Pentester's Approach To AWS IAM

By: Divyanshu Shukla
s3curitydojo@gmail.com

Disclaimer

- The views expressed in this presentation and its content, as well as any accompanying resources, are solely the speaker's own and do not necessarily reflect the opinions or endorsements of the trainer's employer.
- Securitydojo is the personal website of the author and does not represent any business entity.

Hello!

I am Divyanshu | @justmorpheus

- Senior Cloud Security Engineer with 7 years of experience.
- Acknowledged by Airbnb, Google, Microsoft, Apple, Samsung (CVE-2019-8727), AWS, Amazon, Mozilla, etc with various CVEs.
- **Speaker & Trainer:** Blackhat Europe, C0c0n, Nullcon, Bsides/CSA Bangalore, Null Bangalore, Nirmata Meetup, IIT Dharwad
- **Authored:** GCP Inspector, BurpoMation, VeryVulnerableServerless
- Defcon CloudVillage (20/21/22) & AWS Community Builder

Agenda

- What is IAM?
- IAM Concepts
- Policy Types
- Boundary Types
- Policy Evaluation Logic
- Attacks – Least Privilege, PassRole & Assume Role

Talk Prerequisites

- Familiarity with the AWS.
- AWS account with administrative privileges, including billing enabled.
- Registered account on Killercoda.com.

1

IAM Introduction

Basics of AWS Identity & Access Management

Identity & Access Management

- Enables control on who can do what in your AWS Account.
- IAM controls access by defining **who** (identity) has **what** access (role) for **which** resource in the AWS Account.
- IAM also dictates access privileges to your entire AWS instance.

Who, Where & What ?

- Users and Groups $\longrightarrow$ Who
- Roles $\longrightarrow$ Where
- Policies $\longrightarrow$ What

IAM Users

- Refers to a user to your AWS instance. Access can be provided programmatically or through the console OR both.
- An IAM user is **a resource in IAM that has associated credentials and permissions**.
- Access methods must be explicitly assigned.

Do not use root

Instead create an IAM user with “Full Administrative Access” & enable MFA for root user.

IAM Groups

- Users can be organized based on Groups (of Users)
- Example: For developers, Dev (Group) can be created.
- Nested Groups is NOT possible with AWS IAM.

IAM Roles

- Allows applications to access AWS resources without manually providing/hardcoding AWS credentials.
- Steps for the role:
  - Create a role
  - Attach policy (permissions) to a role
  - Attach role to resource & instance.

IAM Policy

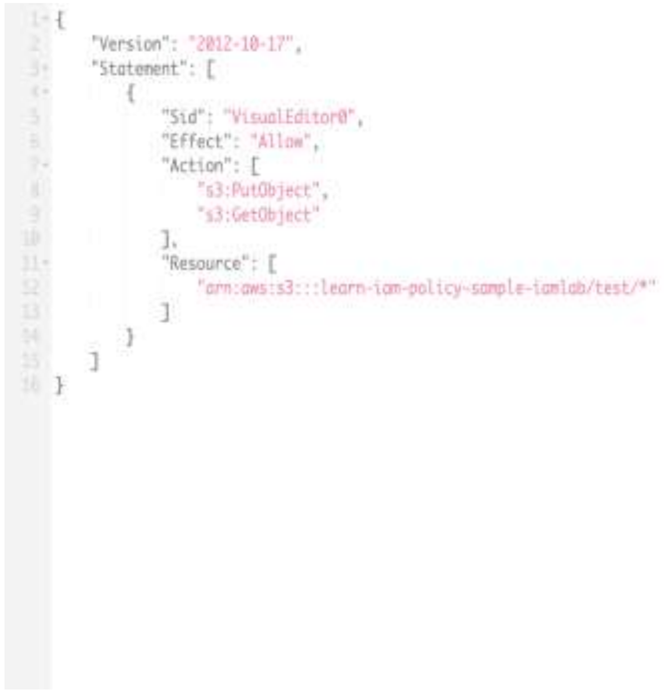
- JSON document that defines permissions.
- No effect until it is attached to the resources.
- It is a list of statements in the json.
- Several canned policies are provided by AWS
- Users, Groups and Roles can be linked with multiple policies.

IAM Policy Terminology

- Statements is definition of the permissions.
- Resources is the resources based on ARN.
- Actions is the API Mapping of actions possible against the resources.
- Effect is the Allow/Deny to actions for resources.
- Policies also have Negative variants like NotResource & NotAction.

IAM Policy Explanation

- Policy is a JSON document.
- Version helps to identify the structure
- Sid is a label to identify the statements
- Effect is **Allow** or **Deny**.
- Action is list of permissions.
- Resource is List of resources



```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:GetObject"
            ],
            "Resource": [
                "arn:aws:s3:::learn-iam-policy-sample-iamlab/test/*"
            ]
        }
    ]
}
```

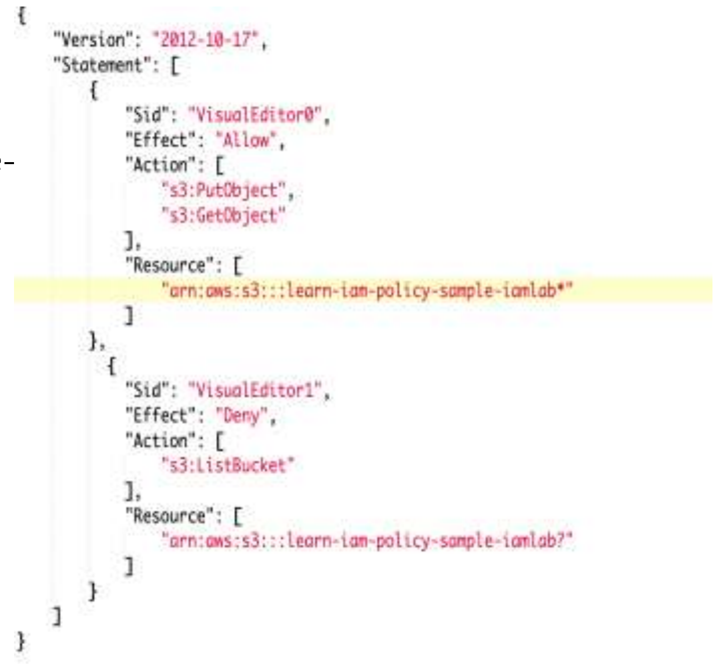
IAM Policy Resource Element

- ARN uniquely identify AWS resources.
- Amazon Resource Name (ARN):

  arn:partition:service:region:account-id:resource-
- Wildcards possible,

  – "Resource": "arn:aws:s3:::learn-iam-policy-sample-iamlab*"

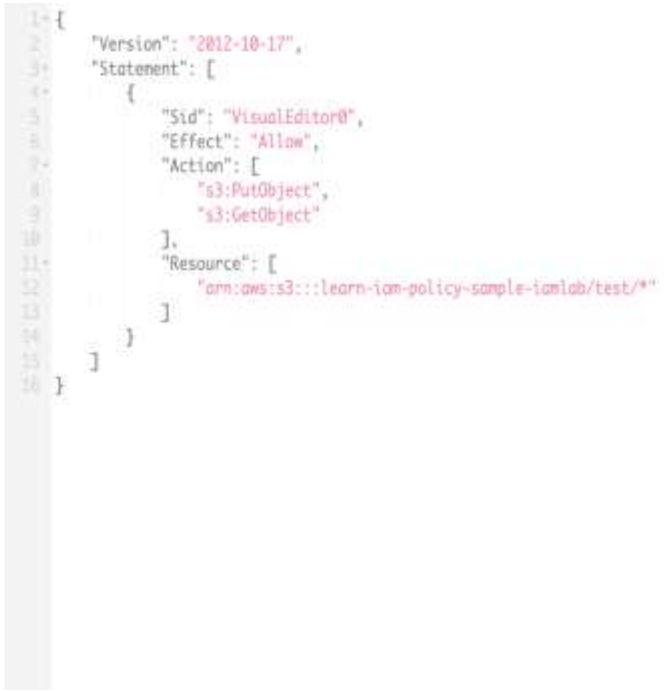
  – "Resource": "arn:aws:s3:::learn-iam-policy-sample-iamlab?"



```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:GetObject"
            ],
            "Resource": [
                "arn:aws:s3:::learn-iam-policy-sample-iamlab*"
            ]
        },
         {
            "Sid": "VisualEditor1",
            "Effect": "Deny",
            "Action": [
                "s3:ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::learn-iam-policy-sample-iamlab?"
            ]
        }
    ]
}
```

