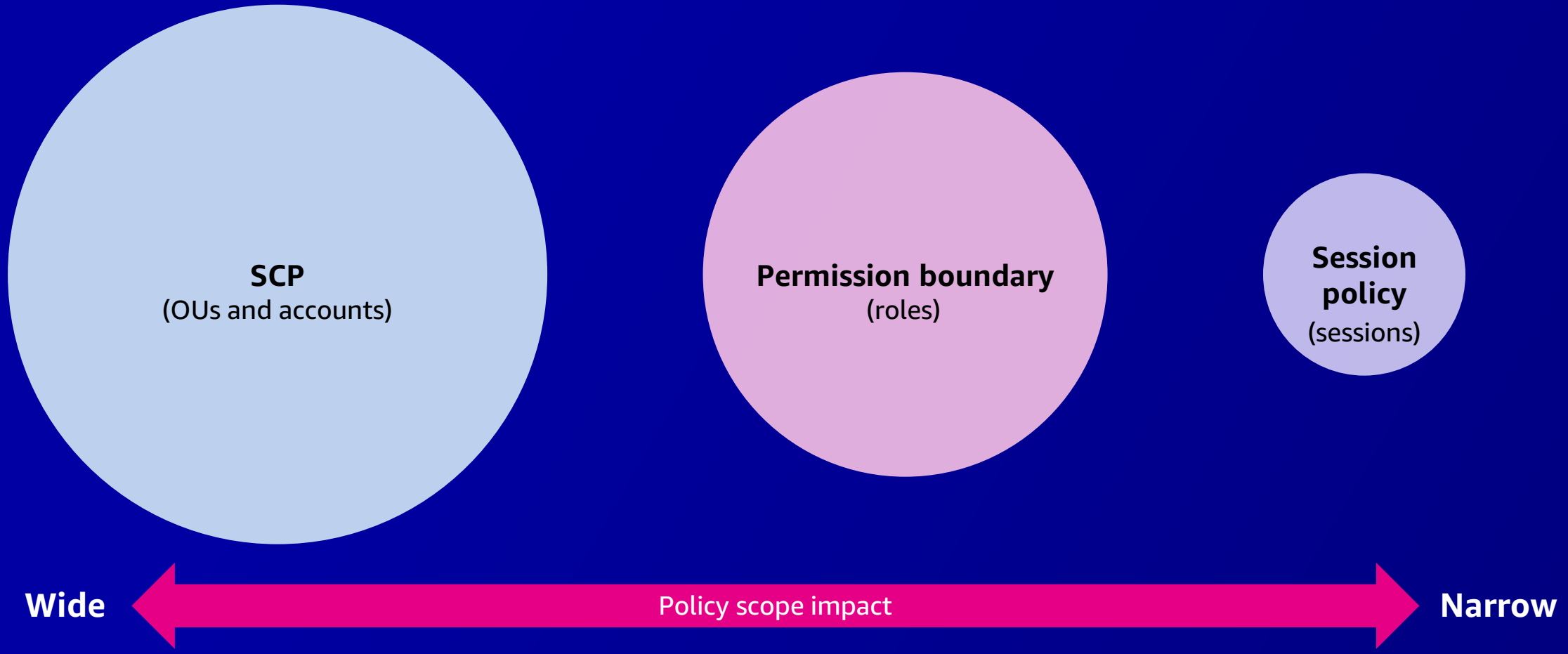


Using SCPs effectively

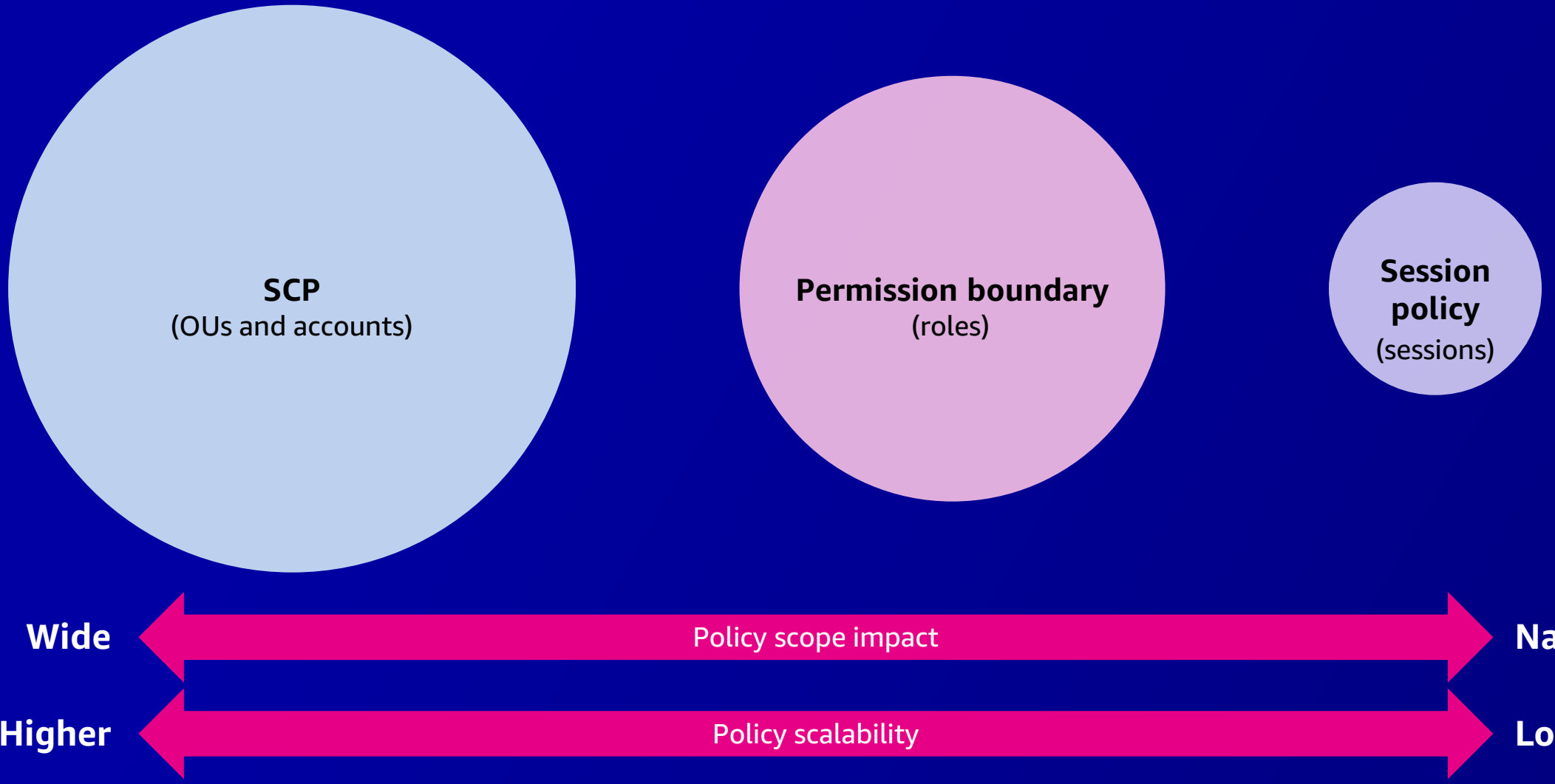
Use the right guardrail for the right controls



