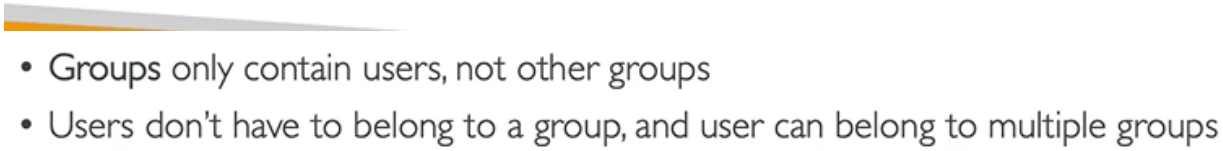
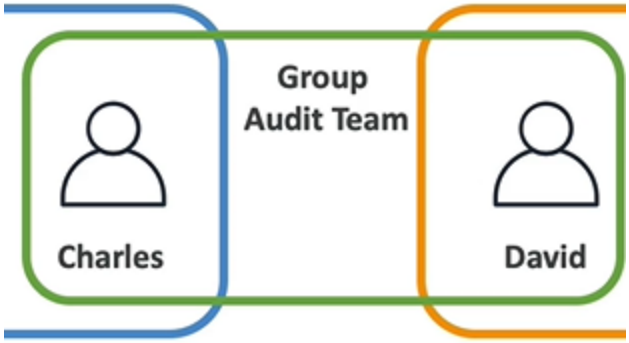
IAM: Users & Groups

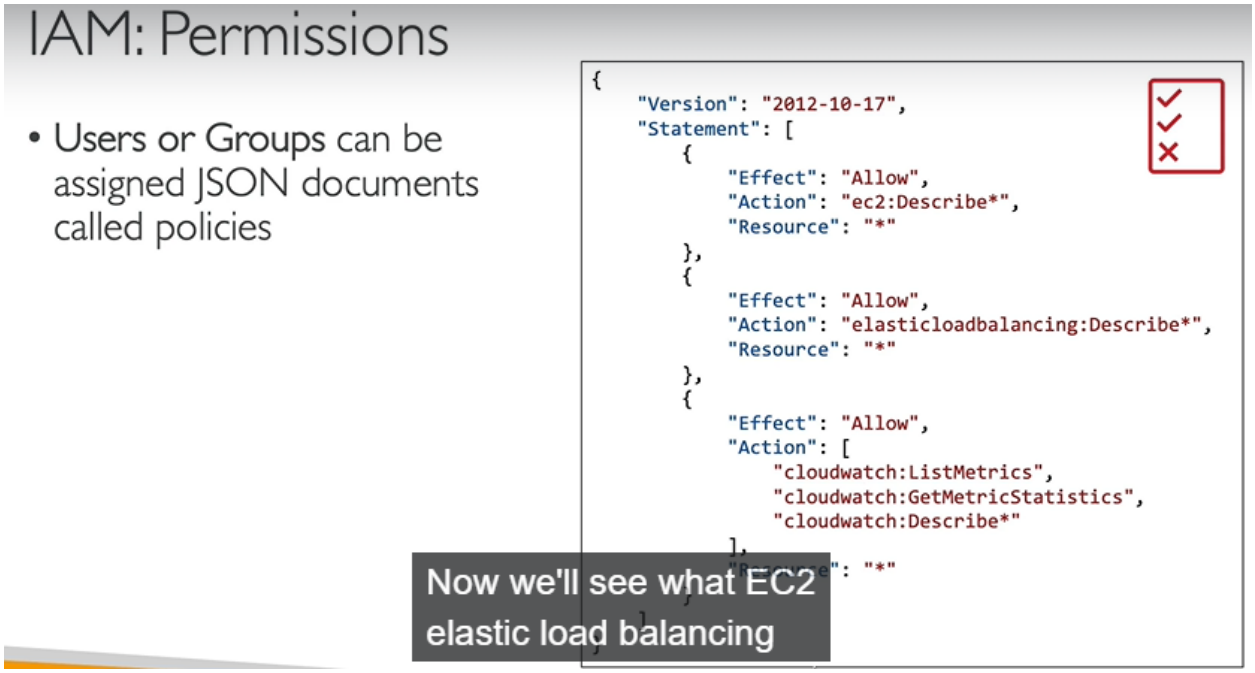
1. IAM stands for Identity and Access Management and it is a global service..in IAM we r going to create users and assign them groups
2. Root Account should not be used or shared..
3. What you should be doing instead, is create users.
4. So you will create users in IAM,

5.
6. We have created 6 people and all of them are in our organisation and we have created groups based on their roles

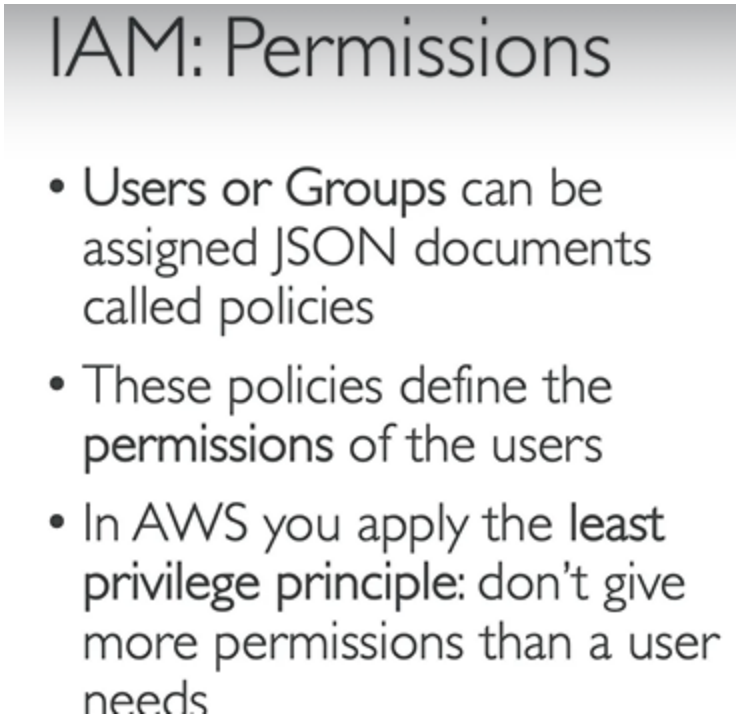
7. 

8.
9. A user can work for multiple groups ..

10. 