IAM Policy Example

- Actions Put object and Get object are allowed on the resources i.e. on the S3 bucket (learn-iam-policy-sample-iamlab).



```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:GetObject"
            ],
            "Resource": [
                "arn:aws:s3:::learn-iam-policy-sample-iamlab/test/*"
            ]
        }
    ]
}
```

Multiple Statements

Multiple Statements per policy is allowed in IAM Policies.

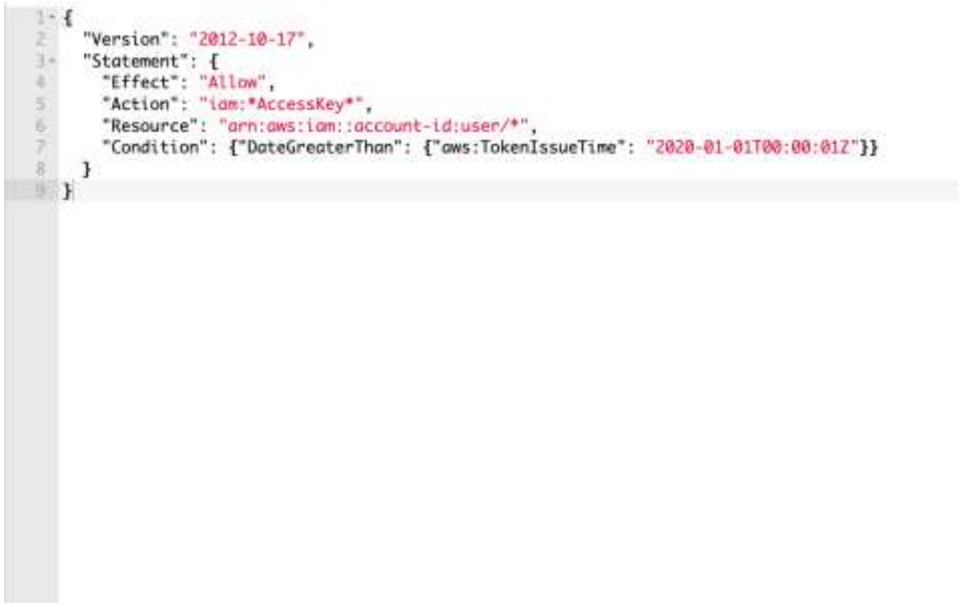
IAM Policy Statement

- Statement has Effect must be set to either Allow or else Deny.
- Action must be specific actions that will be allowed or denied.
- Resource is referred to by the ARN.
- Condition is additional conditions when the policy is in effect.
- Principal is the IAM user used to specify an IAM role

- Policy Statements also have NOT Policy operators.
- NotAction is the action which applies to everything except the action given.
- NotResource applies to everything **except** provided resource.
- NotPrincipal applies to every principal except one given.

IAM Conditional Operators

- String Operators are equals, like, not like, etc
- Numeric are equals, Not Equals, less than, greater than.
- DateTime are equals, NotEquals, GreaterThan, LessThan Boolean.
- Binary are the **key-value** pairs in the base64 encoded format.
- IPAddress is based on IPAddress OR NotIpAddress conditions.

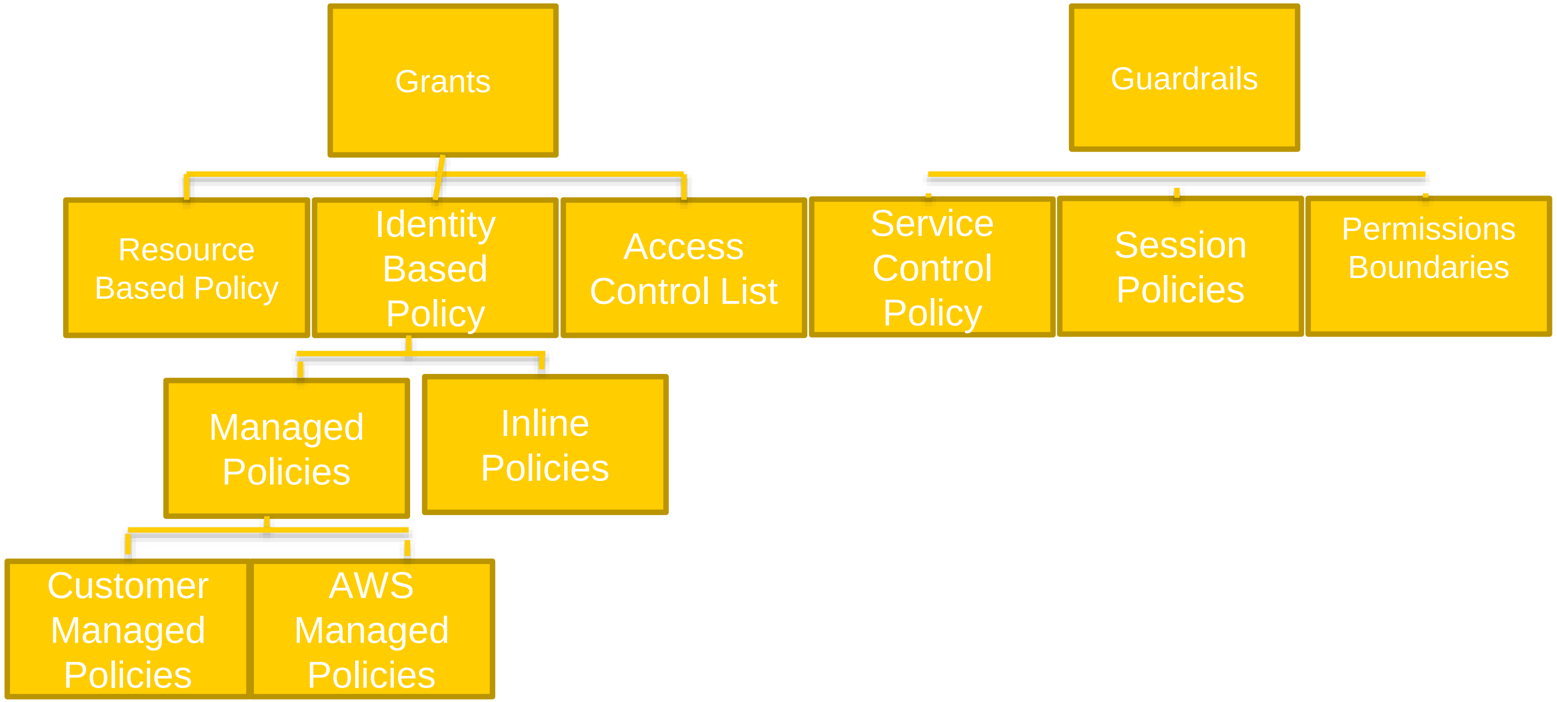


```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "iam:*AccessKey*",
    "Resource": "arn:aws:iam::account-id:user/*",
    "Condition": {"DateGreaterThan": {"aws:TokenIssueTime": "2020-01-01T00:00:01Z"}}
  }
}
```

