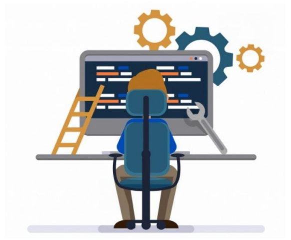
Identity and Access Management

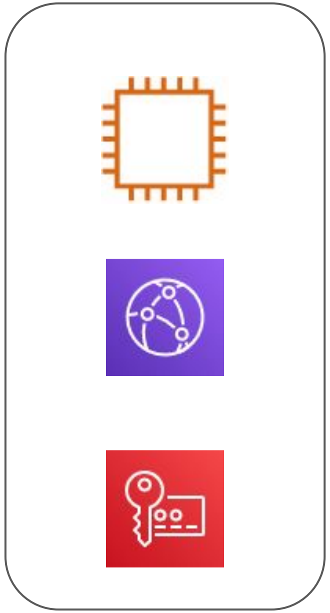
Access Management

Overview of IAM

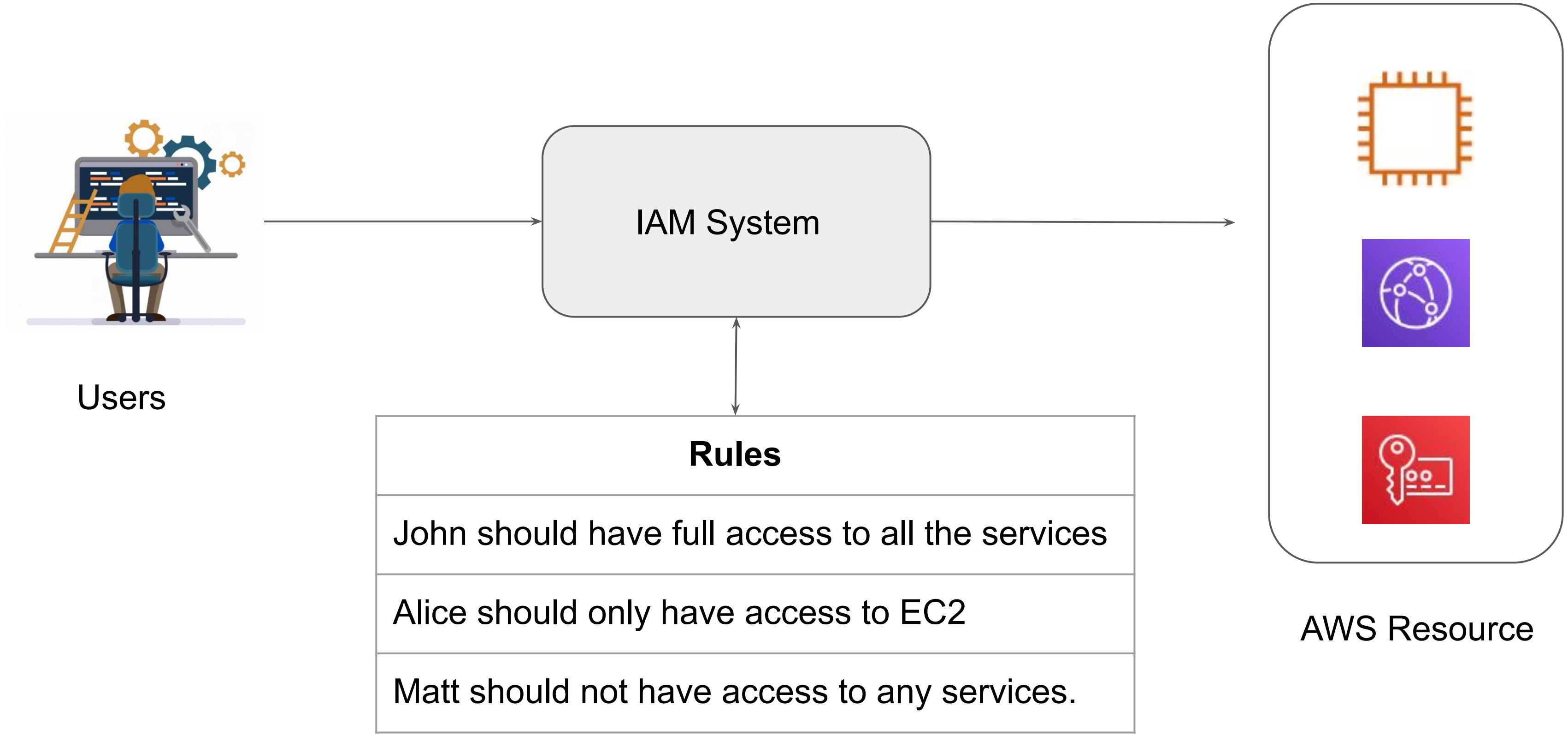
Identity and Access Management is a framework of policies for ensuring that the proper people in an enterprise have the appropriate access to technology resources.



Rules
John should have full access to all the services
Alice should only have access to EC2
Matt should not have access to any services.