11. IAM Permission: This is just describing in, I think plain English,
12. what a user is allowed to do or what a group
13. and all the users within that group are allowed to do. It comes in the JSON documents called policies
14. 

15. Here in the picture we can see …that we have allowed users to use ec2,elasticloadbalancing and cloudwatch
16. It help us to give permissions to users
17. We have a principle called least privilege principle



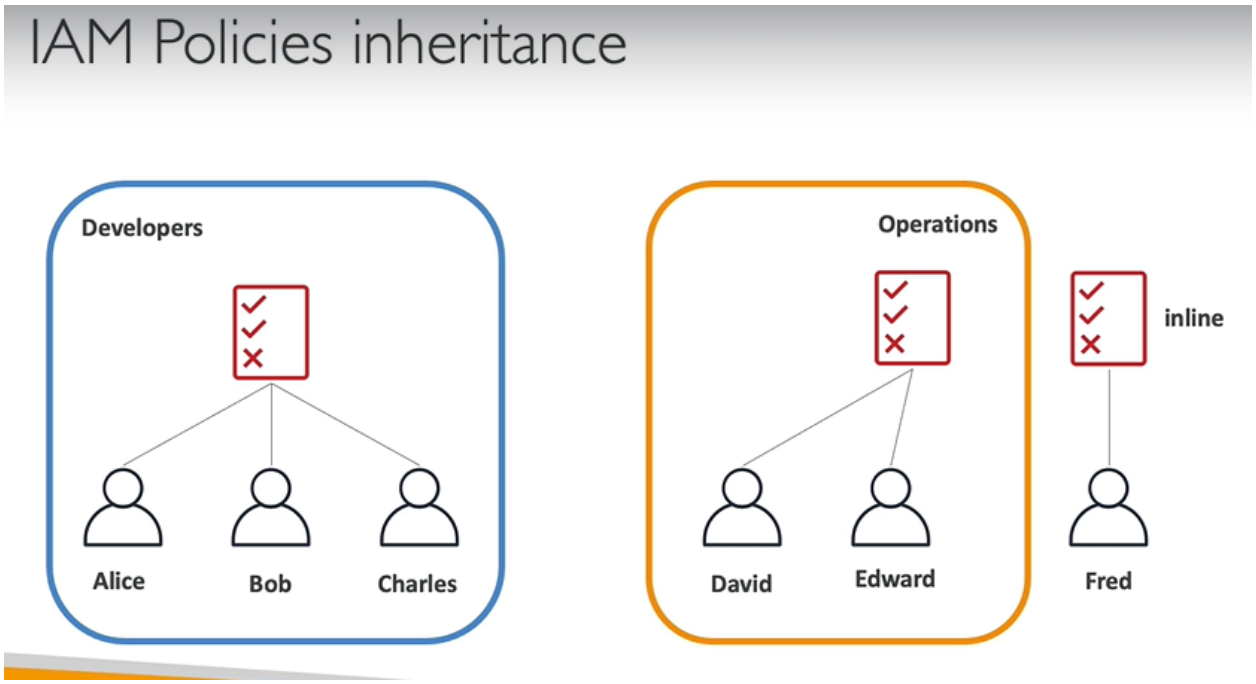
18.
19. Now if a user wants 3 services … We just create permission for that user

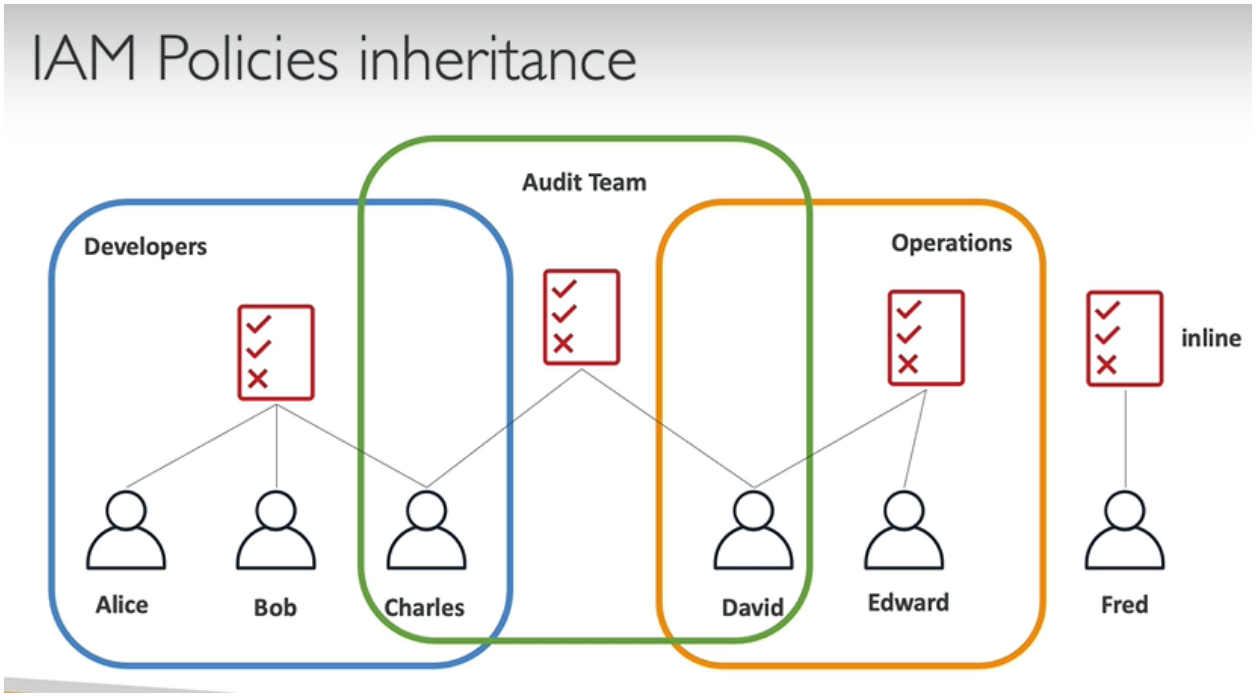
IAM hands on:

1. As the root user have many permission..we create admin users and regular users
2. To create an user we can go to IAM -> users
3. After that we have create a group called admin and gave administrative policy permission
4. Every permission we assign to group..it can be assigned to user
5. After that we have created Account alias..as account number will be hard to remember
6. After that we will login as IAM user

IAM Policies:

1. Every group has their own set of policies…like developer group can have some policies which are diffrent from Operations group
2. If Fred is a user, it has the possibility not to belong to a group. And we have the possibility to create what's called an inline policy which has a policy that's only attached to a user.