IAM Policy Demo

Youtube Demo Link

Types of IAM Policies



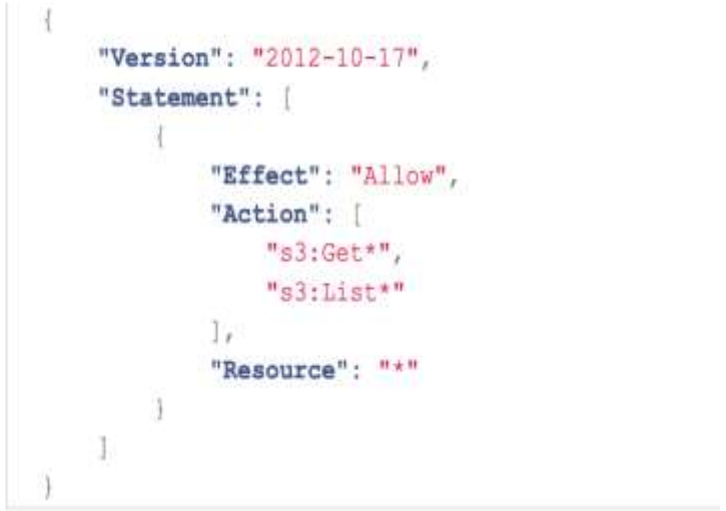
Ref: https://osamaoracle.com/2021/08/15/aws-iam-policy-basics/

Guardrails vs. Grants

Guardrails are the policies used to restrict permissions & grants are used to grant access.

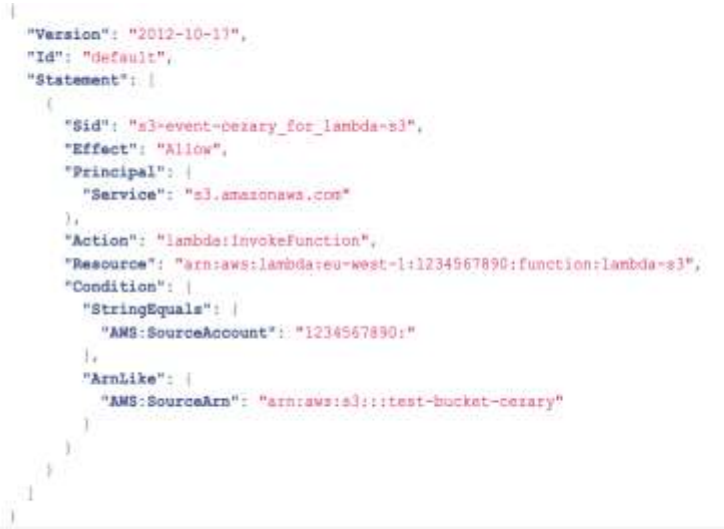
Resource Based Policy

AWS Resource Based Policy



```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:Get*",
                "s3:List*"
            ],
            "Resource": "*"
        }
    ]
}
```

Identity-based policies grant permissions to an identity. An identity-based policy dictates whether an identity to which this policy is attached is allowed to make API calls to specific resource or not.



```
{
  "Version": "2012-10-17",
  "Id": "default",
  "Statement": [
    {
      "Sid": "s3-event-cezary_for_lambda-s3",
      "Effect": "Allow",
      "Principal": {
        "Service": "s3.amazonaws.com"
      },
      "Action": "lambda:InvokeFunction",
      "Resource": "arn:aws:lambda:eu-west-1:1234567890:function:lambda-s3",
      "Condition": {
        "StringEquals": {
          "AWS:SourceAccount": "1234567890:"
        },
        "ArnLike": {
          "AWS:SourceArn": "arn:aws:s3:::test-bucket-cezary"
        }
      }
    }
  ]
}
```

Resource-based policies grant permissions to the principal that is specified in the policy. For example, the policy below specifies that S3 events on the bucket arn:aws:s3:::test-bucket-cezary can be handled by the Lambda (lambda-s3) in account id 1234567890 in eu-west-1 region.

Identity Based Policy

Managed Policy

AWS managed policies

- Standalone policy created & administered by AWS.
- arn:aws:iam::aws:policy/IAMReadOnlyAccess is an AWS managed policy.
- Read only policies.

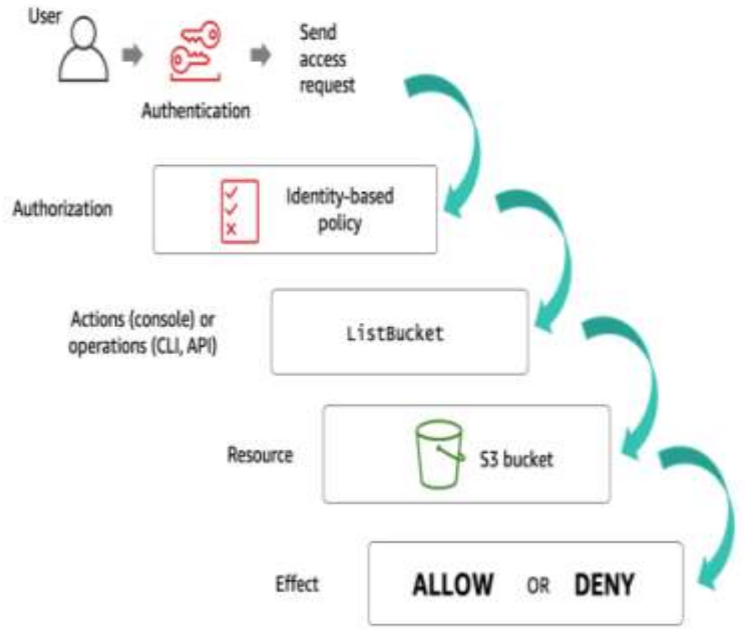
Customer managed policies

- Standalone policies that you administer in your own AWS account.
- arn:aws:iam::<AWSAccount ID>:policy/<Policy_Name>
- Read, Write & Modify with maximum 5 versions.

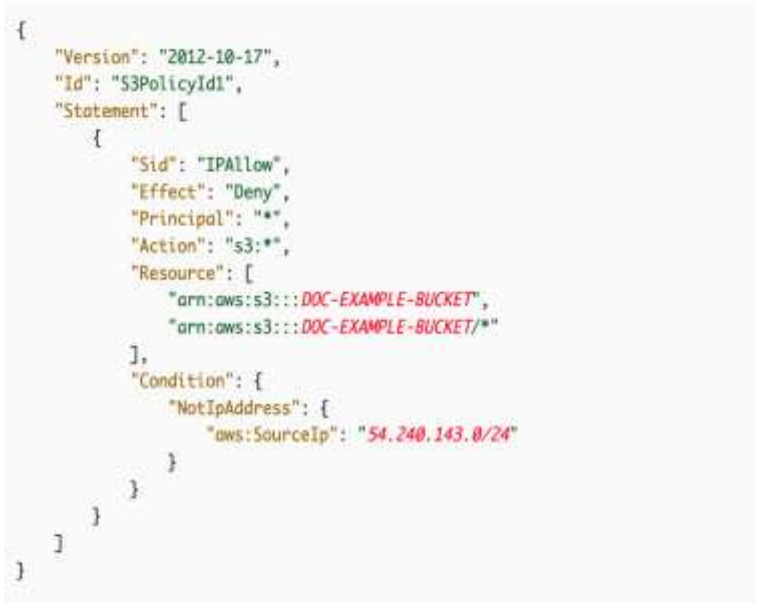
Inline Policy

Inline policies

- An inline policy is a policy that's embedded in an IAM identity (a user, group, or role).