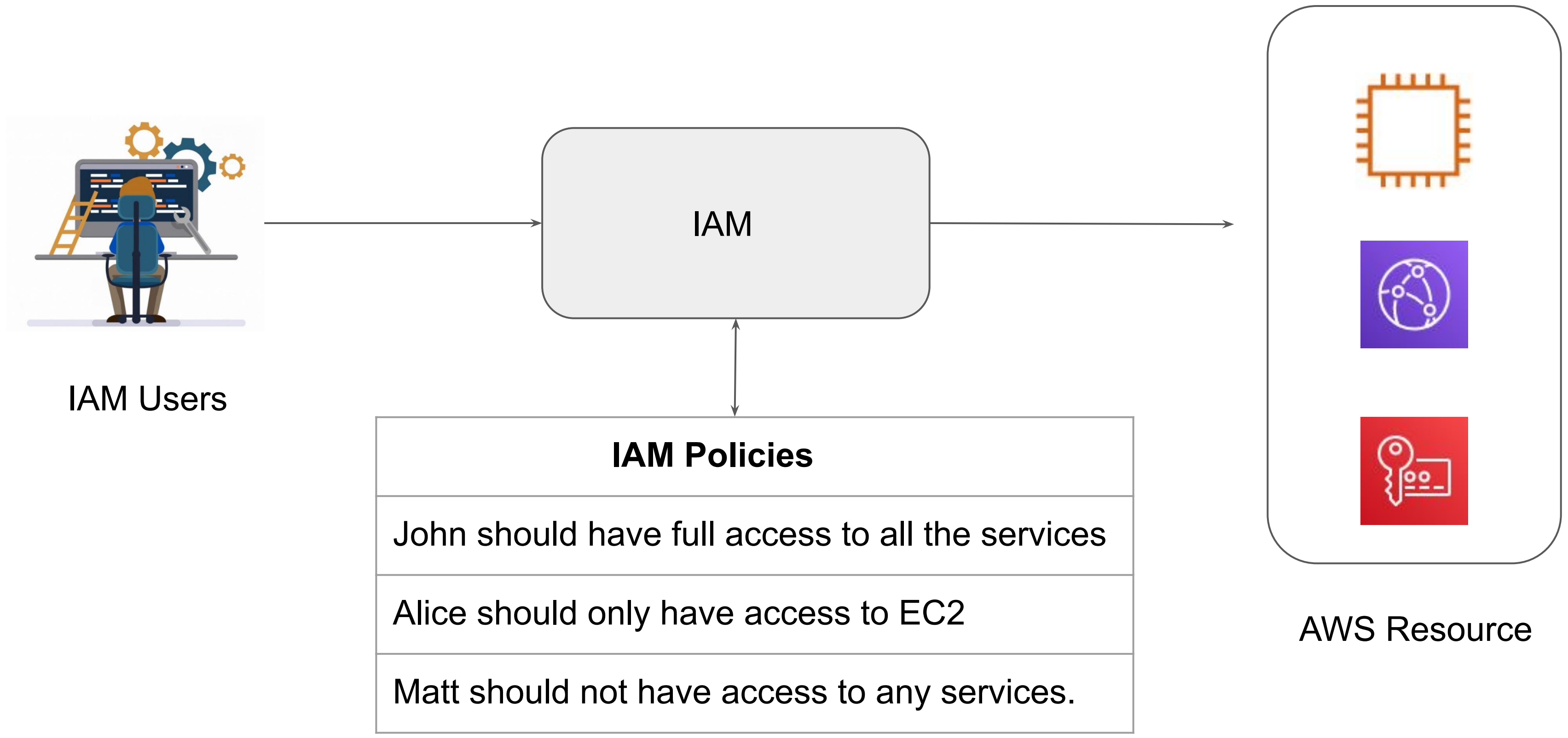


Simplifying the Architecture



Rules
John should have full access to all the services
Alice should only have access to EC2
Matt should not have access to any services.

AWS Terminology

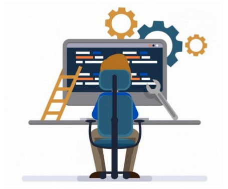


IAM Policies
John should have full access to all the services
Alice should only have access to EC2
Matt should not have access to any services.

IAM User

IAM user is an entity that you create in AWS to represent the person or application that uses it to interact with AWS.

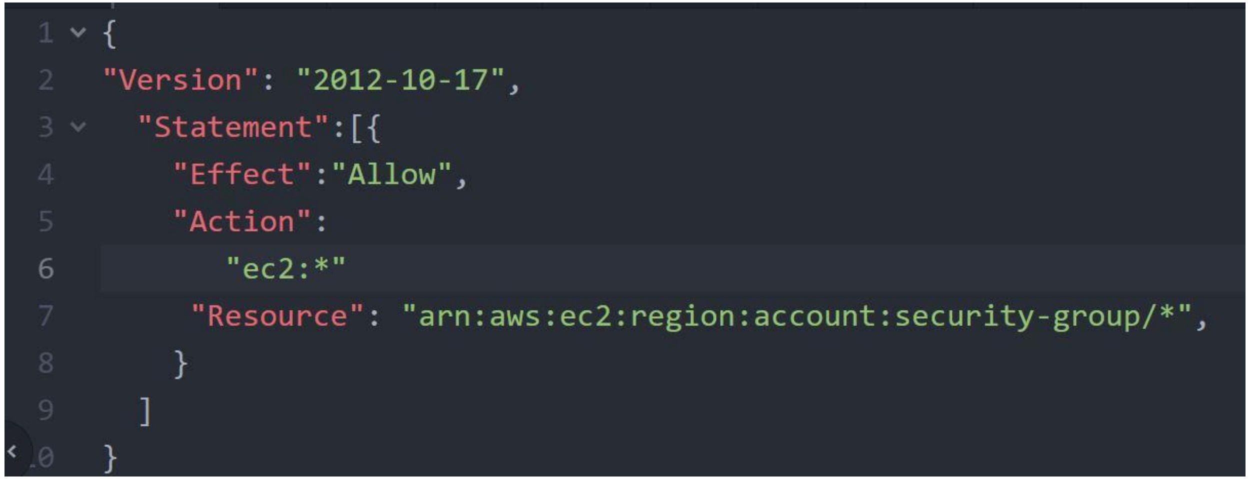
By default, the IAM user does not have any permission associated with it.



IAM Policy

IAM Policy is an object in AWS that defines the permission of a specific object.

AWS evaluates the policies, and depending on that, the permission is granted or denied.

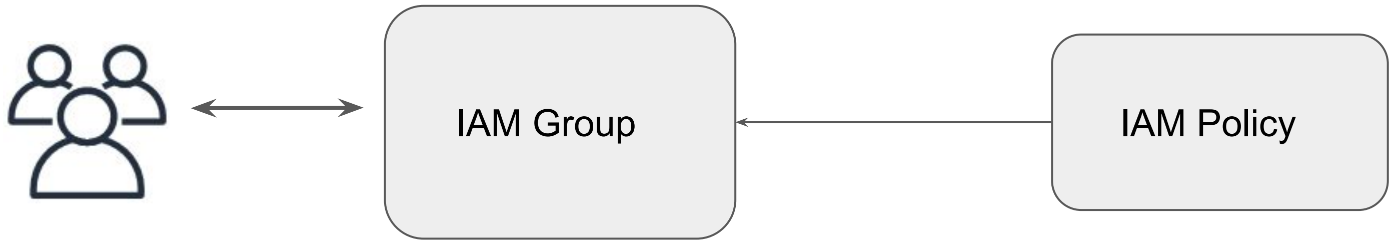


```
{
"Version": "2012-10-17",
  "Statement":[{
    "Effect":"Allow",
    "Action":
       "ec2:*"
     "Resource": "arn:aws:ec2:region:account:security-group/*",
    }
  ]
}
```

IAM Groups

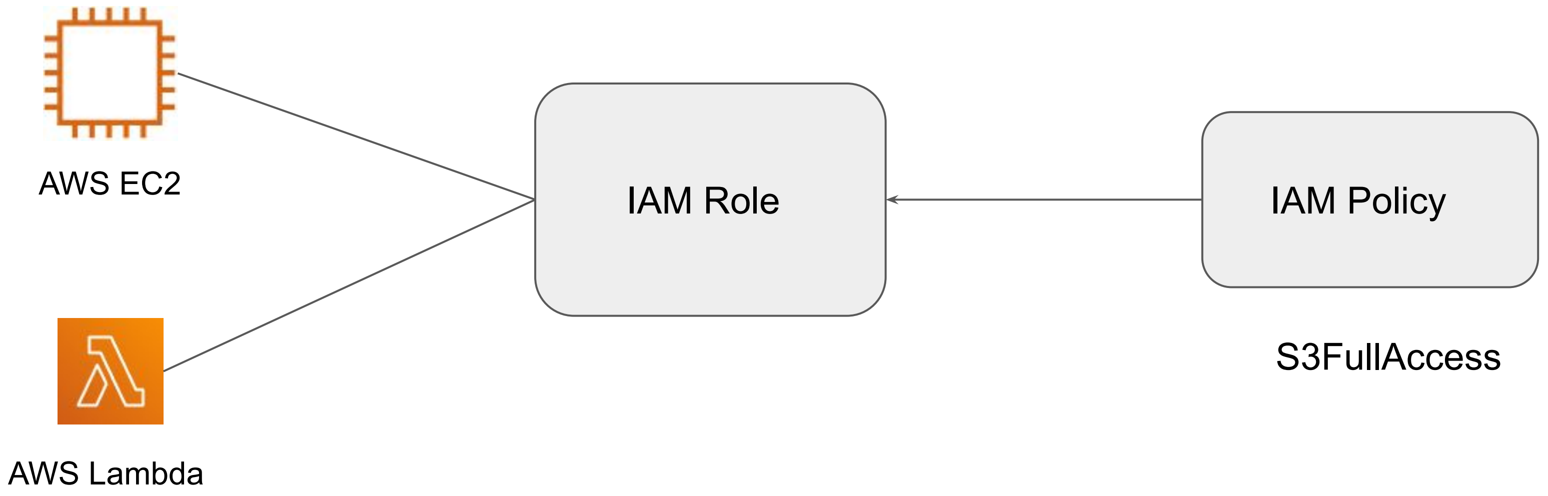
An IAM group is a collection of IAM users.