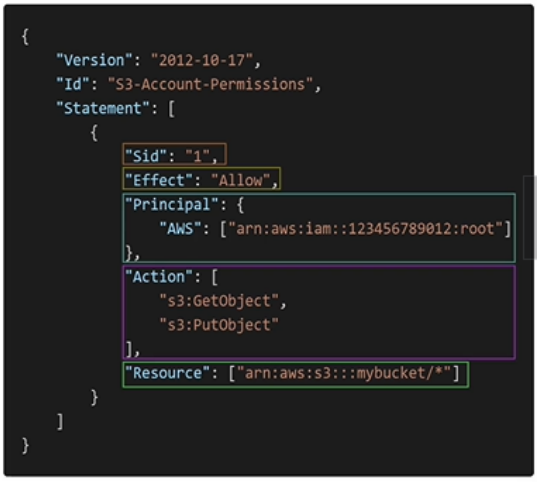
3. 

4. And if charles and david are in multiple group..like for example here they both are in audit group..now they have their own set of policies
5. 

```
{
    "Version": "2012-10-17",
    "Id": "S3-Account-Permissions",
    "Statement": [
        {
            "Sid": "1",
            "Effect": "Allow",
            "Principal": {
                "AWS": ["arn:aws:iam::123456789012:root"]
            },
            "Action": [
                "s3:GetObject",
                "s3:PutObject"
            ],
            "Resource": ["arn:aws:s3:::mybucket/*"]
        }
    ]
}
```

6. IAM Policies Structure
7. It consists of version policy language version ..we'll always inckude "2012-10-17"

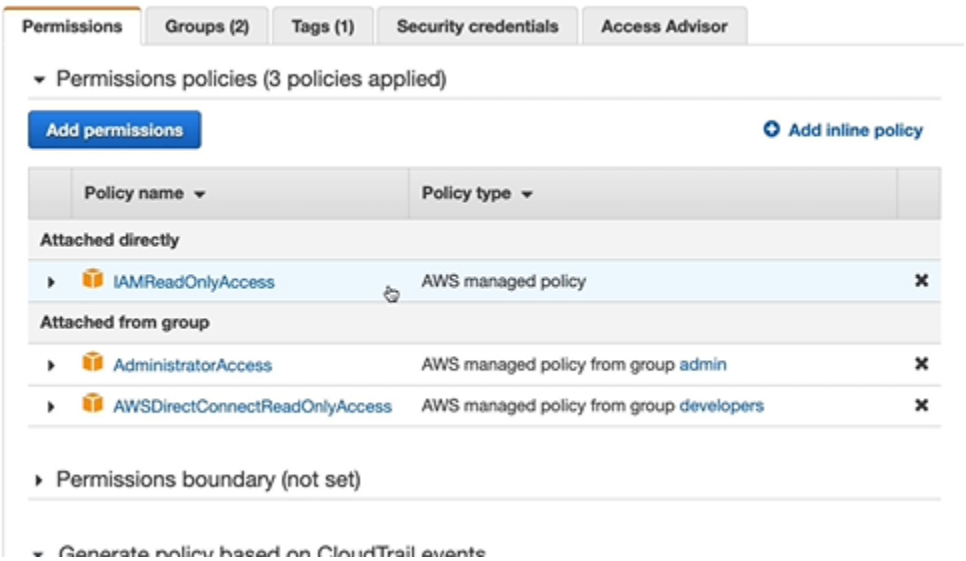
- Consists of
  - Version: policy language version, always include "2012-10-17"
  - Id: an identifier for the policy (optional)
  - Statement: one or more individual statements (required)
- Statements consists of
  - Sid: an identifier for the statement (optional)
  - Effect: whether the statement allows or denies access (Allow, Deny)
  - Principal: account/user/role to which this policy applied to
  - Action: list of actions this policy allows or denies
  - Resource: list of resources to which the actions applied to
  - Condition: conditions for when this policy is in effect (optional)

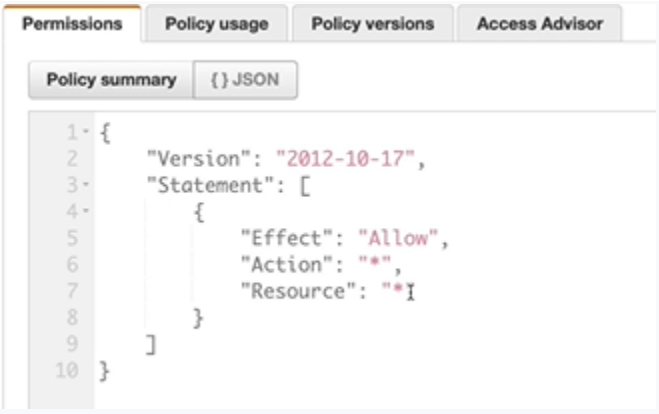


8.
9. Here in statement we have effects…it allows or denies access to certain API's
10. In principle here we are applying our policies to the root user

IAM Hands On:

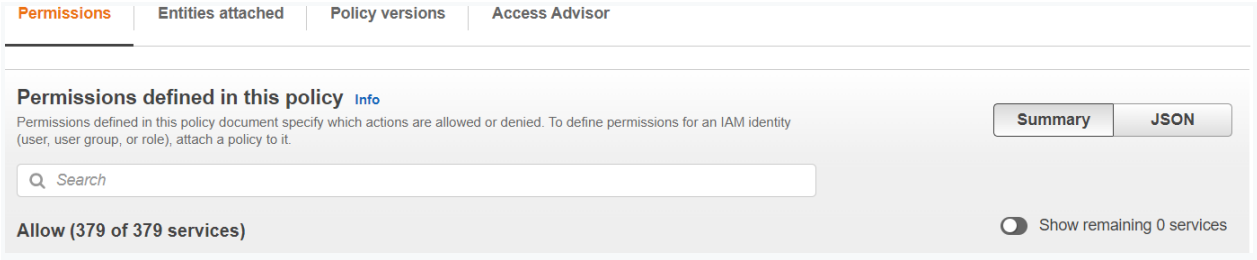
1. Here like if a user is removed from the group..The policies of the user will also be removed

2. We can also direct attach some policies to the user ..we have attache dIAMReadOnlyAccess to the user directly
3. Later we have added him again to admin group as the admin group has Administrative Access policy user will also get that
4. We have added same user to the developer group and gave AWSDIrectConnectReadOnlyAccess
5. 