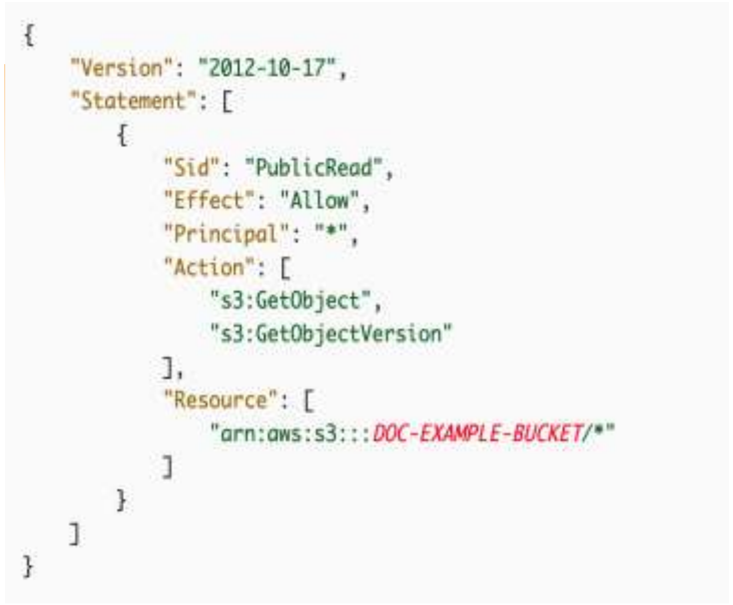


AWS Policy Deny vs Allow



```
{
    "Version": "2012-10-17",
    "Id": "S3PolicyId1",
    "Statement": [
        {
            "Sid": "IPAllow",
            "Effect": "Deny",
            "Principal": "*",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET",
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
            ],
            "Condition": {
                "NotIpAddress": {
                    "aws:SourceIp": "54.240.143.0/24"
                }
            }
        }
    ]
}
```

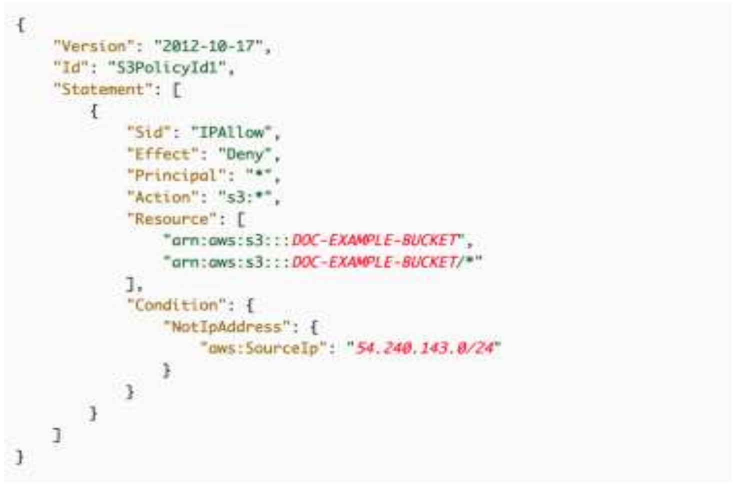
Denies permissions to any user to perform any Amazon S3 operations on objects in the specified S3 bucket unless the request originates from the range of IP addresses specified in the condition.



```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "PublicRead",
            "Effect": "Allow",
            "Principal": "*",
            "Action": [
                "s3:GetObject",
                "s3:GetObjectVersion"
            ],
            "Resource": [
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
            ]
        }
    ]
}
```

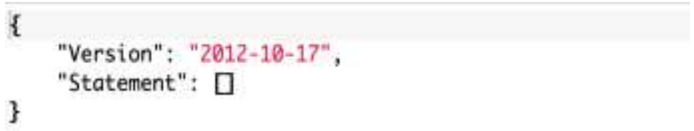
Policy allows the s3:GetObject permission to any public anonymous users.

AWS Policy Implicit Deny vs Explicit Deny



```
{
    "Version": "2012-10-17",
    "Id": "S3PolicyId1",
    "Statement": [
        {
            "Sid": "IPAllow",
            "Effect": "Deny",
            "Principal": "*",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET",
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
            ],
            "Condition": {
                "NotIpAddress": {
                    "aws:SourceIp": "54.240.143.0/24"
                }
            }
        }
    ]
}
```

Explicit Deny permissions to any user to perform any Amazon S3 operations on objects in the specified S3 bucket unless the request originates from the range of IP addresses specified in the condition.



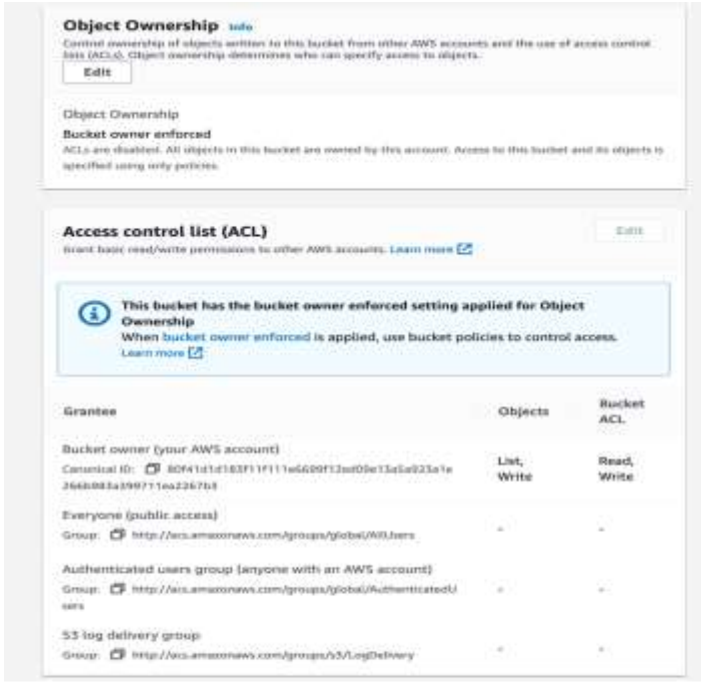
```
{
    "Version": "2012-10-17",
    "Statement": []
}
```

An implicit denial occurs when there is no applicable Deny statement but also, no applicable Allow statement.

Access Control Lists

AWS Access Control Lists

- ACLs are supported by Amazon S3 buckets and objects.
- They are similar to resource-based policies.
- Contains Grantee & Permissions.



Object Ownership Info

Control ownership of objects written to this bucket from other AWS accounts and the use of access control lists (ACLs). Object ownership determines who can specify access to objects.

Edit

Object Ownership

Bucket owner enforced

ACLs are disabled. All objects in this bucket are owned by this account. Access to this bucket and its objects is specified using only policies.

