Class 29

**Bucket Policy**

i, S3 -> Amazon S3 -> dheerajpalvai.xyz -> **[Permissions]** -> Bucket Policy

Access Control List & Bucket Policy are two policies

**Access Control List**: is a legacy, there is no granular level control to block some IP’s or give access to particular IP

**Bucket Policy**: gives particular level of access to block some particular IP’s or Access some particular IP’s

**Make bucket Public:**

* We gave “make public” for each file, instead of that we have to make public for whole bucket

**Google**: aws s3 bucket policy example

Granting Read-Only Permission to an Anonymous User

{

"Version":"2012-10-17",

"Statement":[

{

"Sid":"PublicRead",

"Effect":"Allow",

"Principal": "\*",

"Action":["s3:GetObject","s3:GetObjectVersion"],

"Resource":["arn:aws:s3:::*awsexamplebucket1*/\*"]

}

]

}

* Copy Bucket policy editor ARN**: arn:aws:s3:::dheerajpalvai.xyz** and paste in “Resource”
* Then Copy and paste whole code in Bucket policy text area -> save (Then [permissions] & Bucket Policies become public)
* Now S3 -> Amazon S3 -> dheerajpalvai.xyz -> **[Overview]** -> upload (hello.html) (cloud.html) files
* And try in URL dheerajpalvai.xyz/hello.html it come, if it doesn’t come go to **[Properties]** -> Default encryption (None)

-----------------------

Now all files became public and those can see by every user

**Access IP Level**

🡺Now I want to allow public access to only particular IP’s

For example, take Two user ips (internet router ips)

**Google**: <https://whatismyipaddress.com/>

Access only to this ip: [**73.119.241.21**](https://whatismyipaddress.com/ip/73.119.241.21)

## Limiting Access to Specific IP Addresses

{

"Version": "2012-10-17",

"Id": "S3PolicyId1",

"Statement": [

{

"Sid": "IPAllow",

"Effect": "Deny",

"Principal": "\*",

"Action": "s3:\*",

"Resource": [

"arn:aws:s3:::*awsexamplebucket1*",

"arn:aws:s3:::*awsexamplebucket1*/\*"

],

"Condition": {

"NotIpAddress": {"aws:SourceIp": "*54.240.143.0/24*"}

}

}

]

}

Copy code in “Visual studio code” and Format Document (Shift + Alt + F)

Allow ips

{

    "Version": "2012-10-17",

    "Id": "S3PolicyId1",

    "Statement": [

        {

            "Sid": "IPAllow",

            "Effect": "Allow",

            "Principal": "\*",

            "Action": "s3:\*",

            "Resource": [

                "arn:aws:s3:::dheerajpalvai.xyz/\*"

            ],

            "Condition": {

                "IpAddress": {

                    "aws:SourceIp": "73.119.241.21/32"

                }

            }

        }

    ]

}

🡺Now I want to Deny public access to only particular IP’s

{

    "Version": "2012-10-17",

    "Id": "S3PolicyId1",

    "Statement": [

        {

            "Sid": "IPAllow",

            "Effect": "Deny",

            "Principal": "\*",

            "Action": "s3:\*",

            "Resource": [

                "arn:aws:s3:::dheerajpalvai.xyz/\*"

            ],

            "Condition": {

                "IpAddress": {

                    "aws:SourceIp": "73.119.241.21/32"

                }

            }

        }

    ]

}

* Here I want to make **whole bucket public** and **deny** particular ips

{

    "Version": "2012-10-17",

    "Id": "S3PolicyId1",

    "Statement": [

        {

            "Sid": "PublicRead",

            "Effect": "Allow",

            "Principal": "\*",

            "Action": [

                "s3:GetObject",

                "s3:GetObjectVersion"

            ],

            "Resource": [

                "arn:aws:s3:::dheerajpalvai.xyz/\*"

            ]

        },

        {

            "Sid": "IPAllow",

            "Effect": "Deny",

            "Principal": "\*",

            "Action": "s3:\*",

            "Resource": [

                "arn:aws:s3:::dheerajpalvai.xyz/\*"

            ],

            "Condition": {

                "IpAddress": {

                    "aws:SourceIp": "73.119.241.21/32"

                }

            }

        }

    ]

}

Till now we discussed about Allow or Deny of particular IP’s (internet router IP’s)