6. We can see the policieds got inherited from different ways through the IAM permissions
7. If we check policies tab…here we have all the policies of IAM
8. Here we can see administrativeAccess policy…
9. 

Here we have Action:"*", Resources:"*".. Sp we allows all the actions from all the resources

10. We have a total of

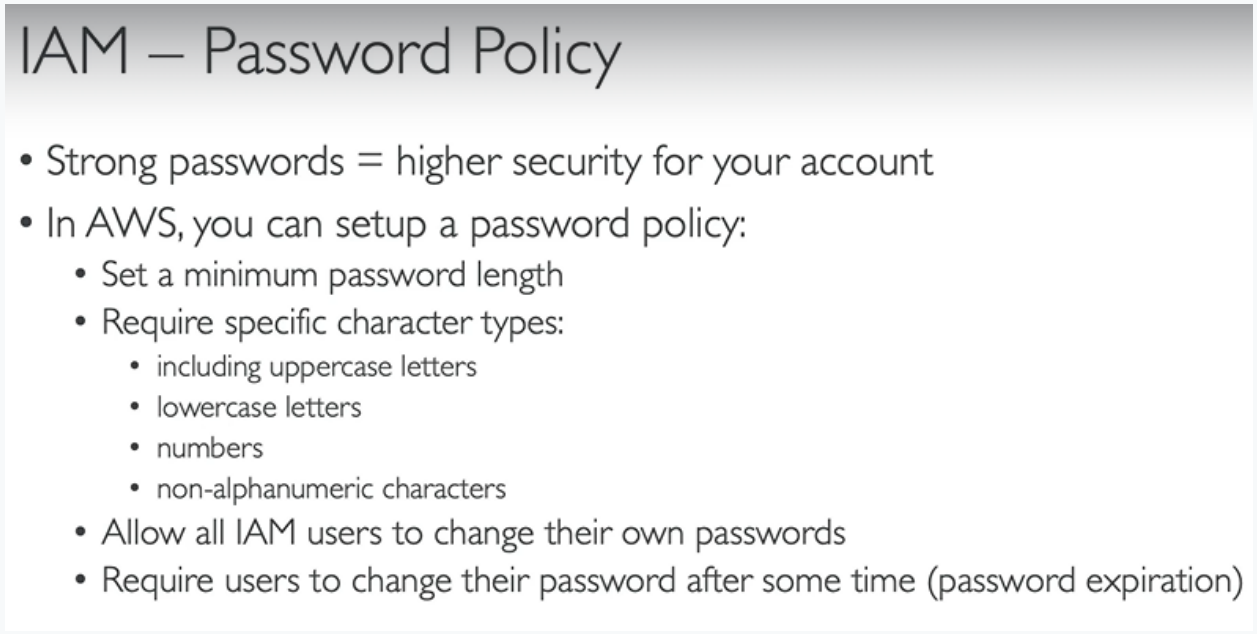


services to avail

11. We can also create our own policies

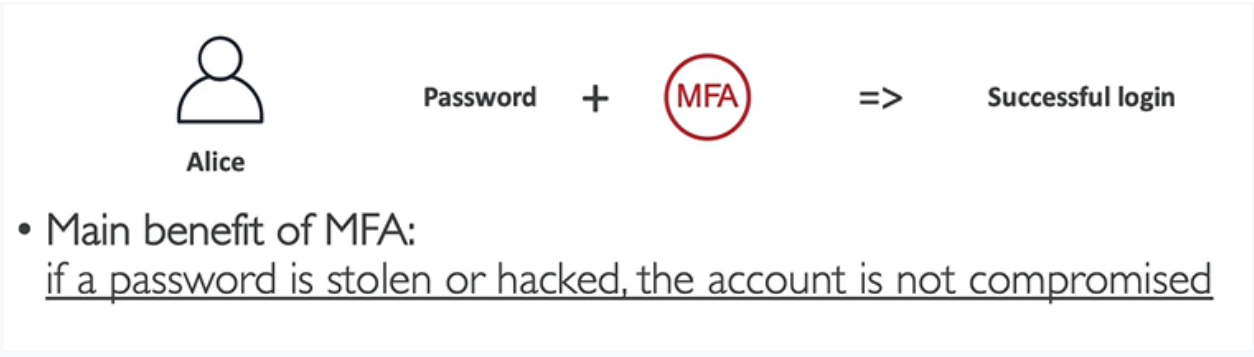
IAM MFA Overview:

1. We can define password policy to protect our users and groups form being compromised
2. We can give some set of rules for password and Allow Users to change their passwords
3. Or We can say change your password for every 90days
4. Also we can prevent re-use of same password



5.
6. Multi Factor Authentication:

7. 

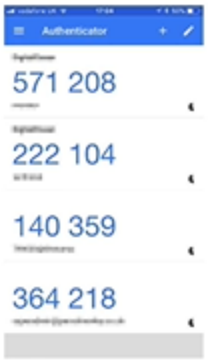
8. 

9. To enable MFA we have diff options in AWS
10. We can use google authenticator in mobiles or by using authy(it works on both phones and lappy)
11. In authy we can get multiple tokens(for root user,IAM user etc) on a single device

12.

MFA devices options in AWS

Virtual MFA device



Google Authenticator
(phone only)

Authy
(multi-device)