Access control list (ACL)

Edit

Grant basic read/write permissions to other AWS accounts. Learn more

This bucket has the bucket owner enforced setting applied for Object Ownership
When bucket owner enforced is applied, use bucket policies to control access.
Learn more

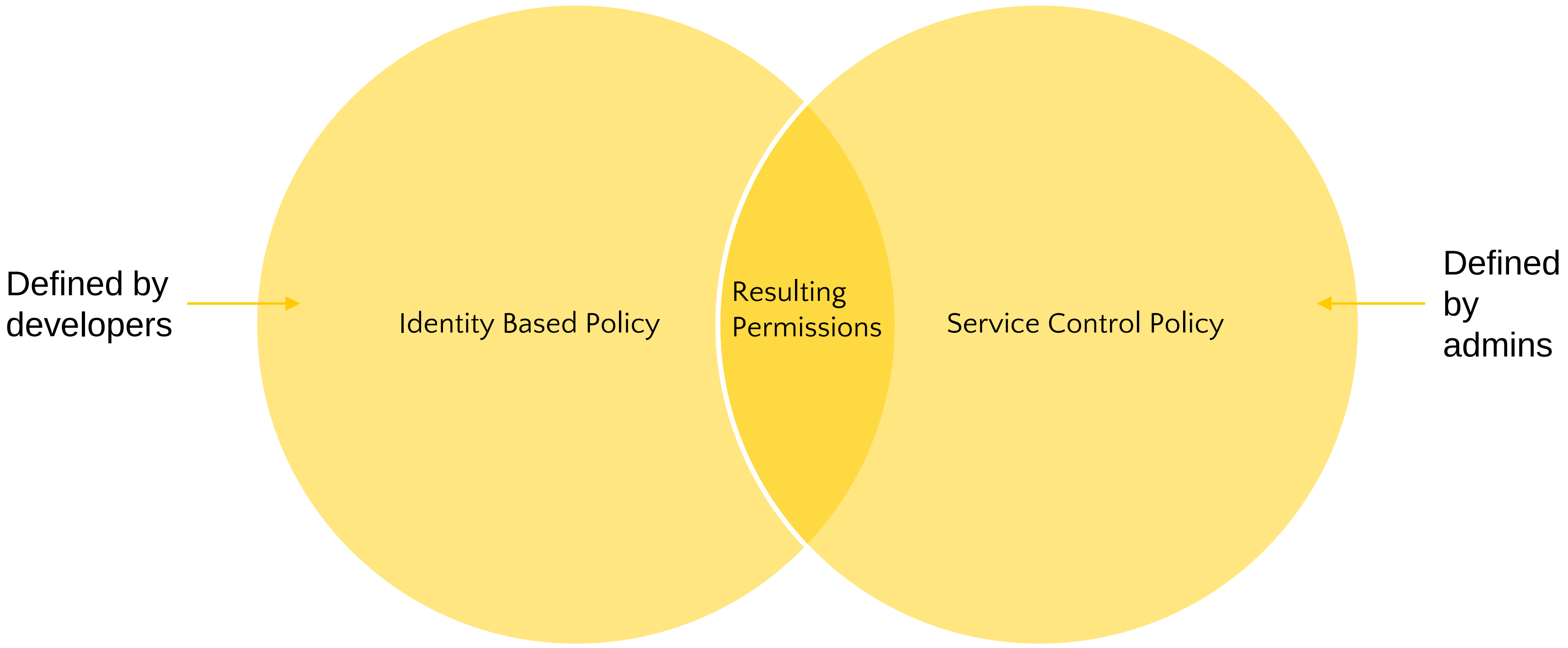
Grantee	Objects	Bucket ACL
Bucket owner (your AWS account) Canonical ID: [illegible]	List, Write	Read, Write
Everyone (public access) Group: http://acs.amazonaws.com/groups/global/AllUsers	-	-
Authenticated users group (anyone with an AWS account) Group: http://acs.amazonaws.com/groups/global/AuthenticatedUsers	-	-
S3 log delivery group Group: http://acs.amazonaws.com/groups/s3/LogDelivery	-	-

Service Control Policies

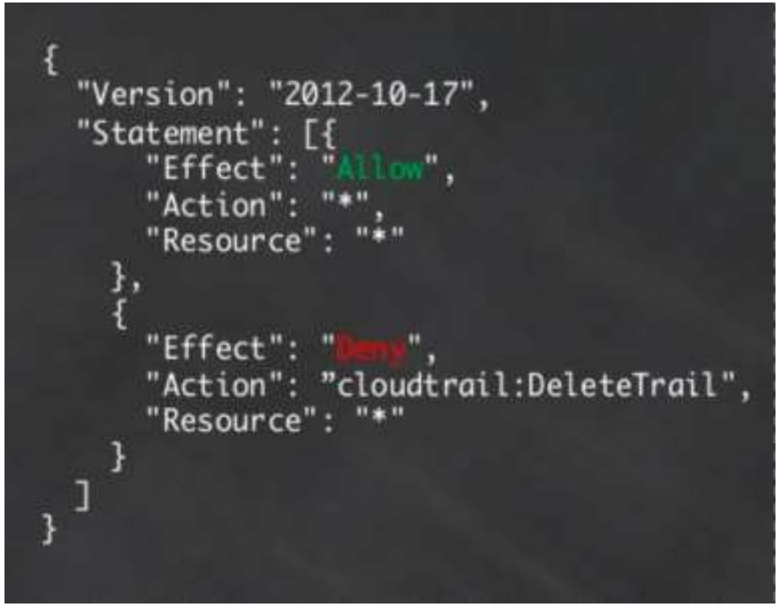
AWS Service Control Policies (SCPs)

- Enables control for the AWS APIs which are accessible.
- Whitelisting, defines the list of APIs that are allowed.
- Blacklisting, defines the list of APIs that are blocked.
- Cannot be overridden by local administrators.
- Resultant permission on IAM user/role is the intersection between the SCP and the assigned IAM permissions.

SCP Permissions- Venn Diagram

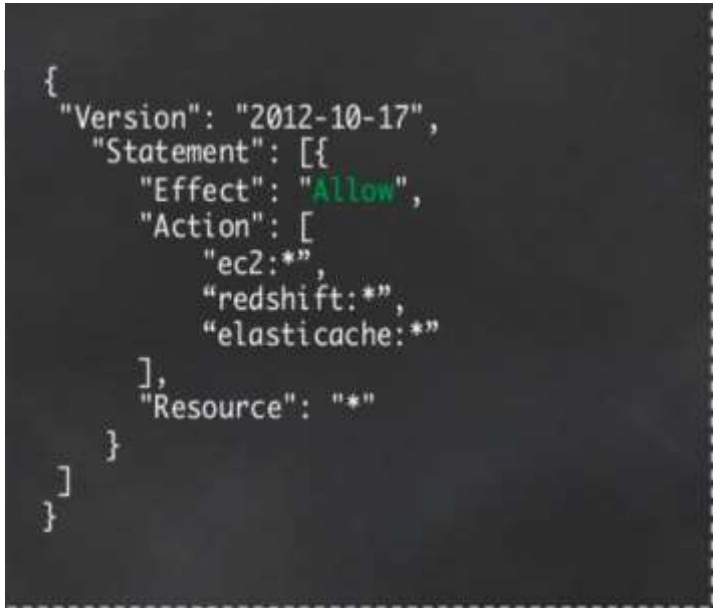


SCP Blacklisting vs Whitelisting



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

Blacklisting Example



```
{
 "Version": "2012-10-17",
   "Statement": [{
     "Effect": "Allow",
     "Action": [
         "ec2:*",
         "redshift:*",
         "elasticache:*"
     ],
     "Resource": "*"
   }
 ]
}
```

Whitelisting Example

Ref: https://www.slideshare.net/AmazonWebServices/aws-iam-introduction

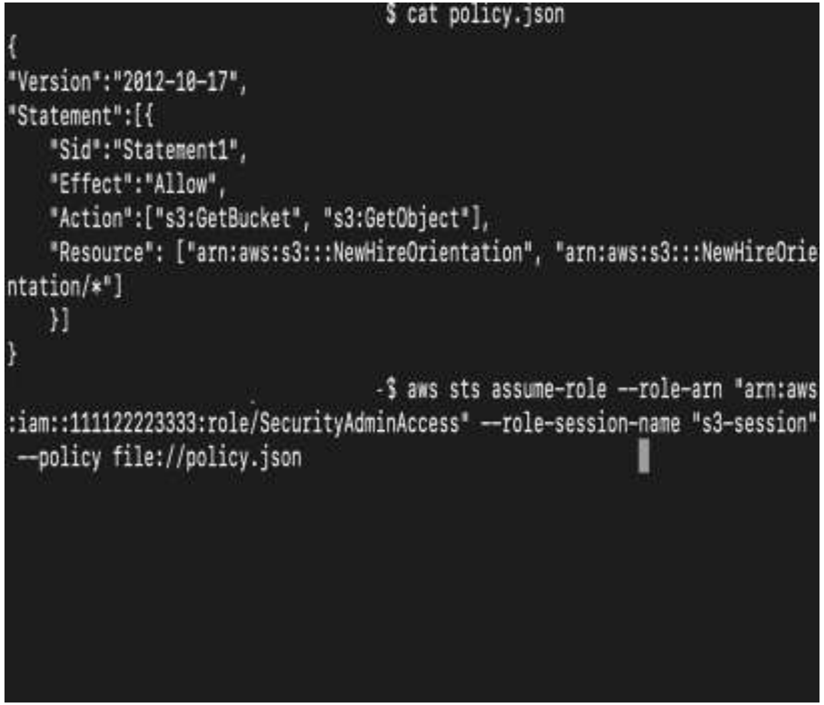
AWS Organizations

It is a service for grouping and centrally managing AWS accounts. If you enable all features in an organization, then you can apply SCPs to any or all of your accounts.