IAM policy attached to the group is associated with all the users that are part of the group.



IAM Role

An IAM role is similar to an IAM user, in that it is an AWS identity with permission policies that determine what the identity can and cannot do in AWS.



IAM Policies

Access Management

Identity Access Management

- IAM Policies allows us to define at granular level access on what permissions needs to be given to access a particular AWS resource.

- There are 4 important elements of IAM Policy :

Statement
    Effect
    Action
    Resource

→

```
{
    "Statement": {
        "Effect": "Allow",
        "Action": "ec2:Describe*",
        "Resource": "*"
    }
}
```

Component 0

- Statement element is the main element for the IAM policy.
- This element is a must.

- The statement element can contain multiple individual statements in order of [{ .. } {...} . Each of the individual statement is enclosed in blocks of { }

```
{
    "Statement": {
        "Effect": "Allow",
        "Action": "ec2:Describe*",
        "Resource": "*"
    }
}
```

Example policy with multiple statements

```
{
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "ec2:Describe*",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "elasticloadbalancing:Describe*",
            "Resource": "*"
        }
  ]
}
```

Component 1

- Effect basically specified whether the statement is allowed or explicitly denied.
- There are two possible values : Allow or Deny

```
{
    "Statement": {
        "Effect": "Allow",
        "Action": "ec2:Describe*",
        "Resource": "*"
    }
}
```

Component 2

- Action element defines list of actions that will be allowed or denied.
- Each AWS service has it’s own set of actions.

Example :

- ec2:CreateKeyPair
- ec2:CreateVpc

- sqs:ListQueues
- sqs:SendMessage

- s3:CreteBucket
- s3:DeleteBucket

Component 3

- Resource element defines the object that the statement covers.
- Amazon Resource Name (ARN) uniquely identified the AWS resources.

Example :

arn:aws:ec2:us-west-2:836802967410:instance/i-0067dce1525f98ab2

$\downarrow$

arn:aws:ec2:region:account-id:instance/instance-id

IAM Policies 02

Access Management

Use Case

Alice has been given one EC2 instance for testing. As part of the access, she must be allowed to start and stop her EC2 instance. The instance ID is i-0508b5c481e123fd9 in the Oregon region.

Component 3

- Resource element defines the object that the statement covers.

- Amazon Resource Name (ARN) uniquely identified the AWS resources.

Example :

arn:aws:ec2:us-west-2:836802967410:instance/i-0067dce1525f98ab2

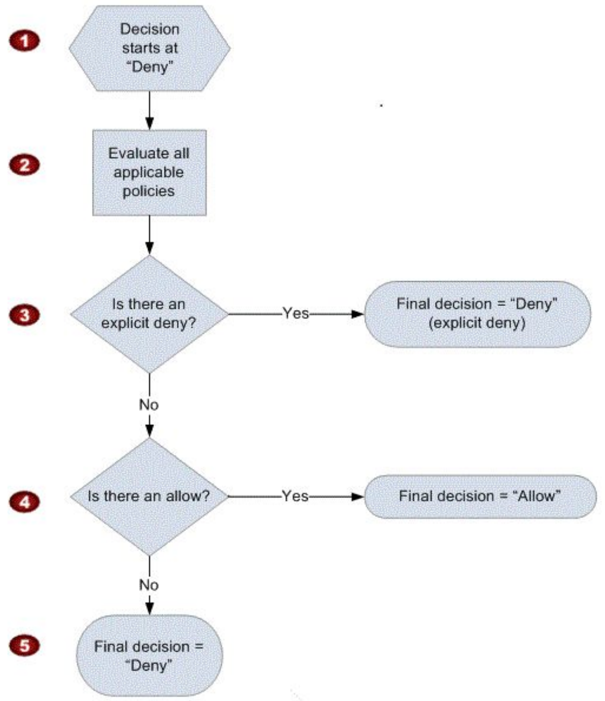
$\downarrow$

arn:aws:ec2:region:account-id:instance/instance-id

IAM Policy Evaluation Logic

Access Management

Decision Making Process


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

1) Decision starts with assumption that the request will be denied.

2) Then, all the attached policies are evaluated.

3) Code will look if there is any explicit deny in the policy.

4) If explicit deny is not found, code will look for allowed instruction and if yes then decision is allowed.

5) If no allow is found, decision is deny.

Example Scenario

- Four AWS S3 buckets are available
- We want to give access to all the buckets in our S3, except 1

Step 1 :

Allow access to all the buckets in S3.

Step 2:

Deny access to bucket 05

Explicit Deny vs Deny by Default

- A request will be denied by default if there is no allow policy present for the resource.

Example:

If user has EC2 ReadOnlyAccess and tries to open S3 console, it will be denied.

- A request will be denied irrespective of full access if it is explicitly denied in the policy

Example:

Deny user from accessing bucket 05.

Condition Element

Mastering IAM

Overview of Condition Element

The Condition element lets you specify conditions for when a policy is in effect.

We build it by making use of condition operators like (equal, less than, etc).

Sample policy:

Allow full access to User Alice on EC2 Resource {ONLY WHEN CONDITION IS MET}

```
"Condition": {
   "DateGreaterThan" : {
       "aws:CurrentTime" : "2013-12-15T12:00:00Z"
    }
}
```

Condition Operators

There are multiple condition operators available which you can make use of .