Use the right guardrail for the right controls



Use the right guardrail for the right controls

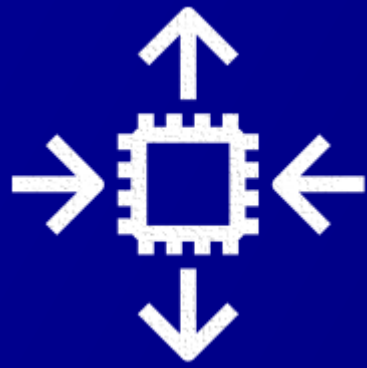


Getting the most out of your SCPs

Get the most out of your SCPs

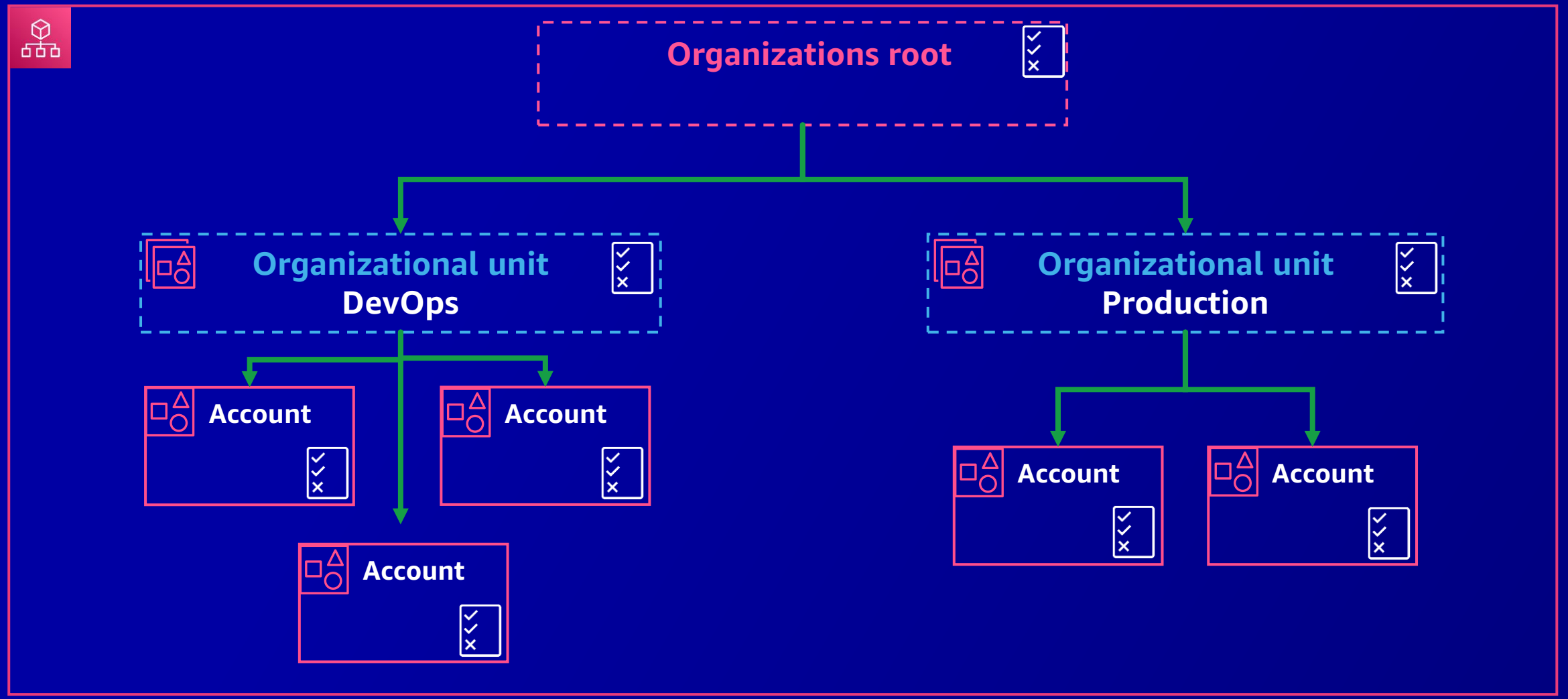
Inheritance

Combine policies

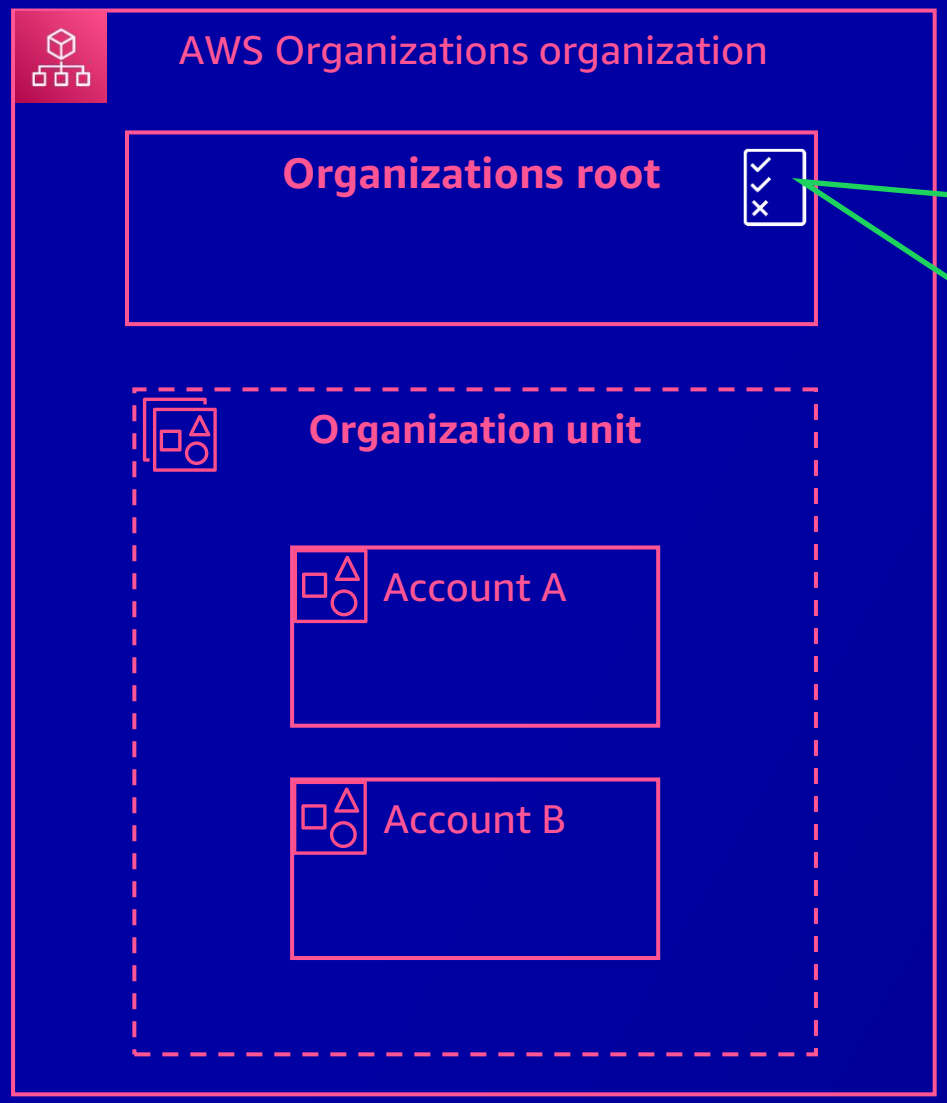


Compact policy

Take advantage of inheritance for Deny

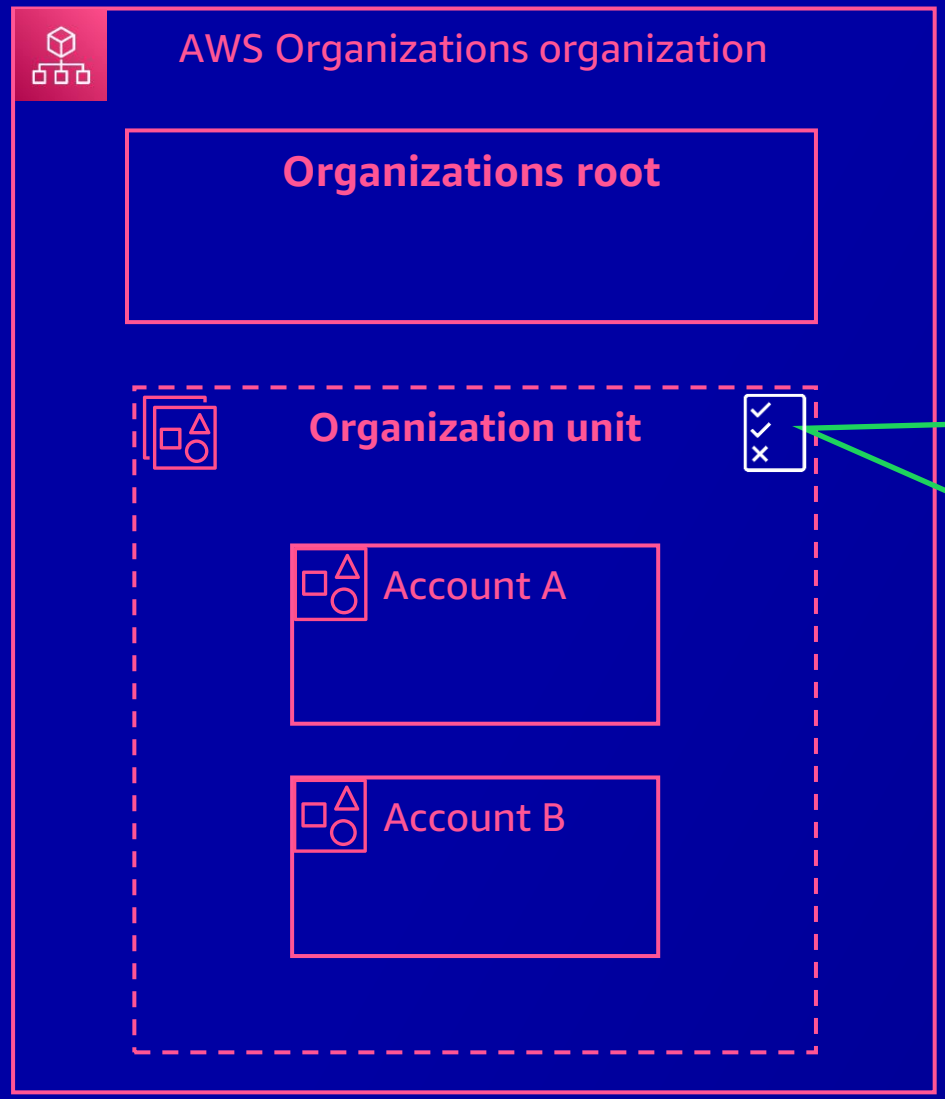


Take advantage of inheritance for Deny



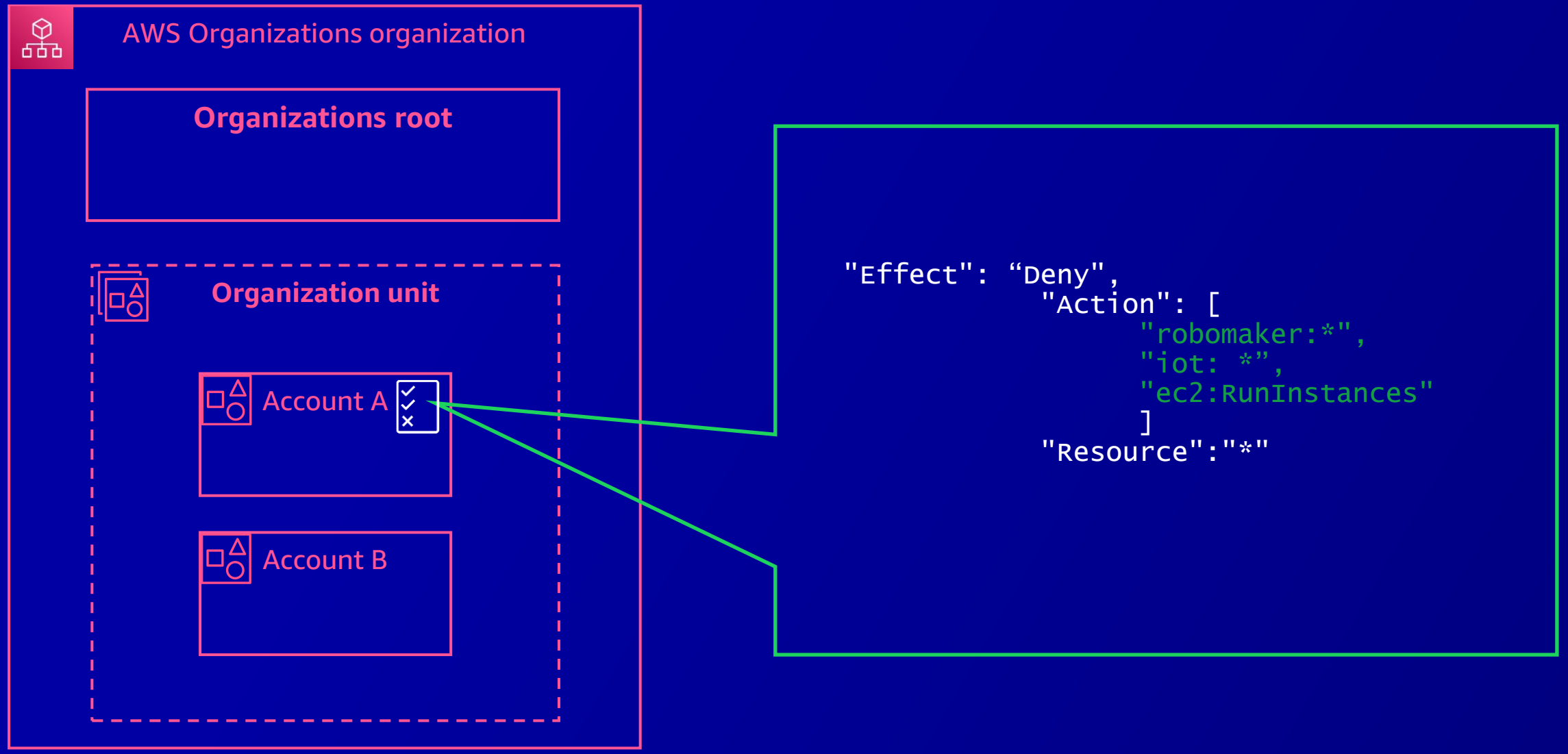

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": “ApprovedServicesOnly",
            "Effect": “Deny",
            "Action": [
                "robomaker:*",
                "iot: *"
]
            "Resource":"*"
        }
    ]
}
```

Take advantage of inheritance for Deny




```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": “ServerlessOnly",
            "Effect": “Deny",
            "Action": "ec2:RunInstances"
            "Resource":"*"
        }
    ]
}
```

Take advantage of inheritance for Deny



Combine policies

MAXIMUM NUMBER YOU CAN ATTACH TO A ROOT, OU, OR ACCOUNT – 5

AWS full access policy [143 bytes]

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

+

Deny bucket deletion and AWS Security Hub disablement [260 bytes]

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": “Deny",
      "Action": "s3:DeleteBucket",
      "Resource": "*"
    },
    {
      "Effect": “Deny",
      "Action":"securityhub:Disable*",
      "Resource": "*"
    }
  ]
}
```

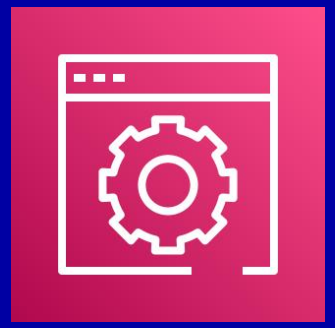
=

Combined policy [274 bytes]

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "*",
      "Resource": "*"
    },
    {
      "Effect": “Deny",
      "Action":[
        "s3:DeleteBucket",
        "securityhub:Disable*“]
      "Resource": "*"
    }
  ]
}
```

Compact your policy