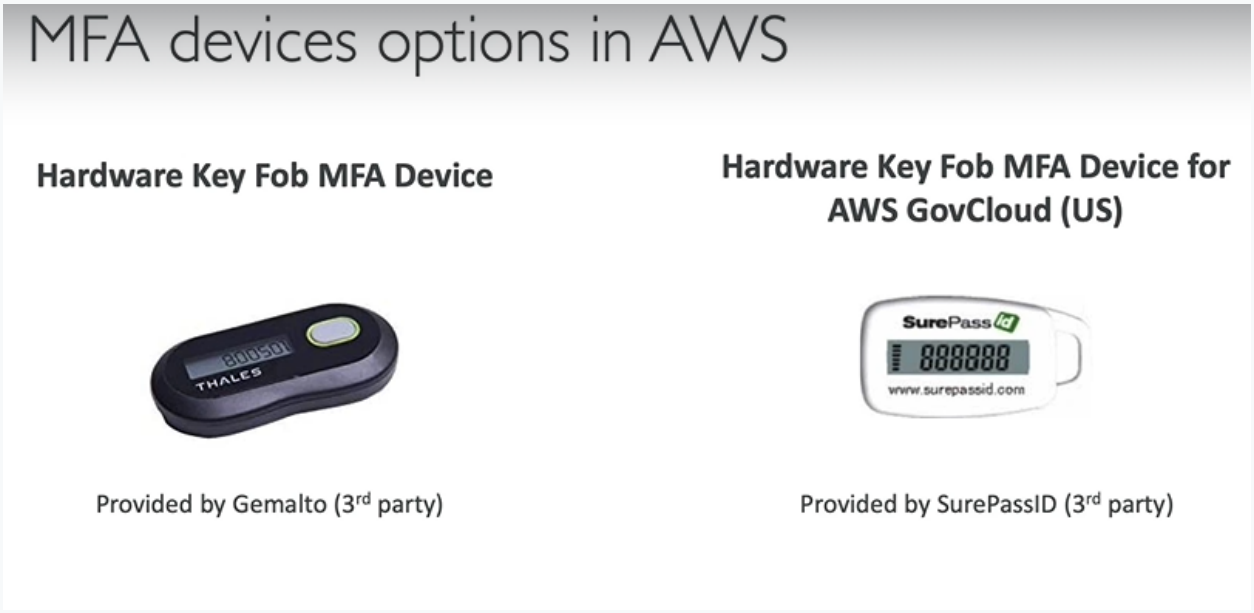
Support for multiple tokens on a single device.

13.

Universal 2nd Factor (U2F) Security Key

YubiKey by Yubico (3rd party)

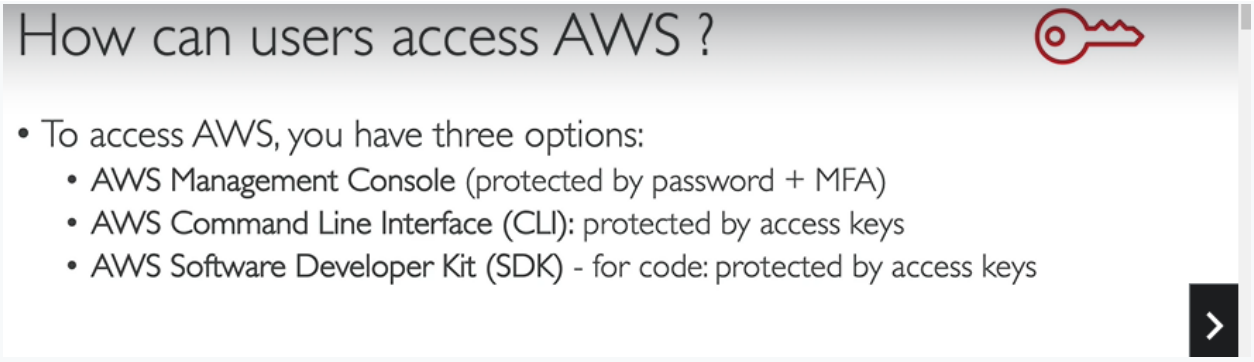
Support for multiple root and IAM users using a single security key

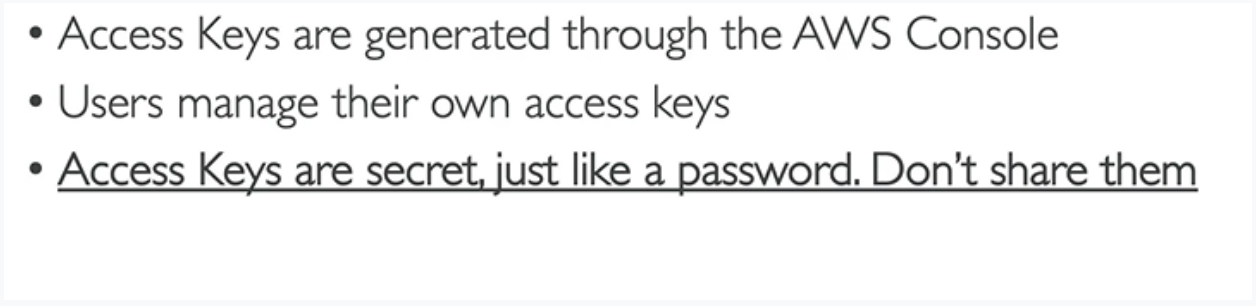
14. 

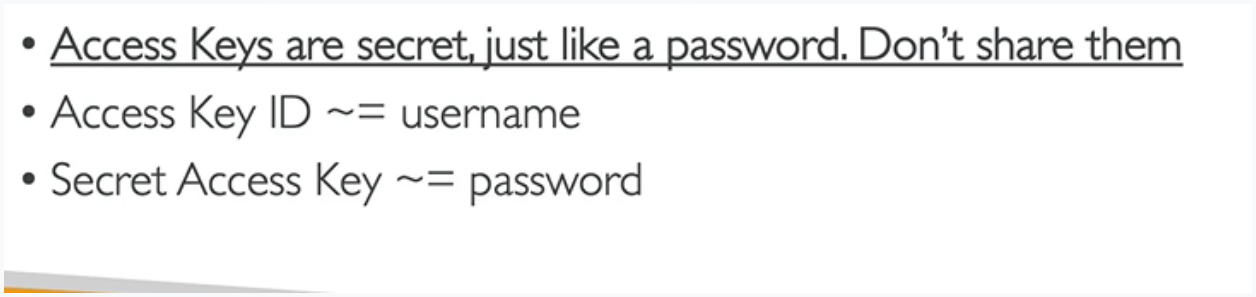
IAM MFA Hands On:

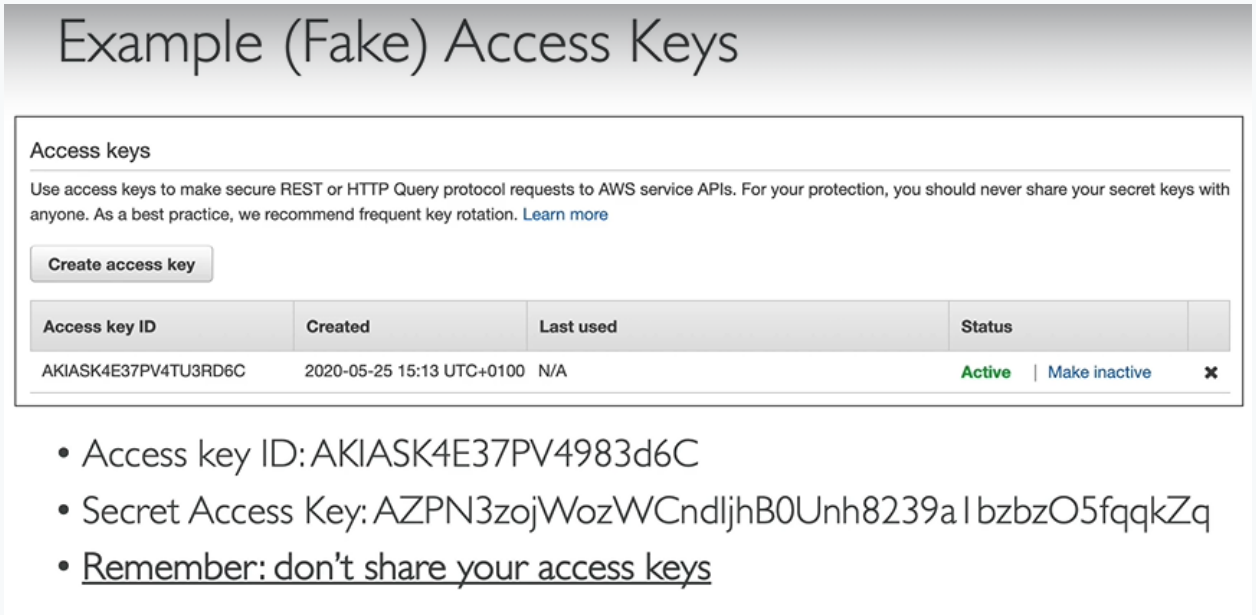
1. Just see online..it is just adding 2 factor authentication

AWS Access Keys, CLI and SDK

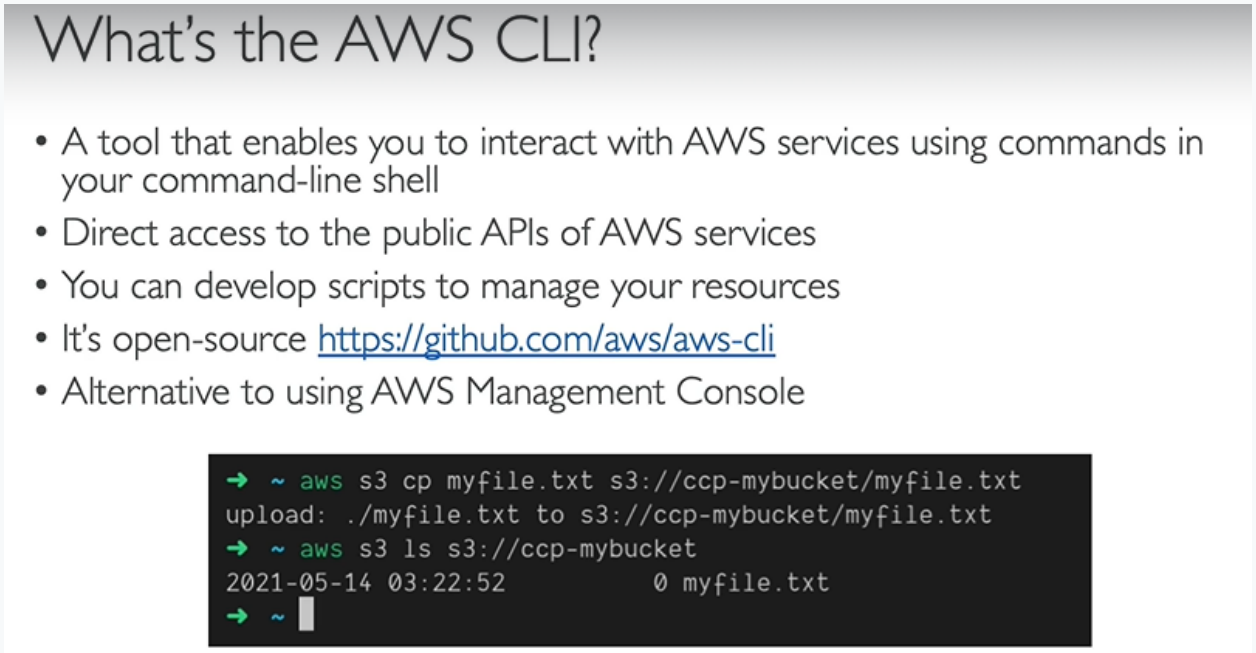
1. We have 3 ways to Access AWS
2. 

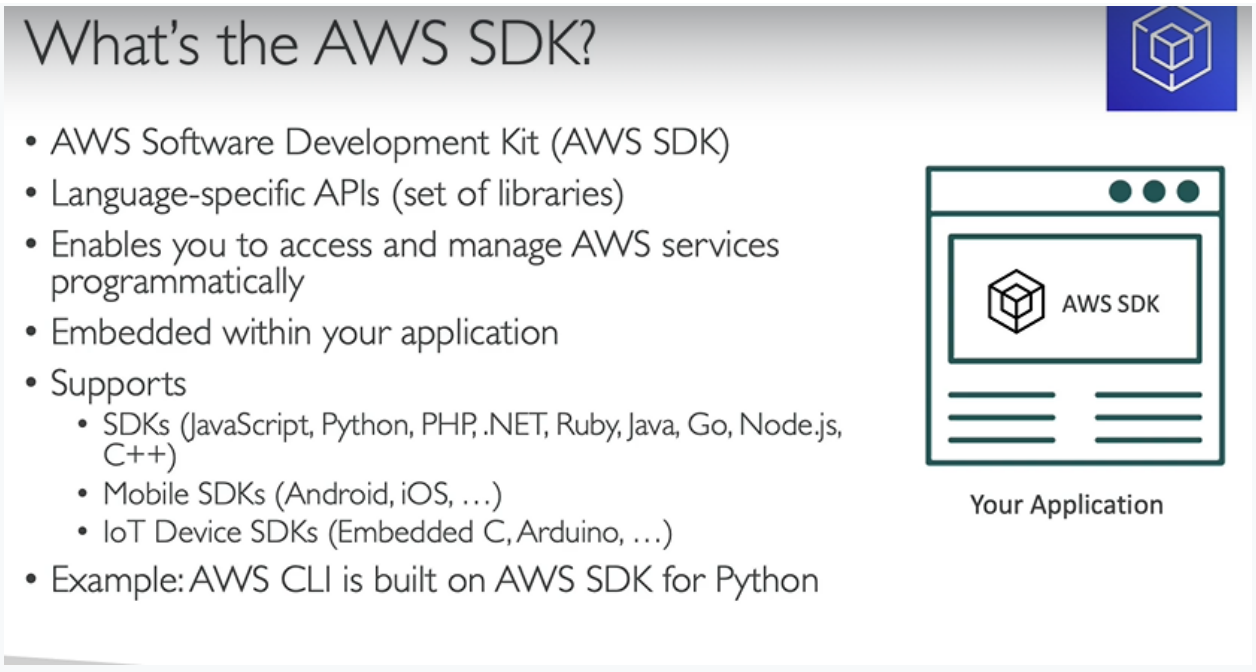
3. AWS CLI is a command line interface which is protected by access keys and access key are credentials which will allow us to access AWS from our terminal
4. AWS SDK which is used to call API's from AWS from within our application code.
5. 