Some of these include:

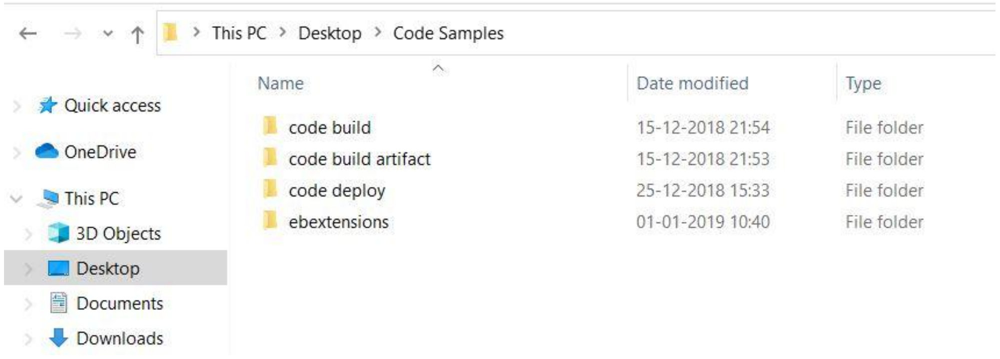
- String Condition Operators
- Numeric Condition Operators
- Date Condition Operators
- Boolean Condition Operators
- Binary Condition Operators
- IP Address Condition Operators
- ARN Condition Operators

AWS Command Line Interface (CLI)

Access Management

GUI and CLI

GUI Console



Name	Date modified	Type
code build	15-12-2018 21:54	File folder
code build artifact	15-12-2018 21:53	File folder
code deploy	25-12-2018 15:33	File folder
ebextensions	01-01-2019 10:40	File folder

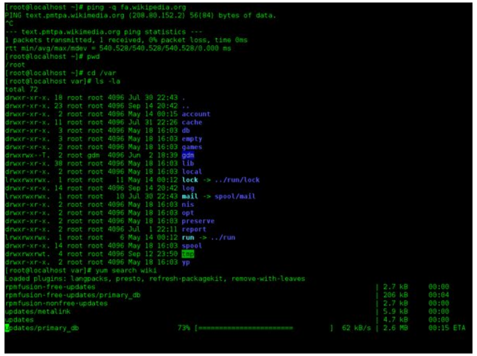
CLI

```
 Directory of C:\Users\Zeal Vora\Desktop\Code Samples

31-12-2018  19:01    <DIR>          .
31-12-2018  19:01    <DIR>          ..
15-12-2018  21:54    <DIR>          code build
15-12-2018  21:53    <DIR>          code build artifact
25-12-2018  15:33    <DIR>          code deploy
01-01-2019  10:40    <DIR>          ebextensions
               0 File(s)              0 bytes
               6 Dir(s)   4,373,610,496 bytes free
```

CLI is always faster

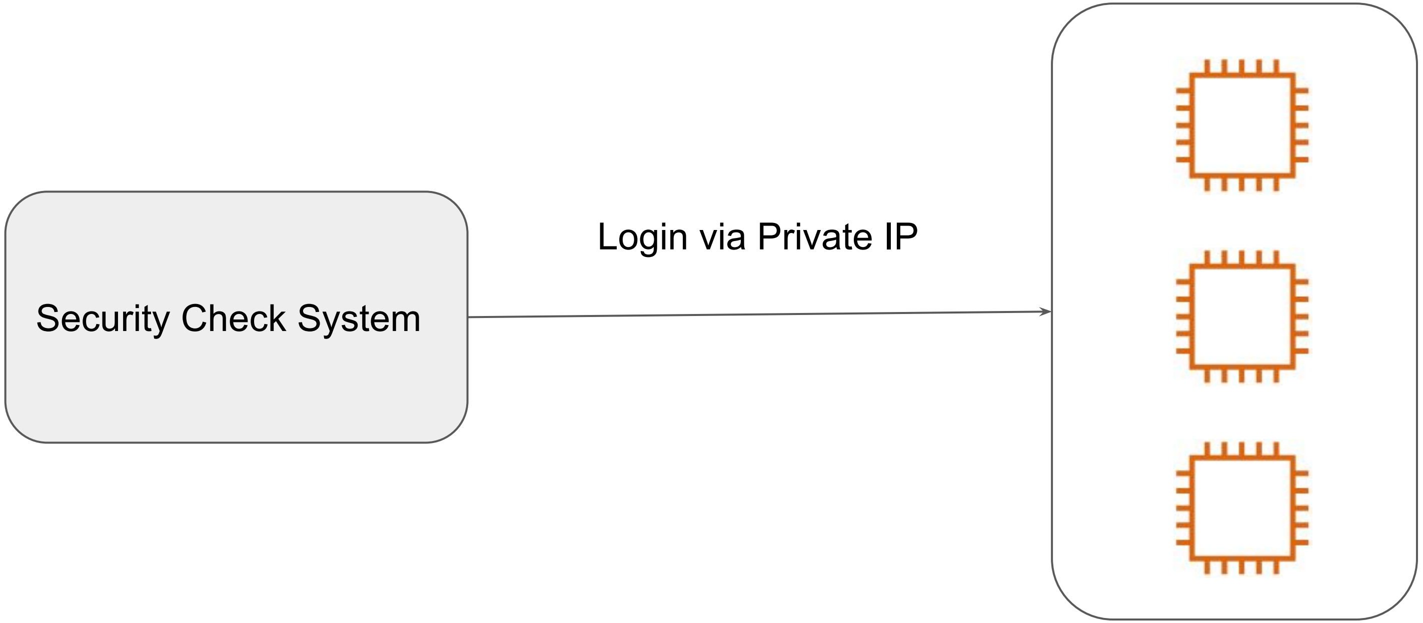
- Command Line Interface (CLI) is way of interacting with the system in form of commands
- It is considered as fastest way of doing things in an repeated, automated fashion.



```
[root@localhost ~]# ping -q fa.wikipedia.org
PING text.pmtpa.wikimedia.org (208.80.152.2) 56(84) bytes of data.
^C
--- text.pmtpa.wikimedia.org ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 540.528/540.528/540.528/0.000 ms
[root@localhost ~]# pwd
/root
[root@localhost ~]# cd /var
[root@localhost var]# ls -la
total 72
drwxr-xr-x. 18 root root 4096 Jul 30 22:43 .
drwxr-xr-x. 23 root root 4096 Sep 14 20:42 ..
drwxr-xr-x.  2 root root 4096 May 14 00:15 account
drwxr-xr-x. 11 root root 4096 Jul 31 22:26 cache
drwxr-xr-x.  3 root root 4096 May 18 16:03 db
drwxr-xr-x.  3 root root 4096 May 18 16:03 empty
drwxr-xr-x.  2 root root 4096 May 18 16:03 games
drwxrwx--T.  2 root gdm  4096 Jun  2 18:39 gdm
drwxr-xr-x. 38 root root 4096 May 18 16:03 lib
drwxr-xr-x.  2 root root 4096 May 18 16:03 local
lrwxrwxrwx.  1 root root   11 May 14 00:12 lock -> ../run/lock
drwxr-xr-x. 14 root root 4096 Sep 14 20:42 log
lrwxrwxrwx.  1 root root   10 Jul 30 22:43 mail -> spool/mail
drwxr-xr-x.  2 root root 4096 May 18 16:03 nis
drwxr-xr-x.  2 root root 4096 May 18 16:03 opt
drwxr-xr-x.  2 root root 4096 May 18 16:03 preserve
drwxr-xr-x.  2 root root 4096 Jul  1 22:11 report
lrwxrwxrwx.  1 root root    6 May 14 00:12 run -> ../run
drwxr-xr-x. 14 root root 4096 May 18 16:03 spool
drwxrwxrwt.  4 root root 4096 Sep 12 23:50 tmp
drwxr-xr-x.  2 root root 4096 May 18 16:03 yp
[root@localhost var]# yum search wiki
Loaded plugins: langpacks, presto, refresh-packagekit, remove-with-leaves
rpmfusion-free-updates                                                          | 2.7 kB     00:00
rpmfusion-free-updates/primary_db                                               | 206 kB     00:04
rpmfusion-nonfree-updates                                                       | 2.7 kB     00:00
updates/metalink                                                                | 5.9 kB     00:00
updates                                                                         | 4.7 kB     00:00
updates/primary_db                    73% [=======================          ]  62 kB/s | 2.6 MB     00:15 ETA
```

Simple Use-Case

- A new “Security Check System” verifies the security of an EC2 instance
- It evaluates the system configuration after logging into the EC2 instance.
- It needs Private IP as an Input.