**[Summary**: Bucket make public -> Allow some ip’s ->Deny some ip’s ->public bucket and Deny some ip’s]

* In real time We don’t expose our bucket to public, until unless we present public domain data or website

**Access User Level**

* Now we see about Allow or Deny of particular Users using Access points

**First method:**

->Create bucket -> dheeraj.accesspoints

S3 -> Amazon S3 -> dheeraj.accesspoints -> create folder -> testfolder1

testfolder2

* Now I want to give access to two users

1. Go to IAM -> Users -> Add user -> User name: testuser1 -> AWS Management Console access -> Custom password
2. Go to IAM -> Users -> Add user -> User name: testuser2 -> AWS Management Console access -> Custom password

Now we need to give S3 Access to two users (testuser1, testuser2)

* We can give in two types
* Using IAM Policy
* Using S3 Bucket Policy

**Google**: aws s3 IAM policy example

## Allowing an IAM User Access to One of Your Buckets

{

"Version":"2012-10-17",

"Statement":[

{

"Effect":"Allow",

"Action": "s3:ListAllMyBuckets",

"Resource":"arn:aws:s3:::\*"

},

{

"Effect":"Allow",

"Action":["s3:ListBucket","s3:GetBucketLocation"],

"Resource":"arn:aws:s3:::awsexamplebucket1"

},

{

"Effect":"Allow",

"Action":[

"s3:PutObject",

"s3:PutObjectAcl",

"s3:GetObject",

"s3:GetObjectAcl",

"s3:DeleteObject"

],

"Resource":"arn:aws:s3:::awsexamplebucket1/\*"

}

]

}

Copydheeraj.accesspoints bucket ARN and replace in above code “Resource”

Go to -> IAM -> Policies -> Create policy -> JSON -> paste code-> Review policy -> Name: S3policyTestuser1

NOW Go to -> IAM -> Users -> testuser1-> Add permissions -> Attach existing policies directly -> S3policyTestuser1

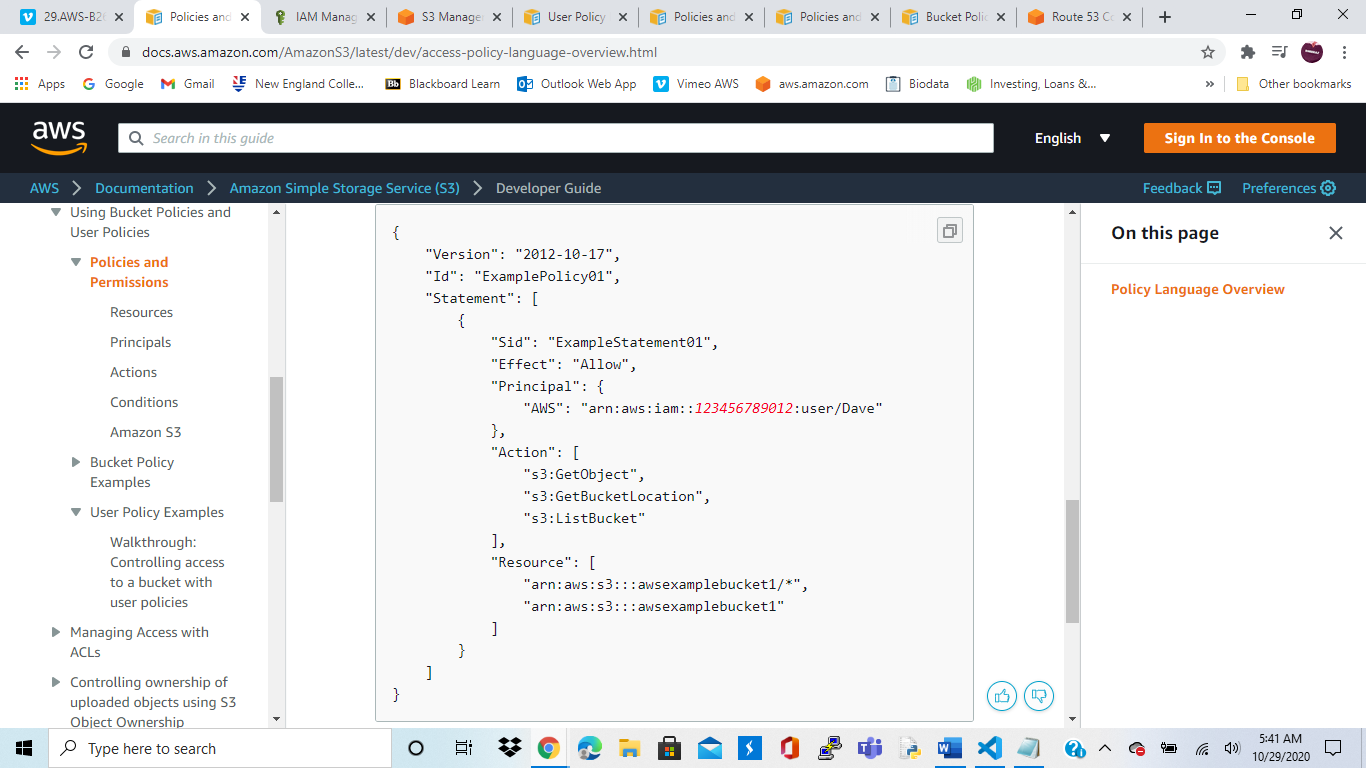
Workflow: **[Bucket ARN – IAM Policy (past in code and name it) & testuser1 – add that name]**