MAXIMUM SIZE OF POLICIES – 5,120 BYTES



Use AWS Management Console

Remove white spa ces

```
{
 "Version":"2012-10-17",
 "Statement":[
  {
   "Sid":“PolicyID",
```

Shorten “Sid”

Our top 5 SCP recommendations

Prevent leaving the organization

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

Why?

- Centrally **secure and audit** your environment across accounts
- Simplify **permission management** and **access control**
- Manage **costs** and **optimize usage**

Only allow usage of approved AWS Regions

```
"Effect": "Deny",
”NotAction": [
  "organizations:*”,
  “iam:*”,
  “budget:*”
  …
],
"Resource": "*",
"Condition" : {
  "StringNotEquals": {
    "aws:RequestedRegion”: [
      ”eu-west-1",
      ”eu-central-1”
    ]}}
```

Policy highlights

- Use `aws:RequestedRegion` condition key to specify allowed Regions with `StringNotEquals` operator
- Use `NotAction` element to create exemptions to the policy, such as global service endpoint

Prevent usage of the root user

```
"Effect": "Deny",
”Action": "*",
"Resource": "*",
"Condition": {
  "StringLike": {
    "aws:PrincipalArn": [
      "arn:aws:iam::*:root"
    ]
  }
}
```

Policy highlights

- Deny all actions[1] by the root user in member accounts
- Use `aws:PrincipalArn` condition key for `root` user with wildcard (`"*"`) for `account-id`
- Create process for temporary exceptions using the `aws:PrincipalAccount` condition key

[1] Tasks that require root user: amzn.to/2VLTB2p

Prevent disabling security services

AWS CloudTrail

```
"cloudtrail:DeleteTrail“
"cloudtrail:PutEventSelectors“
"cloudtrail:StopLogging"
"cloudtrail:UpdateTrail"
```

AWS Config

```
"config:DeleteConfiguration*"
"config:DeleteDeliveryChannel"
"config:DeleteRetention*"
"config:PutConfiguration*“
"config:PutDeliveryChannel“
"config:PutRetention*"
"config:StopConfiguration*"
```

AWS Security Hub