AWS Session Policy

AWS Session Policy

- An inline permissions policy that users pass in the session when they assume the role.
- Effective permissions of the session are the intersection of the role's identity-based policies and the session policy.

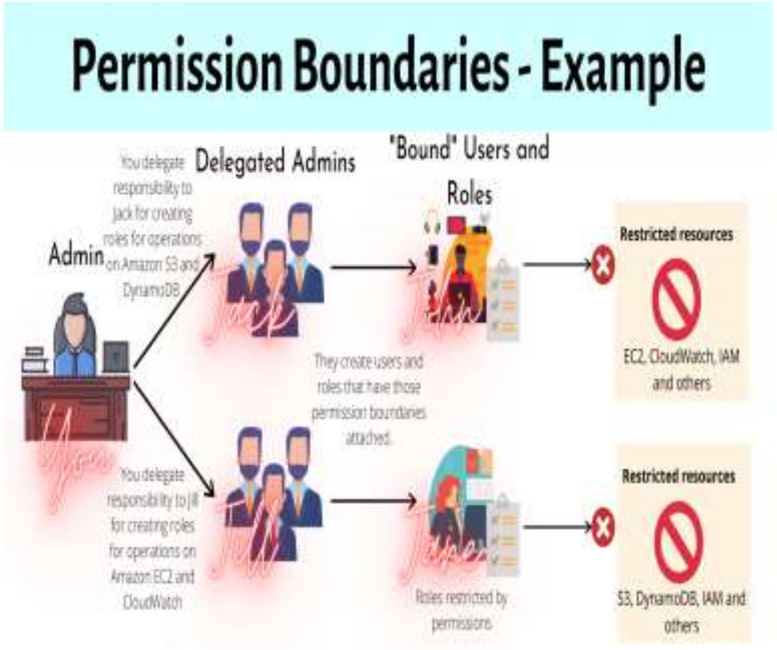


```
                                  $ cat policy.json
{
"Version":"2012-10-17",
"Statement":[{
    "Sid":"Statement1",
    "Effect":"Allow",
    "Action":["s3:GetBucket", "s3:GetObject"],
    "Resource": ["arn:aws:s3:::NewHireOrientation", "arn:aws:s3:::NewHireOrie
ntation/*"]
    }]
}
                                 -$ aws sts assume-role --role-arn "arn:aws
:iam::111122223333:role/SecurityAdminAccess" --role-session-name "s3-session"
 --policy file://policy.json
```

AWS Permissions Boundary

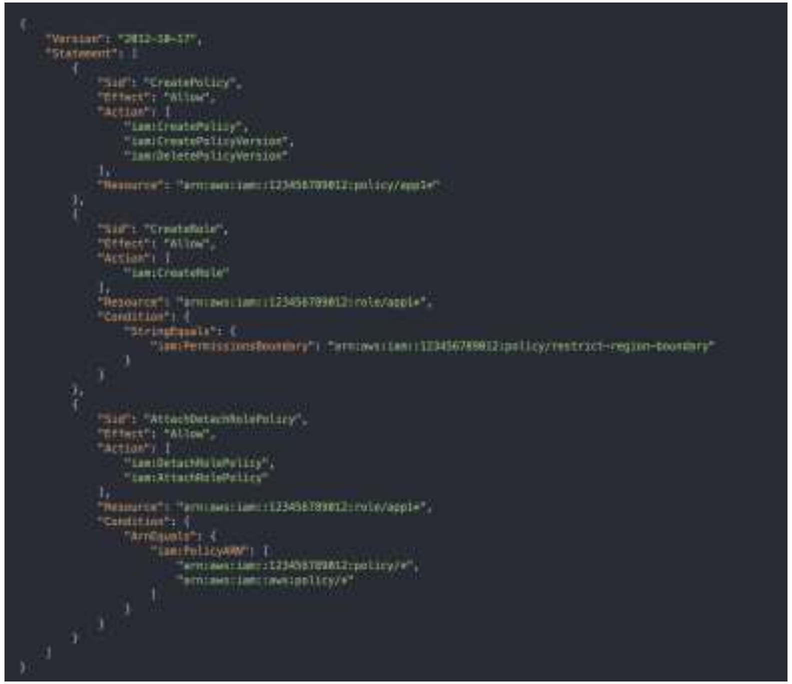
IAM Permissions Boundary

- Helps in setting the maximum permissions the which can be granted to users and roles they create and manage.
- Key for restriction to maximum possible permissions to an IAM.


Permission Boundaries - Example
You delegate responsibility to Jack for creating roles for operations on Amazon S3 and DynamoDB
Delegated Admins
"Bound" Users and Roles
Admin
Jack
John
Restricted resources
EC2, CloudWatch, IAM and others
They create users and roles that have those permission boundaries attached.
You
You delegate responsibility to Jill for creating roles for operations on Amazon EC2 and CloudWatch
Jill
Jane
Roles restricted by permissions
Restricted resources
S3, DynamoDB, IAM and others