{Here we have to understand – to access buckets we no need to use only bucket policy we can also use IAM policies}

* Login with testuser1 & 12 digit code is ARN middle number in testuser1 -> go to S3 -> Amazon S3 -> dheeraj.accesspoints -> testfolder1 -> upload html files (we have all permissions of files)

**Second method**: Give access to individual users

**Google**: aws s3 bucket policy iam user



{

    "Version": "2012-10-17",

    "Id": "ExamplePolicy01",

    "Statement": [

        {

            "Sid": "ExampleStatement01",

            "Effect": "Allow",

            "Principal": {

                "AWS": "arn:aws:iam::683850180077:user/testuser2" <in testuser2 folder>

            },

            "Action": [

                "s3:GetObject",

                "s3:GetBucketLocation",

                "s3:ListBucket"

            ],

            "Resource": [

                "arn:aws:s3:::dheeraj.accesspoints/\*",

                "arn:aws:s3:::dheeraj.accesspoints"

            ]

        }

    ]

}

Copy this code and -> Amazon S3 -> dheeraj.accesspoints -> [Permissions] -> bucket policy -> paste

* Login with testuser2 -> S3 (shows access deny)

Because testuser2 does not have any public access

Go to main account-> IAM -> users -> testuser2-> Add inline policy (to delete with user) -> Json -> Listbuckets

S3 iam policy

{

"Version": "2012-10-17",

"Statement": [

{

"Effect": "Allow",

"Action": "s3:ListAllMyBuckets",

"Resource": "arn:aws:s3:::\*"

}

]

}

* Again, Login with testuser2 -> S3 (you can see all)
* Note: in first method policies are given in IAM policy

In second method policies are given in bucket policy and buckets list in IAM user

Workflow: **[user ARN – bucket Policy & list buckets in testuser1]**

* **In Realtime if there are more users, it will be problem to maintaining, so we go with “Access points”**

**Access points**

ii, S3 -> Amazon S3 -> dheeraj.accesspoints -> **[Access points]**

1. Create a Storage Account(bucket) and Create folder testuser1 and testuser2. Put some data in it.

Also Create two users testuser1 and testuser2 and configure profiles.

Users don’t need to have any sort of bucket policy.

2. Create a Bucket Policy as shown below. It will allow Access Points to Manage the access.

{

  "Version": "2012-10-17",

  "Statement": [

    {

      "Effect": "Allow",

      "Principal": {

        "AWS": "\*"

      },

      "Action": "\*",

      "Resource": [

        "arn:aws:s3:::dheeraj.accesspoints",

        "arn:aws:s3:::dheeraj.accesspoints/\*"

      ],

      "Condition": {

        "StringEquals": {

          "s3:DataAccessPointAccount": "683850180077"

        }

      }

    }

  ]

}

3. Create an access point policy which will allow both testuser1 and testuser2 to use access point

Amazon S3 -> dheeraj.accesspoints -> Create access point -> Access point name: awsb26usersaccesspoints -> Internet -> {write policy}

{

    "Version": "2012-10-17",

    "Statement": [

        {

            "Effect": "Allow",

            "Principal": {

                "AWS": "arn:aws:iam::683850180077:user/testuser1"