```
"securityhub:Disable*"
"securityhub:Dissociate*"
"securityhub:DeleteInvitations"
"securityhub:DeleteMembers"
```

Protect your sensitive Amazon Simple Storage Service (Amazon S3) buckets

Specific S3 bucket deletion

"s3:DeleteBucket"
"s3:DeleteBucketPolicy"

S3 public access settings

"s3:PutAccountPublicAccessBlock"
"s3:PutBucketPublicAccessBlock "

"s3:PutObjectVersionAcl"
"s3:PutObjectAcl"

Object and version deletion

"s3:DeleteObject"
"s3:DeleteObjectVersion“
"s3:DeleteObjectTagging“
"s3:DeleteObjectVersionTagging"

"s3:PutLifecycleConfiguration"

Morgan Stanley AWS Organizations: Evolution and lessons learned

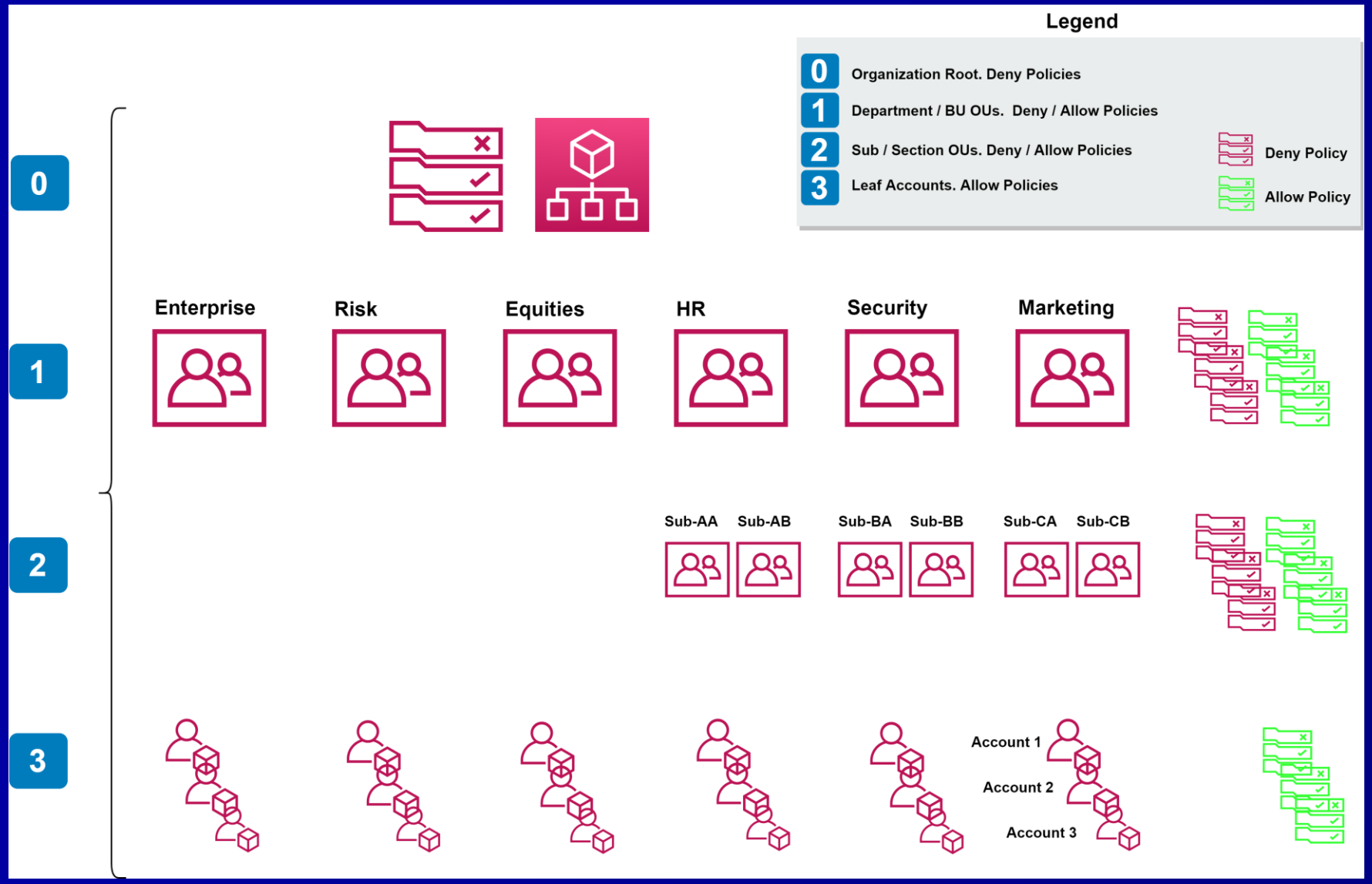
Itay Cohai (he/him)

Executive Director Cloud Platform Engineering
Morgan Stanley

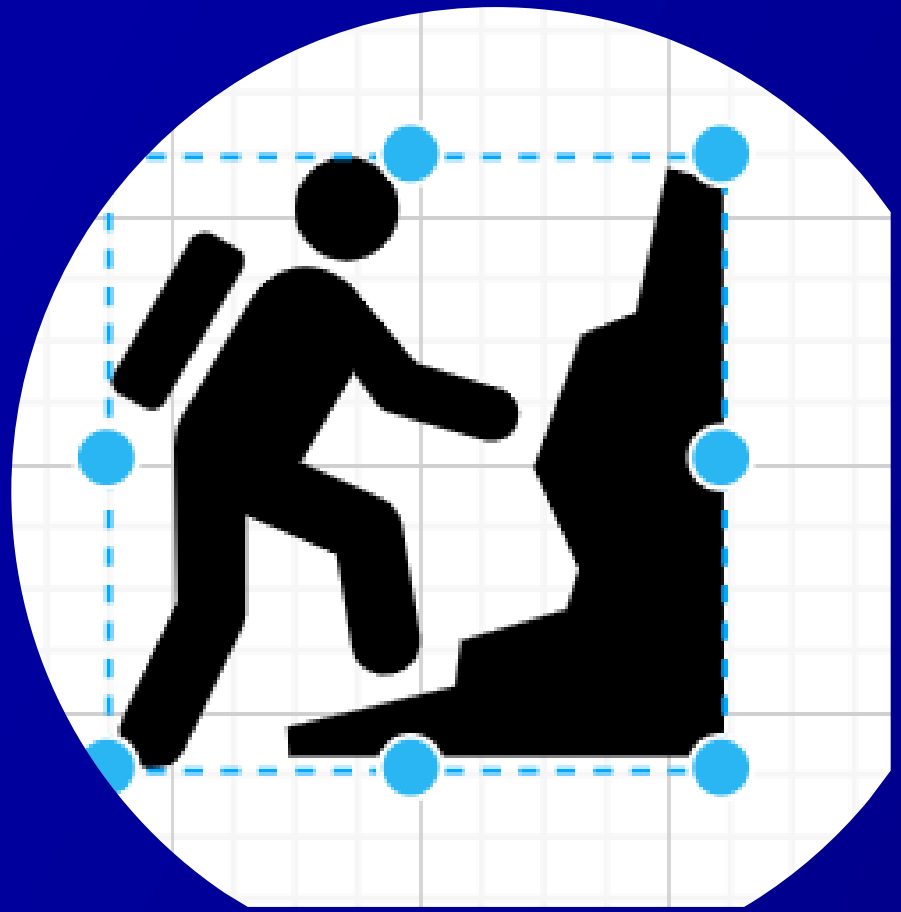
Zulfi Haider (he/him)

Vice President Cloud Platform Engineering
Morgan Stanley

That’s how we started



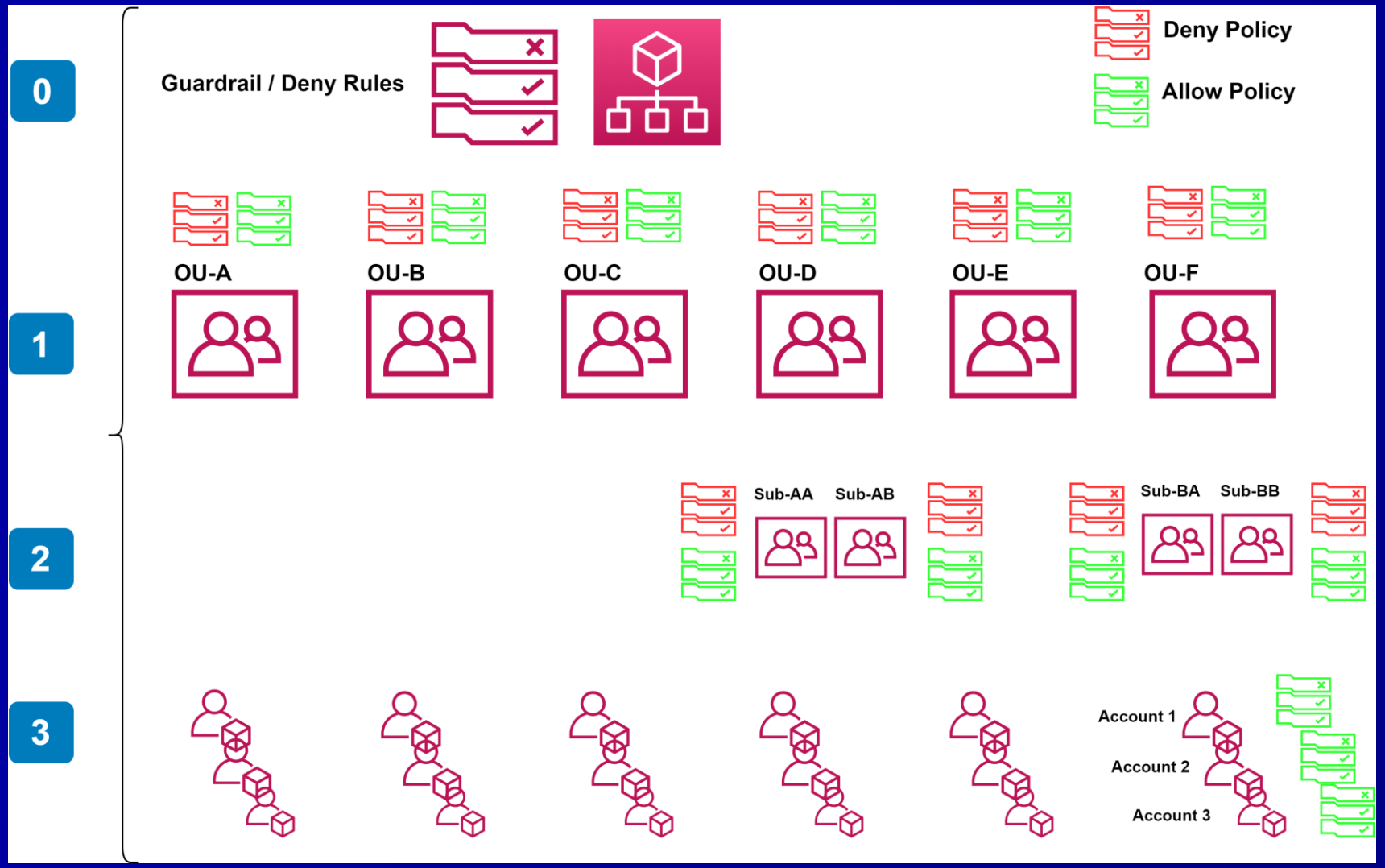
Then comes a re-org



Lessons learned

- **OUs shall be** based on something that **does not** change frequently