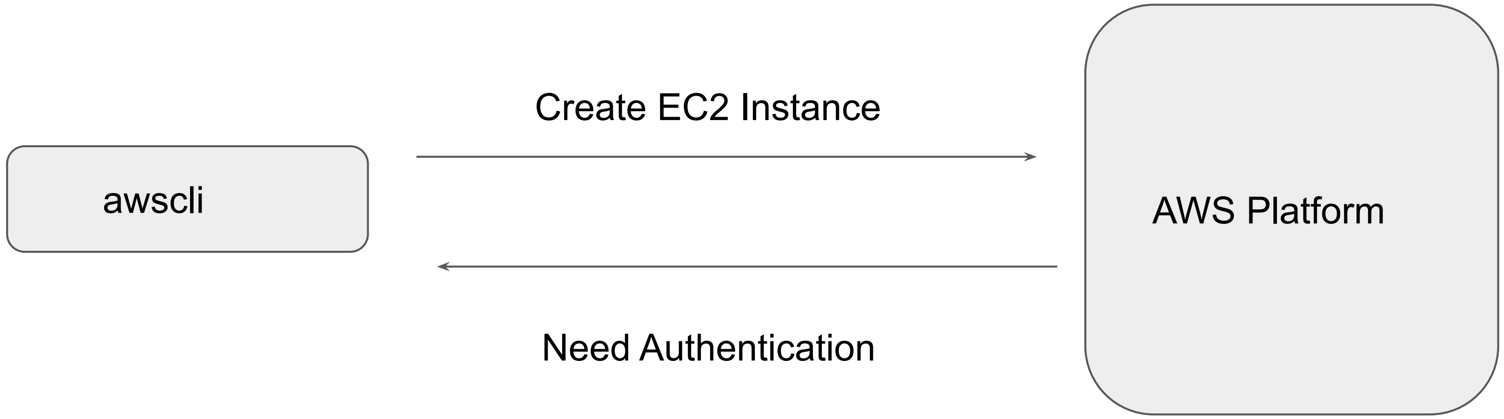
AWS CLI

- AWS CLI is used for managing AWS resources from the terminal.
- It makes room for automation & make things much more faster.

AWS CLI Pre-Requisites

There are two primary prerequisites for AWS CLI

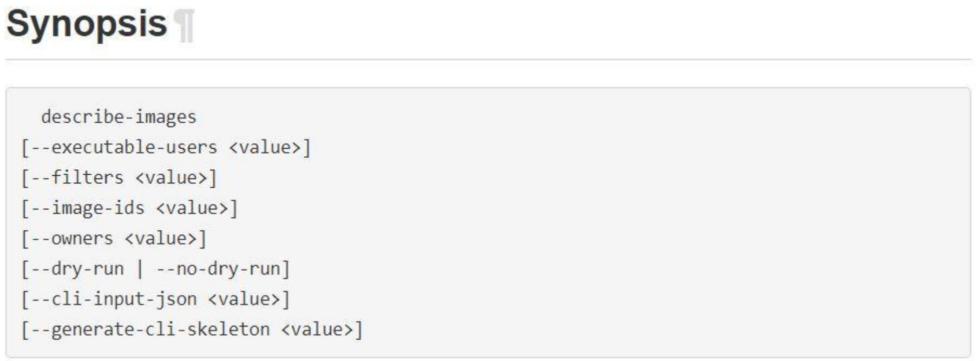
1. AWS CLI Binary
2. AWS Access Secret Keys OR IAM Role



AWS CLI Documentation

There are separate set of documentation available for AWS CLI

These covers the list of all options that can be used while making use of AWS CLI.



Synopsis ¶

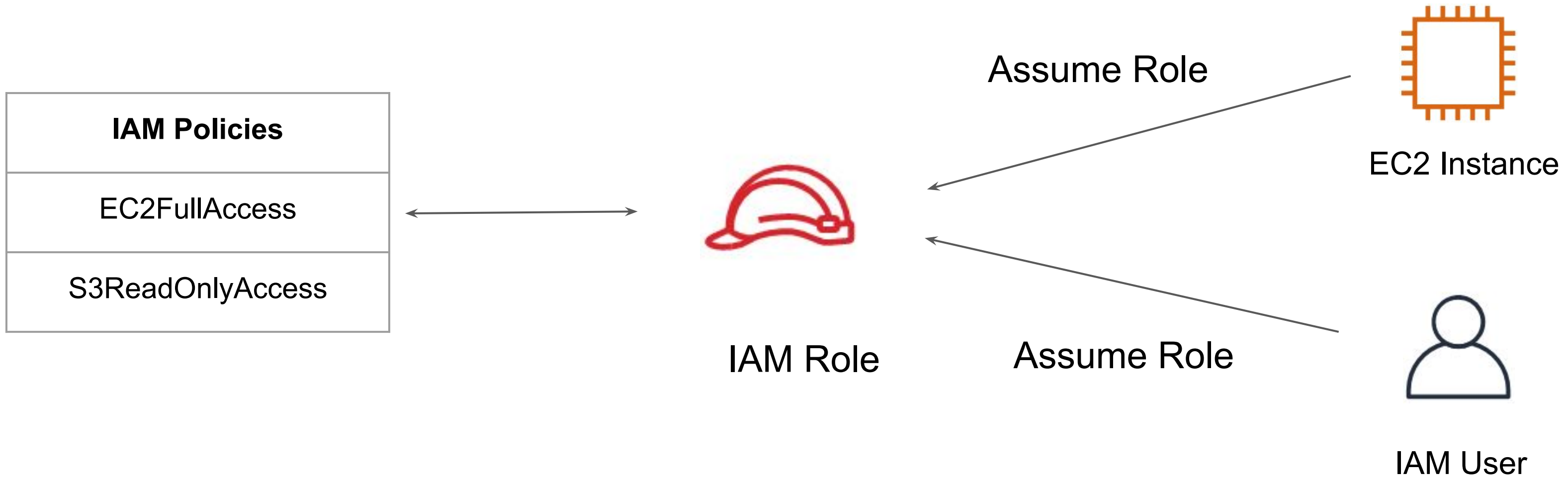
```
  describe-images
[--executable-users <value>]
[--filters <value>]
[--image-ids <value>]
[--owners <value>]
[--dry-run | --no-dry-run]
[--cli-input-json <value>]
[--generate-cli-skeleton <value>]
```