            },

            "Action": "\*",

            "Resource": "arn:aws:s3:us-east-1:683850180077:accesspoint/awsb26usersaccesspoints/object/testuser1/\*"

        },

        {

            "Effect": "Allow",

            "Principal": {

                "AWS": "arn:aws:iam::683850180077:user/testuser2"

            },

            "Action": "\*",

            "Resource": "arn:aws:s3:us-east-1:683850180077:accesspoint/awsb26usersaccesspoints/object/testuser2/\*"

        }

    ]

}

* IAM -> Users -> testuser1 -> Security credentials -> Create access key

AKIAZ6OFFUXWR76LVKRB

nqMZCaiVMAdgXGXhqma2wSEevEspDFoU94AepfGI

* Now we no need to list buckets in IAM policy, just access Endpoints
* So, we detach policies -> IAM -> Users -> testuser1 -> Permissions -> (detach policies)
* IAM -> Users -> testuser2 -> Security credentials -> Create access key

AKIAZ6OFFUXWQANWPPUY

kyXEUJPonjqc9xFotB1a/aoqztEHNh8SWE7+tZuw

IAM -> Users -> testuser2 -> Permissions -> (detach policies)

**Install: aws cli**

**cd /users/dheer/.aws ->**

CMD (administrator): aws configure

AWS Access Key ID [None]: testuser1 accesskey

AWS Secret Access Key [None]: testuser1 Secretkey

Default region name [None]: us-east-1

Default output format [None]: json

* Now if we give access key & secretkey of testuser2, it will override testuser1, so
* **C:\Users\dheer\.aws>dir**
* CMD: notepad.exe credentials

It opens “credentials” file in notepad with testuser1 credentials like this

[default]

aws\_access\_key\_id = AKIAZ6OFFUXWR76LVKRB

aws\_secret\_access\_key = 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

Add testuser2 credentials also

[testuser1]

aws\_access\_key\_id = AKIAZ6OFFUXWR76LVKRB

aws\_secret\_access\_key = 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

[testuser2]

aws\_access\_key\_id = AKIAZ6OFFUXWQANWPPUY

aws\_secret\_access\_key = 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

[Now paths of testuser1 and testuser2 are configured]

4. Upload data using access point with testuser1 & testuser2.

Go inside created accesspoint and copy: (Amazon S3 -> Buckets -> dheeraj.accesspoints(bucket name) -> Access points -> awsb26usersaccesspoints(Accesspoint name) ->)

arn:aws:s3:us-east-1:683850180077:accesspoint/awsb26usersaccesspoints

* In CMD go to “stack-policy.json” file folder (a sample file to upload into testuser1)

C:\Users\dheer\.aws> cd C:\Users\dheer\Desktop\AWS Training\AWS\_LAB\_Files\CloudFormation

C:\Users\dheer\Desktop\AWS Training\AWS\_LAB\_Files\CloudFormation>

aws s3 cp stack-policy.json s3://arn:aws:s3:us-east-1:683850180077:accesspoint/awsb26usersaccesspoints/testuser1/stack-policy.json --profile testuser1

**[coping** stack-policy.json to s3 -> bucket -> accesspoint-> testuser1 folder and assign “stack-policy.json” profile of IAM user: testuser1**]**

**Result is:**

**upload: .\stack-policy.json to s3://arn:aws:s3:us-east-1:683850180077:accesspoint/awsb26usersaccesspoints/testuser1/stack-policy.json**

**same upload stack-policy.json file in accesspoints testuser2**

**now s3 bucket is accessed by IAM->users in their own accounts**

**----------------------------**

**Q, I want to share one bucket to multiple users, what is best practice?**

**A: We have “s3 bucket policy” but instead of that we use “Accesspoint” policy, we just add user in Accesspoint policy**

iii,S3 -> Amazon S3 -> dheerajpalvai.xyz -> **[Management]** -> Lifecycle

-> Replication

**Lifecycle**: the purpose of creating Lifecycle is, if we want move files in Standard-IA storage after 30 days to One Zone-IA storage so files be here for 60 days and deleted automatically, for that we create a rule

**Replication**: copying the data form one bucket to another bucket within region or in different region

This is Asynchronies not synchronous

Asynchronies means it will not wait, as soon as we put data in one bucket it replicates in another bucket

Synchronous means it will ask when to go data from one bucket it replicates in another bucket