- **OUs shall be** based on technical controls and **not HR structure**

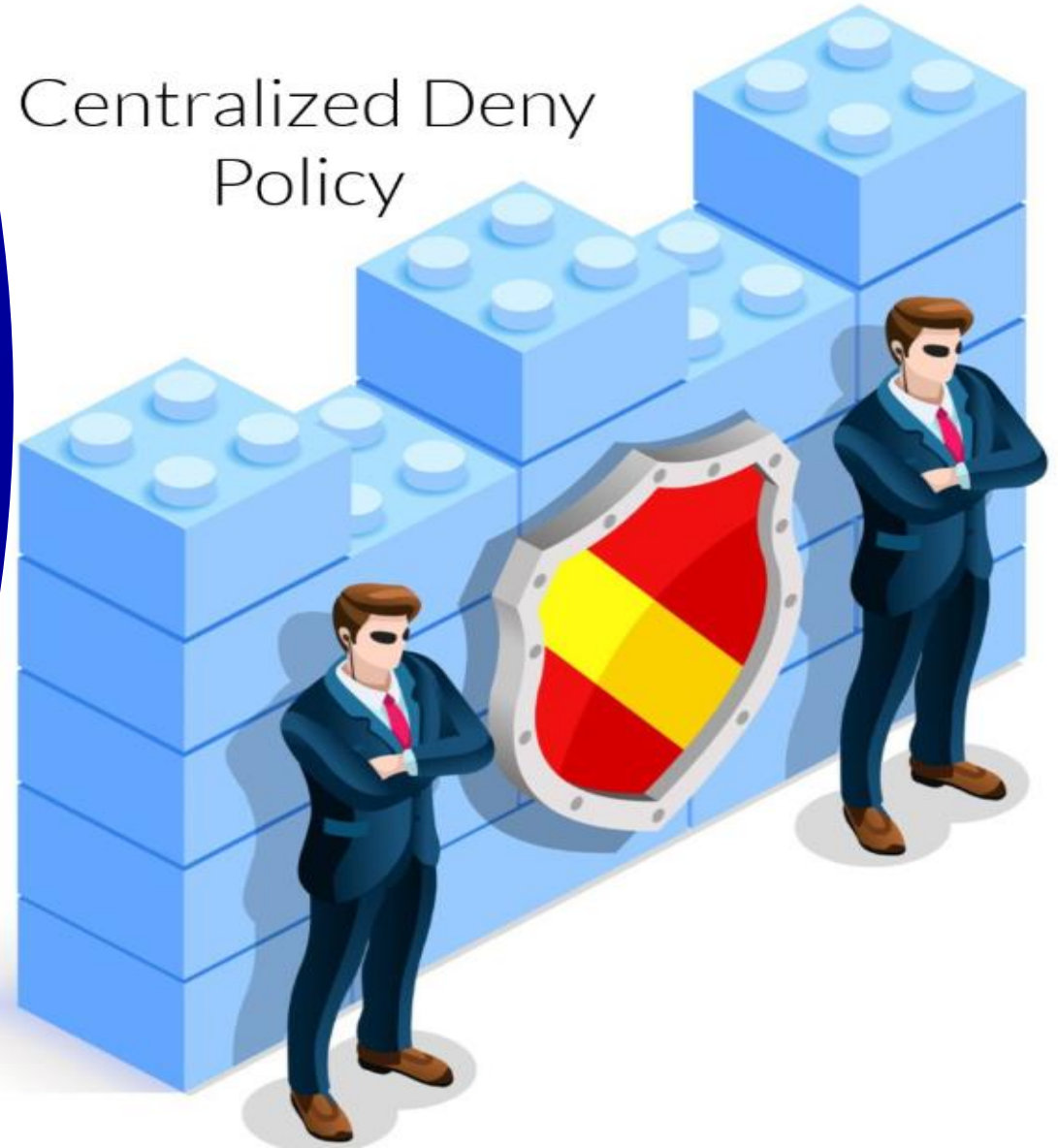
That’s how we deployed deny policies



Then comes troubleshooting

Lessons learned

- Deny policies **shall** be
  - Single layer
  - Immutable
  - Highly guarded



And then we run out of policy space

Lesson learned

- Deny policies shall be
  - Fewer
  - Simpler
  - Coarse grained

Rule of thumb

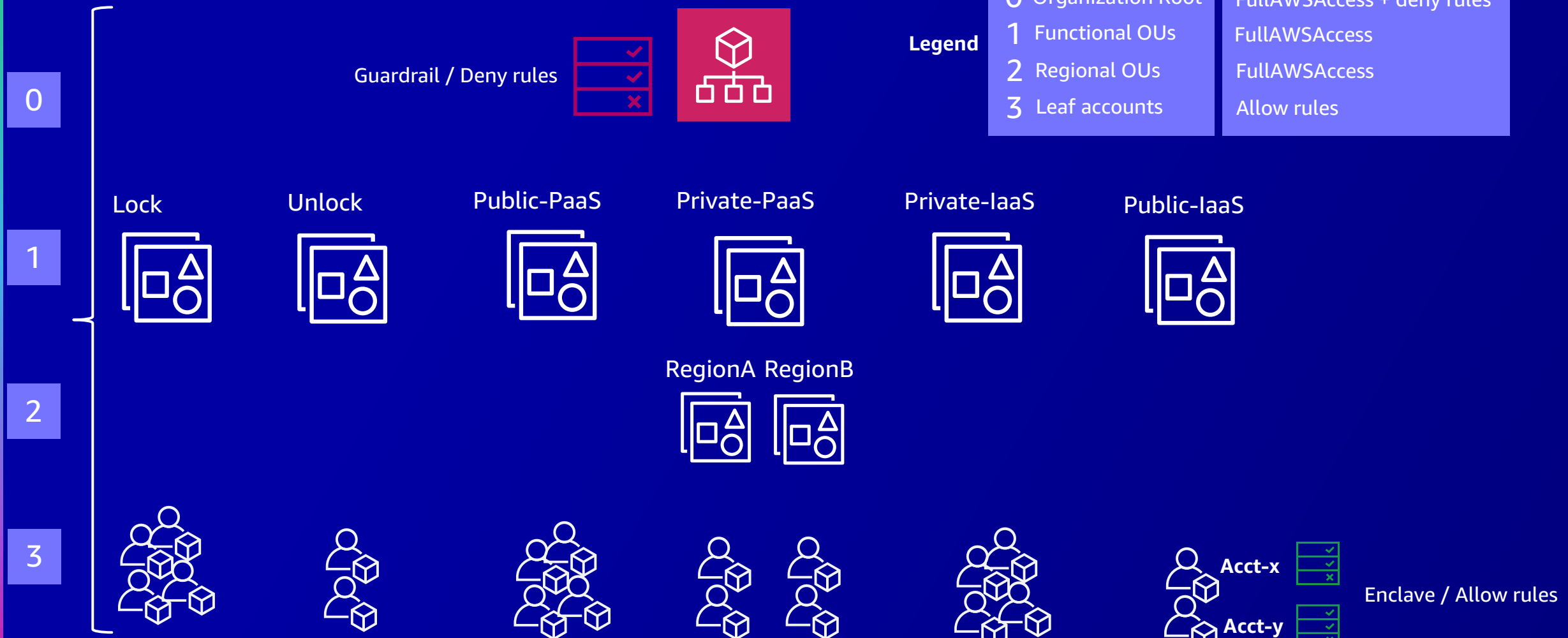
If a deny policy is too fine grained

OR

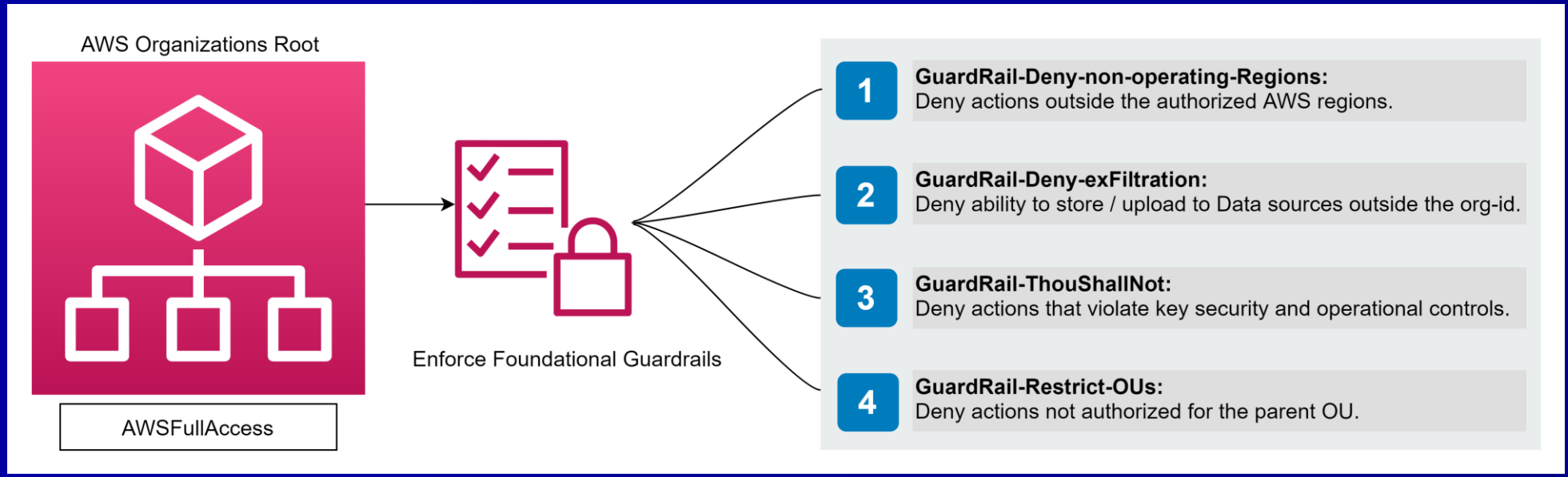
If a deny policy is too long

Then it does not belong in SCP

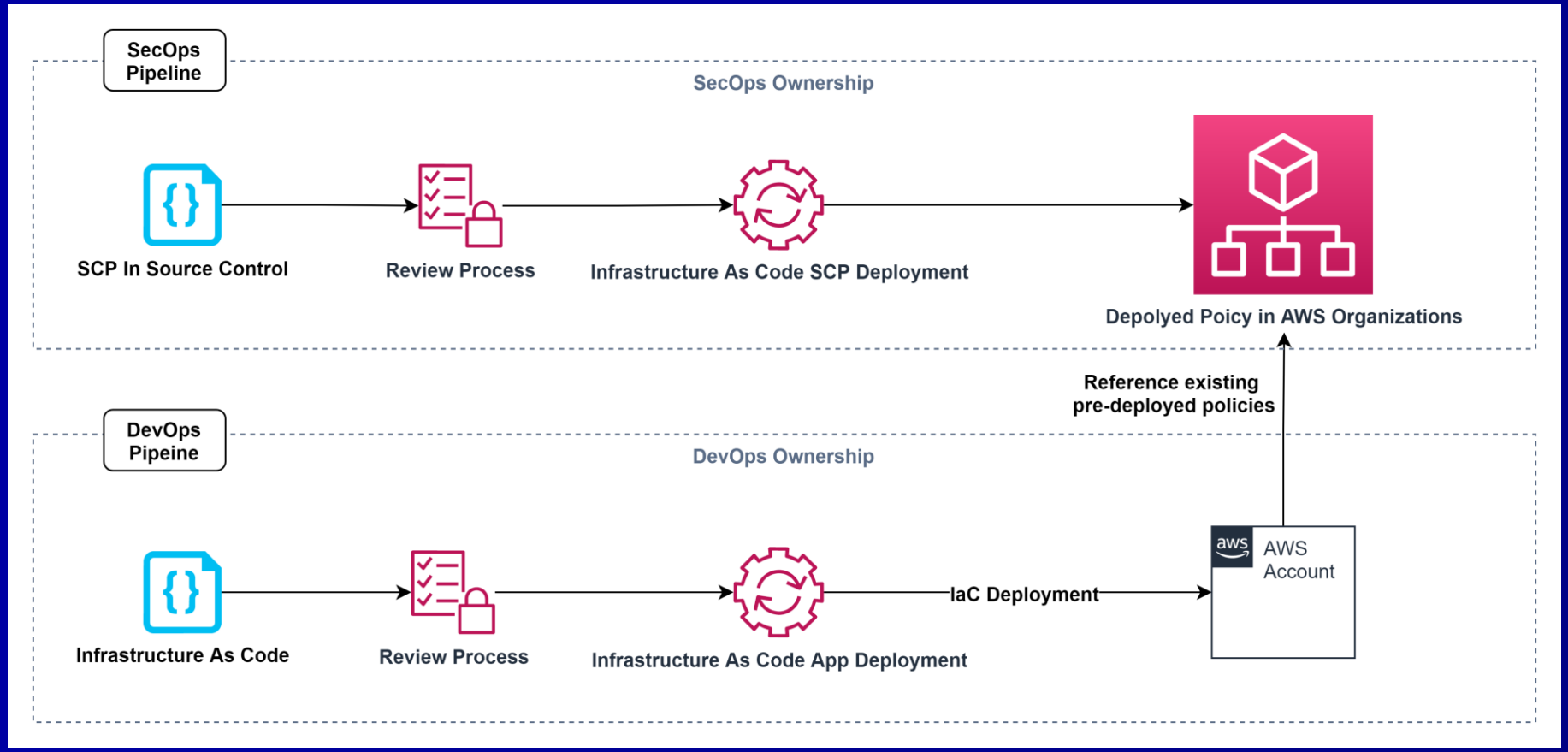
Showcasing our AWS organization



Enforcing our guardrails



Organizing our Organizations deployment



Use cases

UC1: Organization housekeeping

- We manage all phases of account lifecycle through code. `CreateAccount` and `CloseAccount` permissions are only available to automation principals. Human roles cannot create or close accounts.
- We deploy SCP content through code.
- Our organization membership is by birth (account creation) not by invitation.
- A member account cannot leave organization/cannot be removed from organization.
- Permissions like `LeaveOrganization`, `InviteAccountToOrganization`, or `RemoveAccountFromOrganization` are permanently blocked.

Rationale: An account leaving our organization or removed from our organization bypasses SCP guardrails and has security consequences. Accidental account closure has operational consequences. Both options are available through two easy clicks on AWS console, hence needs preventive controls.

UC2: We explicitly allow services per account

- AWS service footprint is growing with new services being added.
- **Never** for `FullAWSAccess` at account Level.
- We take time to review security impact of each service before allowing such service.
- Out of over 200 services, we generally do not expect to use more than 30 services in a given account.

Rationale: Every service has an attack surface. Understand every individual service before allowing such service in our allow rules.

UC2b: We use a single baseline SCP attached to all accounts