6. Users should manage their own Access keys
7. 

8. 

9. With this Access key id and secret access key …these when loaded into CLI will allow me to Access the AWS API

10. 

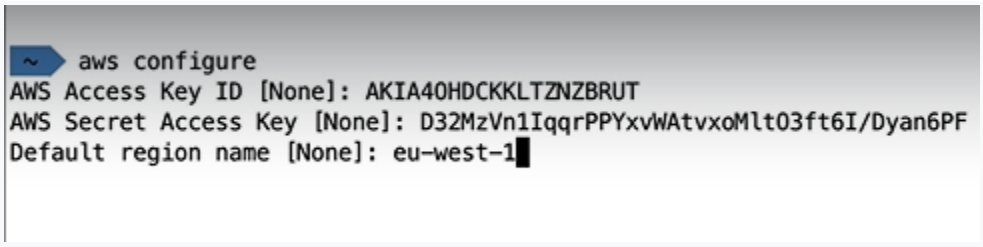
11. 

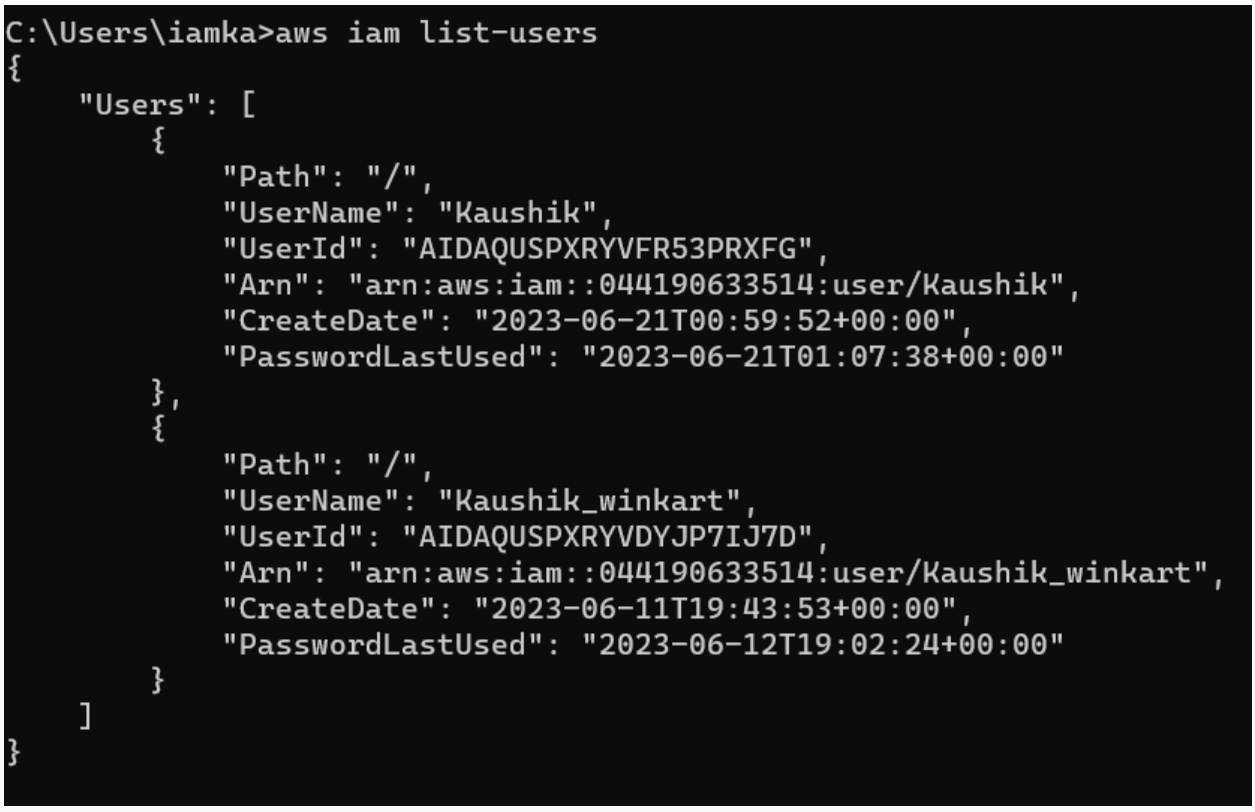
12. We have to integrate AWS SDK inside our application

AWS CLI on Windows:

1. Just follow instructions

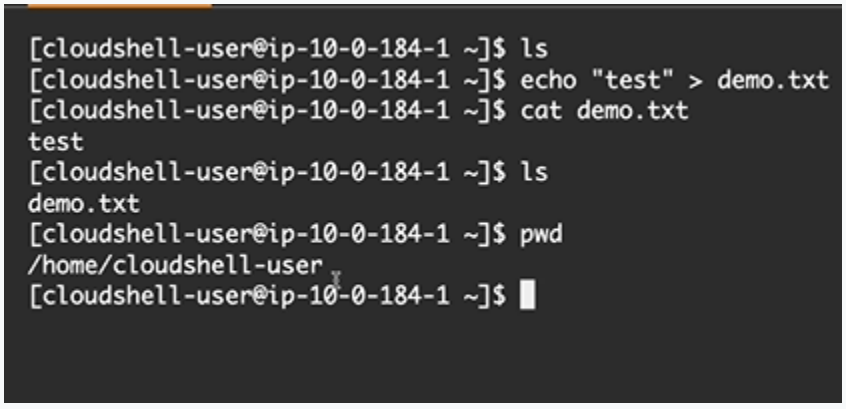
AWS CLI Hands on:

1. We can create access key ..by going to users->kaushik->security credentials->..scroll down… create access key
2. After generating aws access key for CLI..open cmd and type
3. 