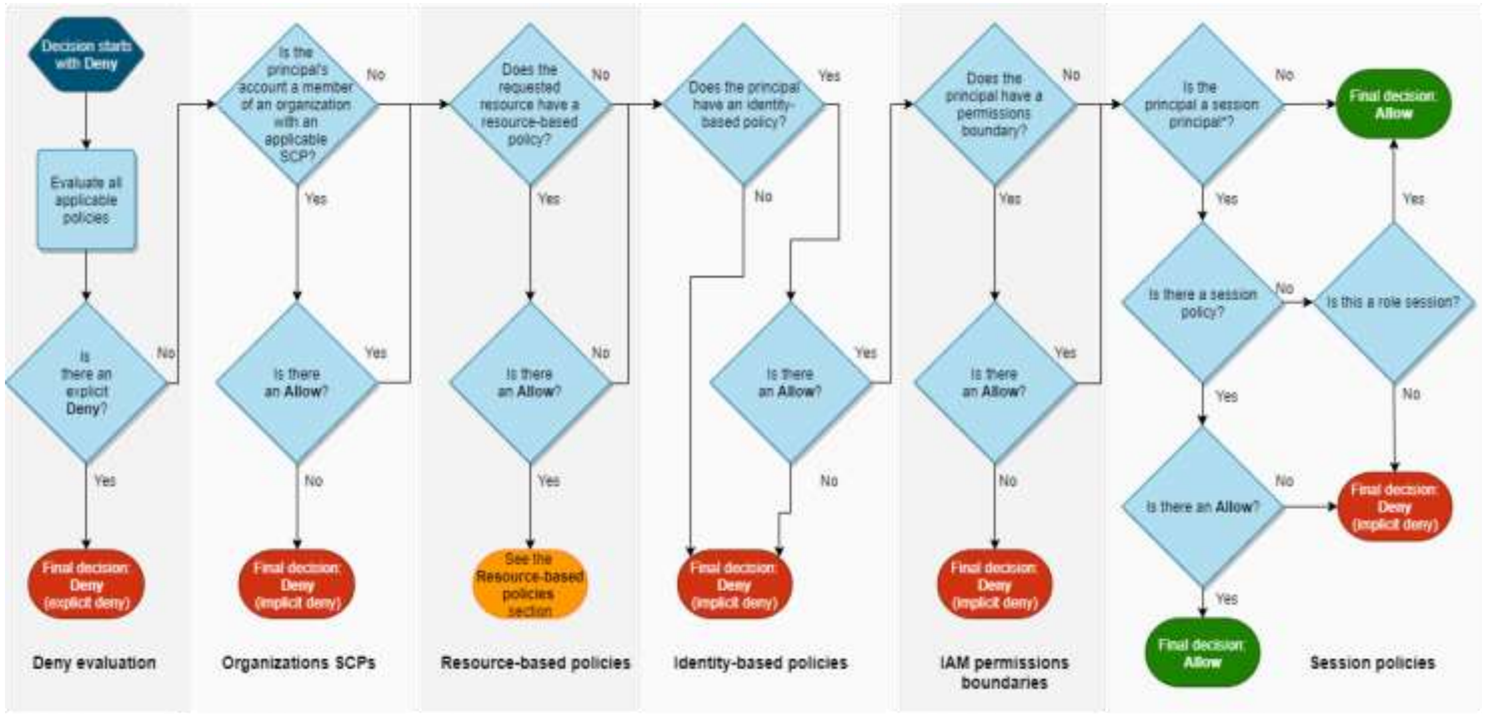
IAM Permissions Boundary

- Inline Policy
- Limit Max permissions that an IAM entity can have
- Prevent Privilege escalation.
- Applies to users and roles



```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "CreatePolicy",
      "Effect": "Allow",
      "Action": [
        "iam:CreatePolicy",
        "iam:CreatePolicyVersion",
        "iam:DeletePolicyVersion"
      ],
      "Resource": "arn:aws:iam::123456789012:policy/apple*"
    },
    {
      "Sid": "CreateRole",
      "Effect": "Allow",
      "Action": [
        "iam:CreateRole"
      ],
      "Resource": "arn:aws:iam::123456789012:role/apple*",
      "Condition": {
        "StringEquals": {
          "iam:PermissionsBoundary": "arn:aws:iam::123456789012:policy/restrict-region-boundary"
        }
      }
    },
    {
      "Sid": "AttachDetachRolePolicy",
      "Effect": "Allow",
      "Action": [
        "iam:DetachRolePolicy",
        "iam:AttachRolePolicy"
      ],
      "Resource": "arn:aws:iam::123456789012:role/apple*",
      "Condition": {
        "ArnEquals": {
          "iam:PolicyARN": [
            "arn:aws:iam::123456789012:policy/*",
            "arn:aws:iam::aws:policy/*"
          ]
        }
      }
    }
  ]
}
```

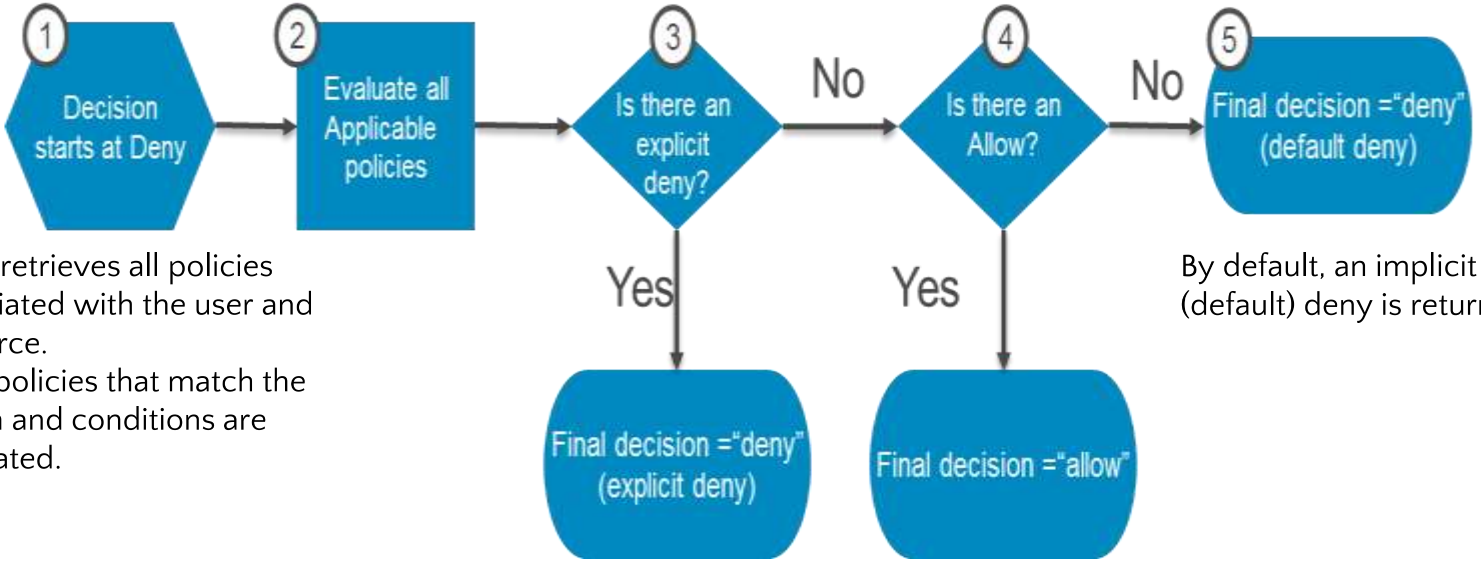
AWS Policy Evaluation



Decision starts with Deny
Evaluate all applicable policies
Is there an explicit Deny?
No
Yes
Final decision: Deny (explicit deny)
Deny evaluation
Is the principal's account a member of an organization with an applicable SCP?
No
Yes
Is there an Allow?
Yes
No
Final decision: Deny (implicit deny)
Organizations SCPs
Does the requested resource have a resource-based policy?
No
Yes
Is there an Allow?
No
Yes
See the Resource-based policies section
Resource-based policies
Does the principal have an identity-based policy?
Yes
No
Is there an Allow?
Yes
No
Final decision: Deny (implicit deny)
Identity-based policies
Does the principal have a permissions boundary?
No
Yes
Is there an Allow?
Yes
No
Final decision: Deny (implicit deny)
IAM permissions boundaries
Is the principal a session principal*?
No
Yes
Final decision: Allow
Is there a session policy?
No
Yes
Is this a role session?
Yes
No
Is there an Allow?
No
Yes
Final decision: Deny (implicit deny)
Final decision: Allow
Session policies

*A session principal is either a role session or an IAM federated user session.

AWS Policy Evaluation Logic


1
Decision starts at Deny
2
Evaluate all Applicable policies
3
Is there an explicit deny?
No
4
Is there an Allow?
No
5
Final decision ="deny" (default deny)
Yes
Final decision ="deny" (explicit deny)
Yes
Final decision ="allow"

AWS retrieves all policies associated with the user and resource.
Only policies that match the action and conditions are evaluated.

If policy statement has a deny, it wins over all other policy statements.

Access is granted if there is explicit allow and no deny.

By default, an implicit (default) deny is returned.

Ref: https://www.slideshare.net/AmazonWebServices/aws-iam-introduction

IAM Hands on Exploit

Implementing IAM Policies with Least Privilege to Managed S3 Bucket

- **Create IAM User:** Define a user with minimal permissions.
- **Policy Creation:** Attach a policy granting specific S3 access.
- **Validate Permissions:** Test user access to ensure least privilege.