```
{
  "Sid":"AllowBaselineServices", "Effect":"Allow",
  "Action":[
        "billing:*",
        "cloudtrail:*",
        "cloudwatch:*",
        "config:*",
        "logs:*",
        "guardduty:*",
        "health:*”,
        ……
        ],
  "Resource":"*”
}
```

Rationale: Allow consistent baseline services required to enable automation, as well as security and operational controls for all accounts

UC3: We use SCP to deny exfiltration

```
{
 "Sid":"DenyS3WritesToUnauthorizedBuckets”
 "Effect":"Deny",
 "Action":[
      "s3:Put*"
      ],
 "Resource":"*",
 "Condition":{
   "StringNotEquals":{
      "aws:ResourceOrgID":"o-XXXXXX"
      }
   }
}
```

Rationale: Block path for uploading data to personal Amazon S3 buckets from within VPCs

UC4: Our guardrail policy attachments are immutable

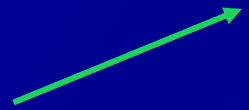
- What is immutable – Build once, attach forever
- Guardrail SCPs are **permanently** attached to the org root
- **Deny** rules prevent actions for the whole org – No exception
- **Restrict** rules restrict actions for specific OUs
- How we do that – By using condition key `aws:PrincipalOrgPaths`
- So to exclude a restriction, we do not need to detach SCP or modify SCP, we move account to an OU that has exclusion to such restriction. And we monitor OU memberships based on rules.

Rationale: Affecting changes to SCP controls through OU membership allows us to be flexible while keeping our policy attachments immutable

UC5: We Disable all actions for Lock OU