IAM Roles

Access Management

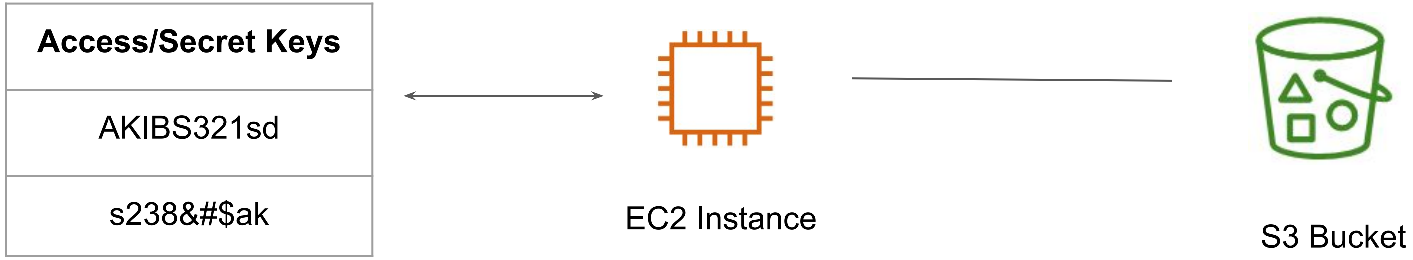
Overview of IAM Role

IAM Role is an entity that contains set of policies and any resource assuming that role will be able to have permissions mentioned in the role.



Sample Use-Case- No No Scenario

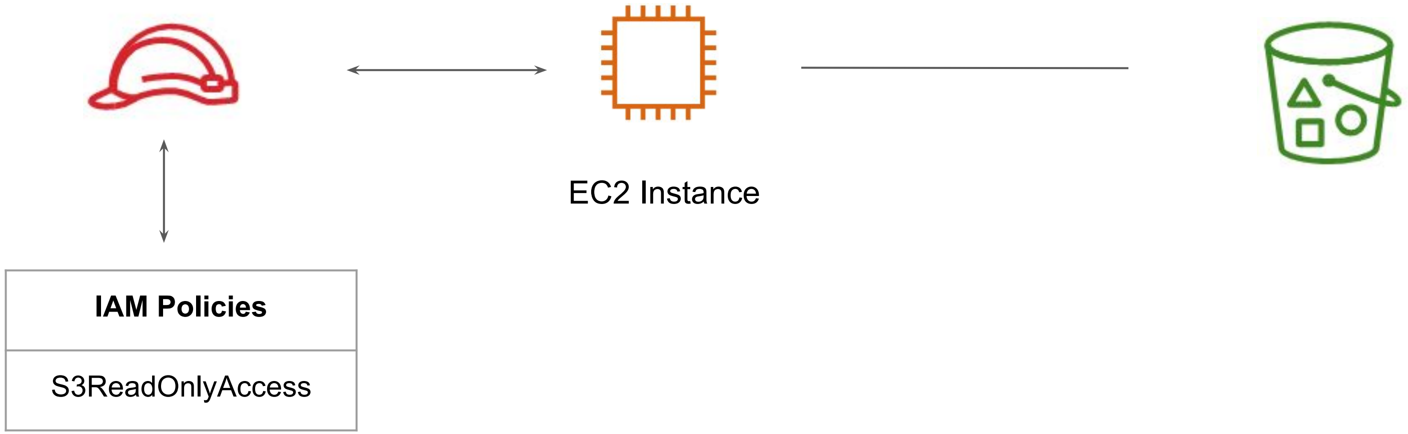
EC2 Instance wants to get data from an S3 bucket.



Access/Secret Keys
AKIBS321sd
s238&#$ak

Example - Best Practice

It is recommended to always make use of IAM Role instead of hard coding the AWS Access Keys within EC2 instance / software code.



IAM Permissions Boundaries

Advanced IAM

Getting Started

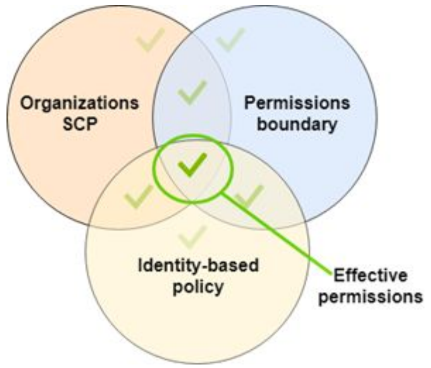
A permissions boundary is an advanced feature in which you use a managed policy to set the maximum permissions that an identity-based policy can grant to an IAM entity.

When you set a permissions boundary for an entity, the entity can perform only the actions that are allowed by both its identity-based policies and its permissions boundaries.

Evaluating Effective Permission with Boundaries

The effective permissions for an entity are the permissions that are granted by all the policies associated with the user/role/account.

Within an AWS account, the permissions for an entity can be affected by identity-based policies, resource-based policies, permissions boundaries, Organizations SCPs, or session policies.

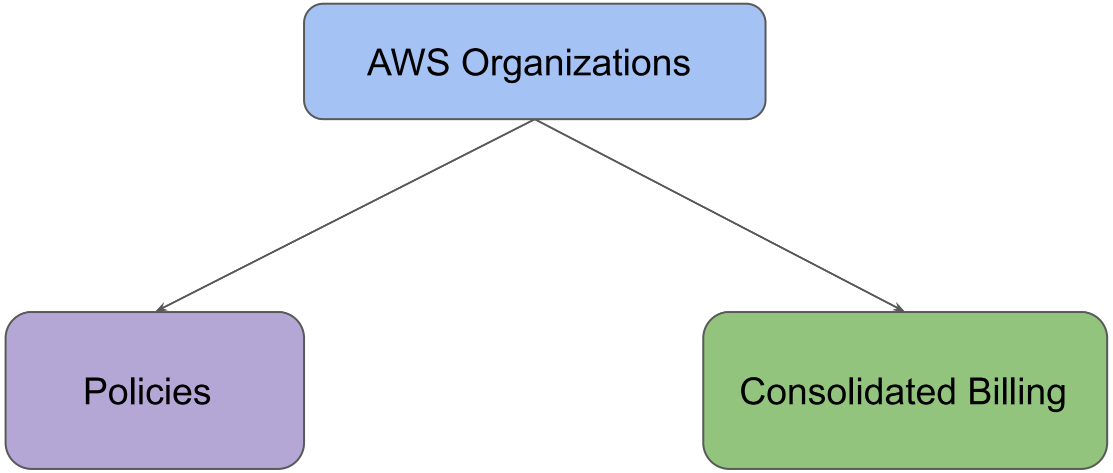


AWS Organizations

Centralized Control

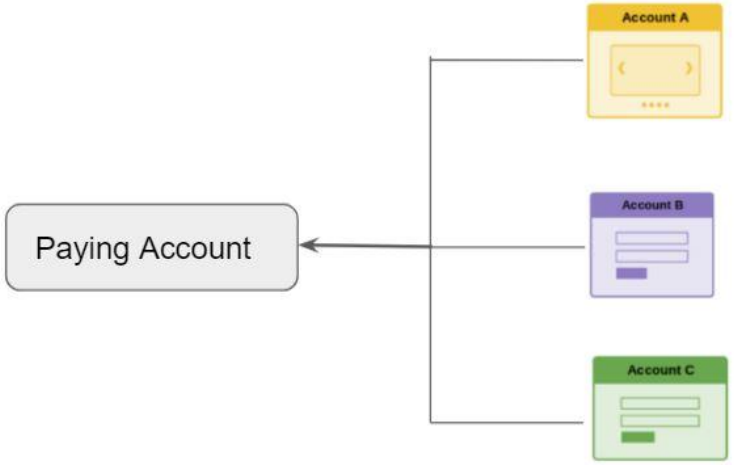
Getting the basics right

AWS offers centralized policy-based management as well as the feature of consolidated billing for multiple AWS accounts through the feature of AWS Organizations.