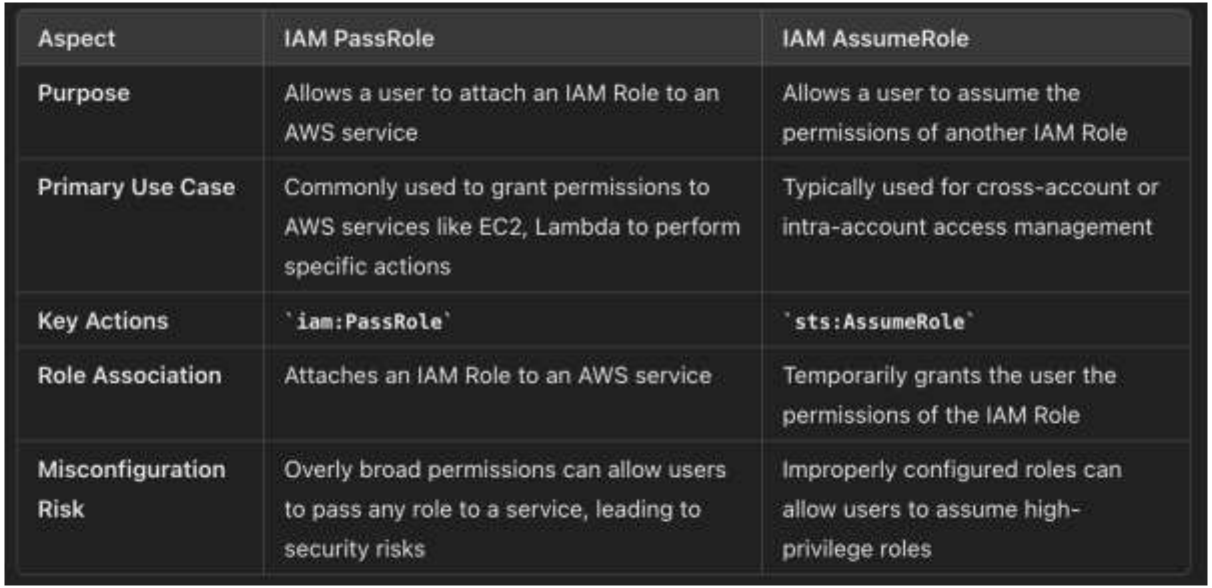
Exploiting IAM PassRole Misconfiguration

- **Define Role with PassRole Permission:** Allow user to pass specific roles.
- **Attach Policy:** Ensure the policy is appropriately scoped.
- **Exploitation Risk:** Highlight potential privilege escalation if misconfigured.

IAM AssumeRole Misconfiguration with Overly Permissive Role

- **Define Role with PassRole Permission**: Allow user to pass specific roles.
- **Attach Policy**: Ensure the policy is appropriately scoped.
- **Exploitation Risk**: Highlight potential privilege escalation if misconfigured.

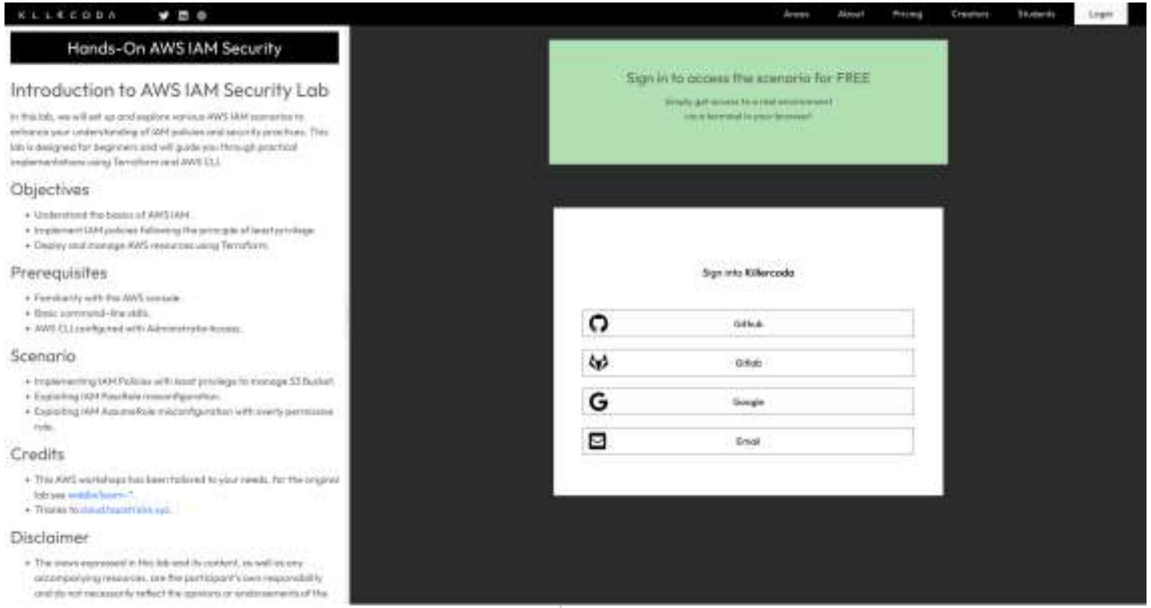
IAM PassRole vs IAM AssumeRole



Aspect	IAM PassRole	IAM AssumeRole
Purpose	Allows a user to attach an IAM Role to an AWS service	Allows a user to assume the permissions of another IAM Role
Primary Use Case	Commonly used to grant permissions to AWS services like EC2, Lambda to perform specific actions	Typically used for cross-account or intra-account access management
Key Actions	`iam:PassRole`	`sts:AssumeRole`
Role Association	Attaches an IAM Role to an AWS service	Temporarily grants the user the permissions of the IAM Role
Misconfiguration Risk	Overly broad permissions can allow users to pass any role to a service, leading to security risks	Improperly configured roles can allow users to assume high-privilege roles

https://demacia.medium.com/difference-between-iam-passrole-and-iam-assumerole-en-id-3cb1ffd71a36

https://killercoda.com/cloudsecurity-scenario



Hands-On AWS IAM Security

Introduction to AWS IAM Security Lab

In this lab, we will set up and explore various AWS IAM scenarios to enhance your understanding of IAM policies and security practices. This lab is designed for beginners and will guide you through practical implementations using Terraform and AWS CLI.

Objectives

- Understand the basics of AWS IAM.
- Implement IAM policies following the principle of least privilege.
- Deploy and manage AWS resources using Terraform.

Prerequisites

- Familiarity with the AWS console.
- Basic command-line skills.
- AWS CLI configured with Administrator Access.

Scenario

- Implementing IAM Policies with least privilege to manage S3 Bucket.
- Exploring IAM PassRole misconfiguration.
- Exploiting IAM AssumeRole misconfiguration with overly permissive role.

Credits

- This AWS workshops has been tailored to your needs. for the original lab see [illegible].
- Thanks to [illegible].

Disclaimer

- The views expressed in this lab and its content, as well as any accompanying resources, are the participant's own responsibility and do not necessarily reflect the opinions or endorsements of the

Sign in to access the scenario for FREE

Simply get access to a real environment via a terminal in your browser!

Sign into Killercoda

Killercoda Free Community

AWS IAM Killercoda Lab

References & Credits

- chatgpt.com
- killercoda.com
- docs.aws.amazon.com
- cloud.hacktricks.xyz
- steampipe.io/blog/aws-iam-policy-wildcards-reference
- www.tenable.com/blog

Thanks!

Any questions ?

You can find me at

- /@justm0rph3u5
- justmorpheus1@gmail.com
- **training@securitydojo.co.in**