```
{
  "Sid": "Lock",
  "Effect": "Deny",
  "Action": "*",
  "Resource": "*" ,
  "Condition": {
        "ForAnyValue:StringEquals": {
           "aws:PrincipalOrgPaths": [
                          "o-xxxxx/r-xx/",
                          "o-xxxxxx/r-xxx/ou-xxx-xxxxxxx/"
                         ]}}
}
```

Deny actions for lock OU and root OU members



Rationale: Disable all control plane actions for an account

Morgan Stanley SCP best practices

1. **Deny Policies are our guardrails**. Attach guardrails to org root level
2. **Allow Policies are our enclaves**. Attach enclaves to leaf account level
3. Security Monitoring to assure **leaf accounts do not attach FullAWSAccess**, and operational monitoring to assure **OUs always attach FullAWSAccess**
4. Separation of duty between managing organization versus the rest of the account
    - Key organization artifacts are managed **by DevSecOps pipeline**, rest of AWS managed by **DevOps pipeline**
5. Separation of entitlement between org management account and leaf accounts
    - Access to org management account follows separate entitlement and approval
6. Every new service allowed in enclave follows a security review
7. OUs are our best friends for managing AWS account group restrictions
    - Guardrail policy immutably attached to org root and denies actions based on OU membership (aws:PrincipalOrgPaths)
8. Restrict Regions based on OUs
9. Put the **unneeded accounts in LOCK OU**
10. Only **allow root login in a highly monitored OU**

Resources

- Blog : Get more out of service control policies in a multi-account environment
- Example service control policies to get started
- Getting started with AWS Organizations

Thank you!

Swara Gandhi
ganswara@amazon.com

Itay Cohai
Itay.cohai@morganstanley.com

Zulfi Haider
Zulfiqar.haider@morganstanley.com