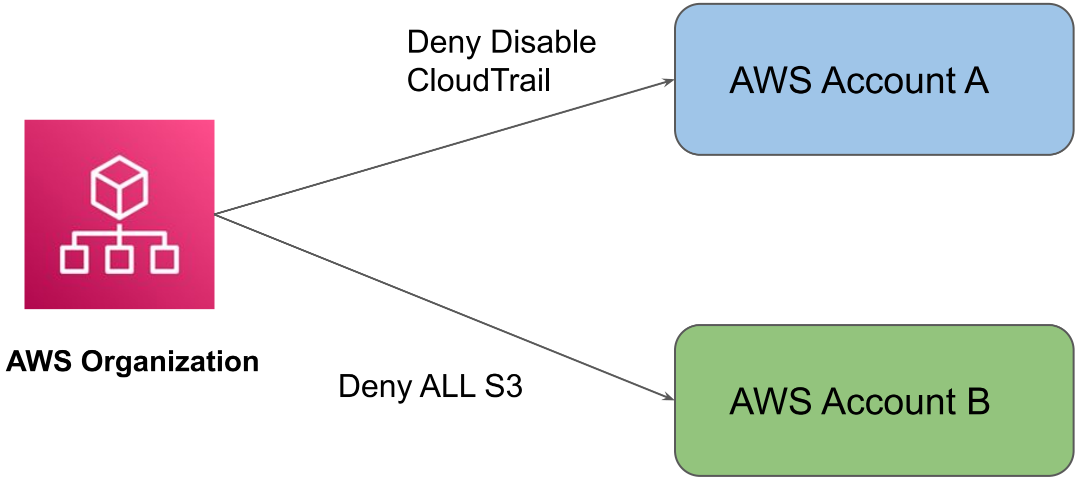
Part 1 - Consolidated Billing

In consolidated billing, management account to access the billing information and pay for all member accounts.



Part 2 - Policies

Policies in AWS Organizations enable you to apply additional types of management to the AWS accounts in your organizations.

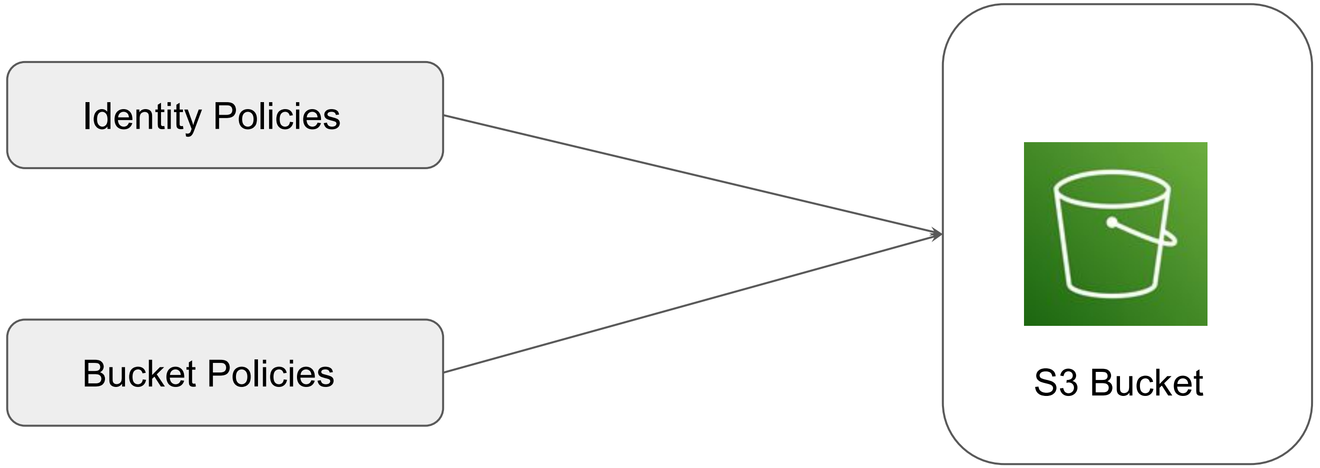


S3 Bucket Policy

Bucket Policies

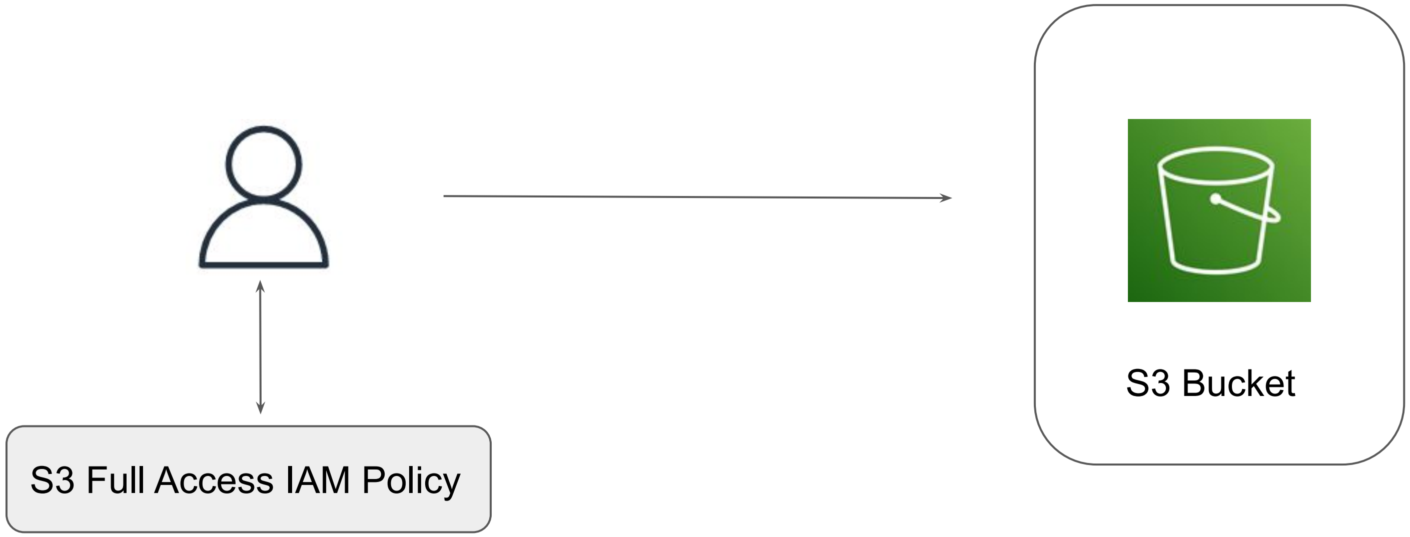
Granting Permission for S3 Resource

There are two primary ways in which a permission to a S3 resource is granted.



Use-Case 1: IAM User Needs Access to S3 Bucket

IAM User Named Bob needs Full Access to S3 Bucket.

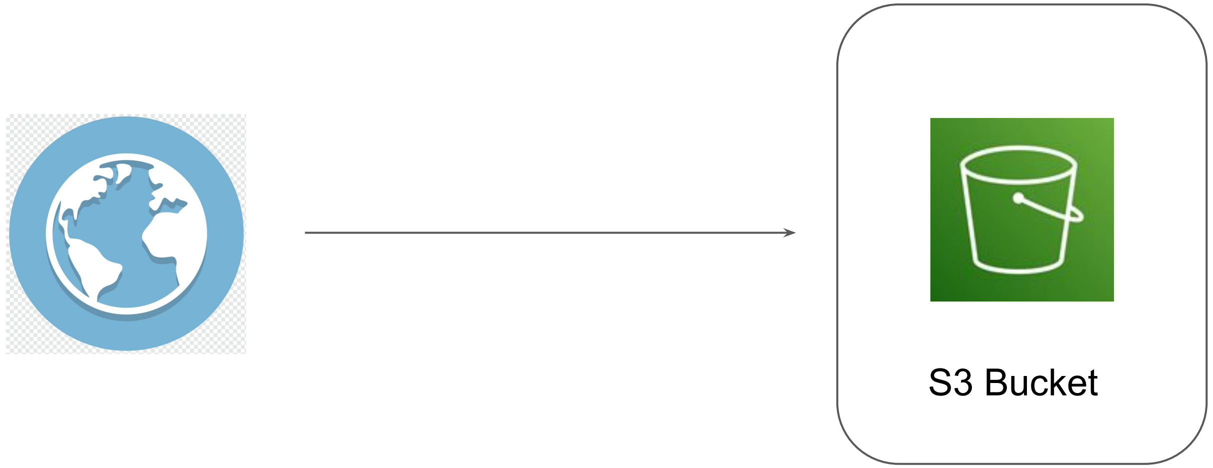


Wider Scope of S3 Bucket

Files within the S3 bucket can have scope beyond the IAM entity.

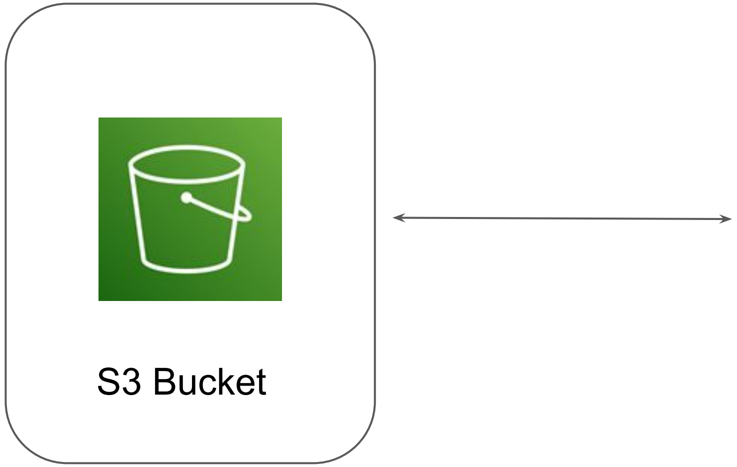
Organization can host entire websites in S3 Bucket.

S3 Buckets can even be used to host central files for download.



S3 Bucket Policy

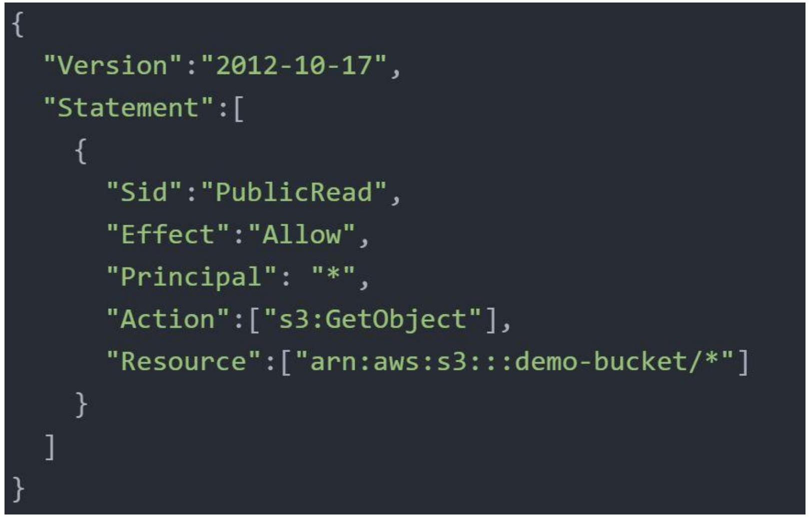
A bucket policy is a resource-based AWS IAM policy associated with the S3 Bucket to control access permissions for the bucket and the objects in it .



S3 Bucket Policy Rules
Allow Access from Internet
Allow Access from only 10.77.3.20 IP
Only Allow Access from Specific VPC
Only allow HTTPS connections

Bucket Policy 1 - Public Access

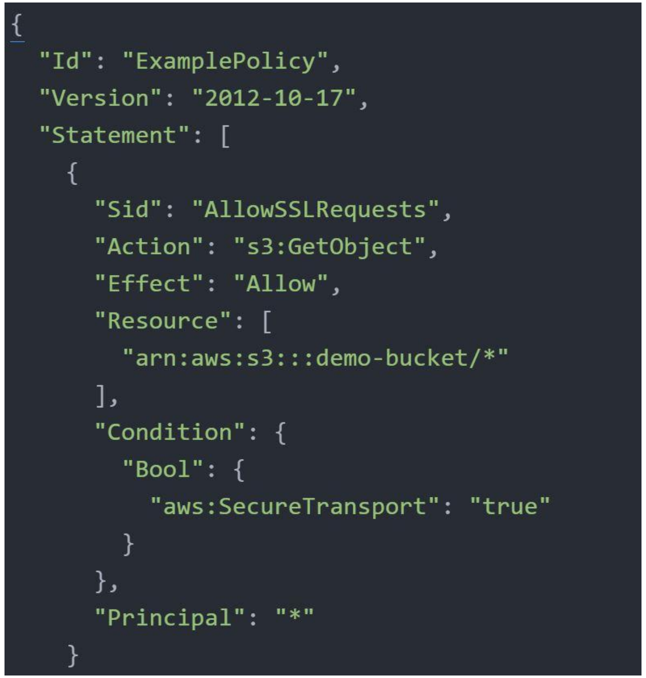
The following example policy grants the s3:GetObject permission to any public anonymous users.



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

Bucket Policy 2 - Only HTTPS

Only the HTTPS requests should be allowed. All HTTP requests should be blocked.



```
{
  "Id": "ExamplePolicy",
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowSSLRequests",
      "Action": "s3:GetObject",
      "Effect": "Allow",
      "Resource": [
        "arn:aws:s3:::demo-bucket/*"
      ],
      "Condition": {
        "Bool": {
          "aws:SecureTransport": "true"
        }
      },
      "Principal": "*"
    }
```

Join us in our Adventure

Be Awesome

kplabs.in/twitter

kplabs.in/linkedin