4. 

5. If we type aws iam list-users we get all the information same as console
6. Even if we change anything in aws console..CLI will reflect same

AWS CloudShell:

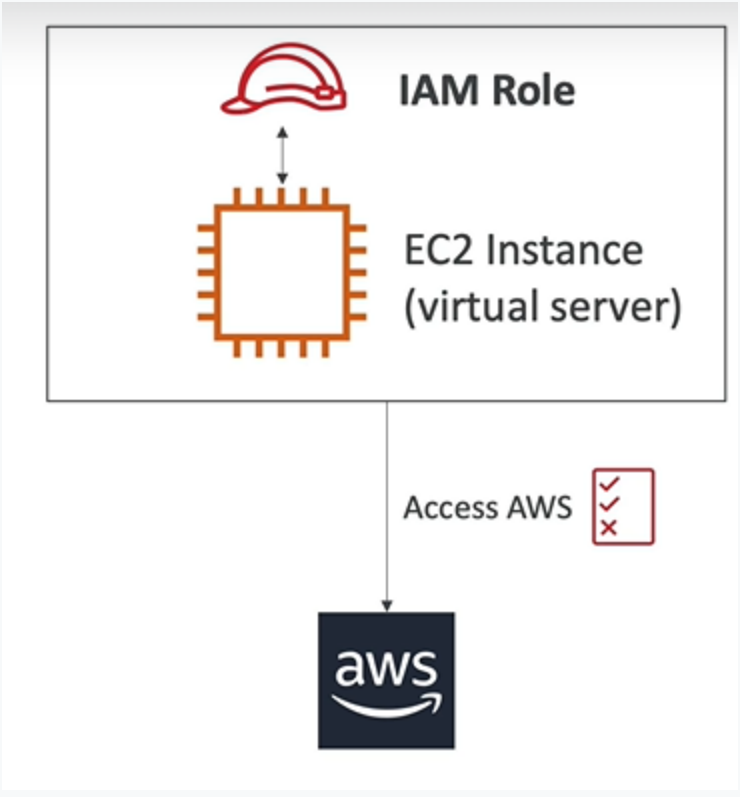
1. Cloud shell is nothing but terminal on cloud
2. In cloudshell pwd gives full path to file in aws
3. 

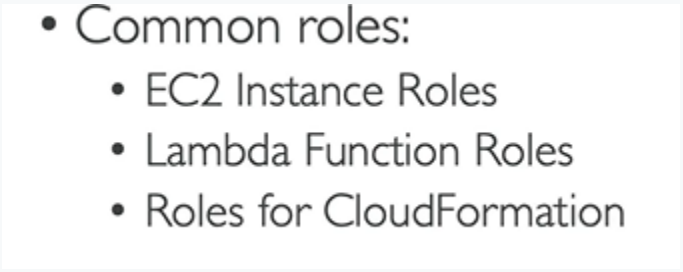
4. We can either use cloudshell or CLI…but prefer cloudshell bcuz it has more options

IAM Roles:

1. 

2. IAM roles are similar to users…they will need some kind of permissions..so we need to assign them
3. So these IAM role will be just like a user, but they are intended to be used not by physical people, but instead they will be used by AWS services.

4. 

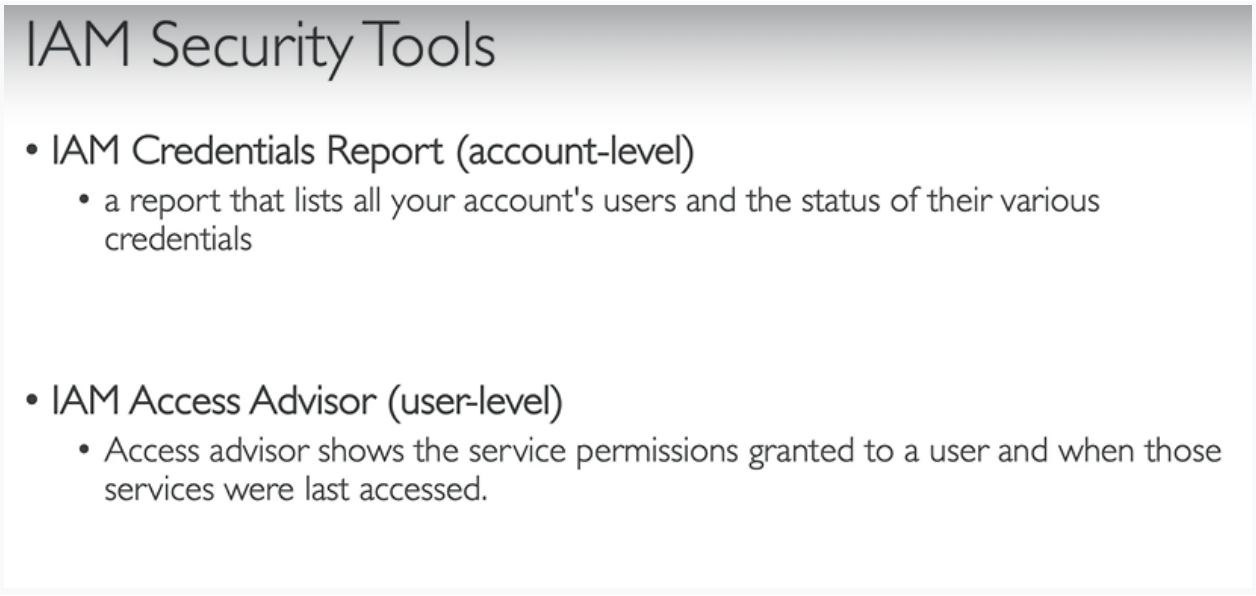
5. Here when we create an EC2 instance..it might need some permission to access AWS..so to avail them it uses IAM Role
6. 

IAM Roles Hands On:

1. To create roles..we go to roles section and click create role
2. And click on AWS servies and everything is out of syllabus ..then choose Ec2
3. Then we give policy permission and a name to the role

IAM security Tools:

1. So we can create an IAM Credentials Report
2. and this is at your account-level, This report will contain all your accounts users and the status of their various credentials.
3. We can also create IAM Access Advisor which is on User level

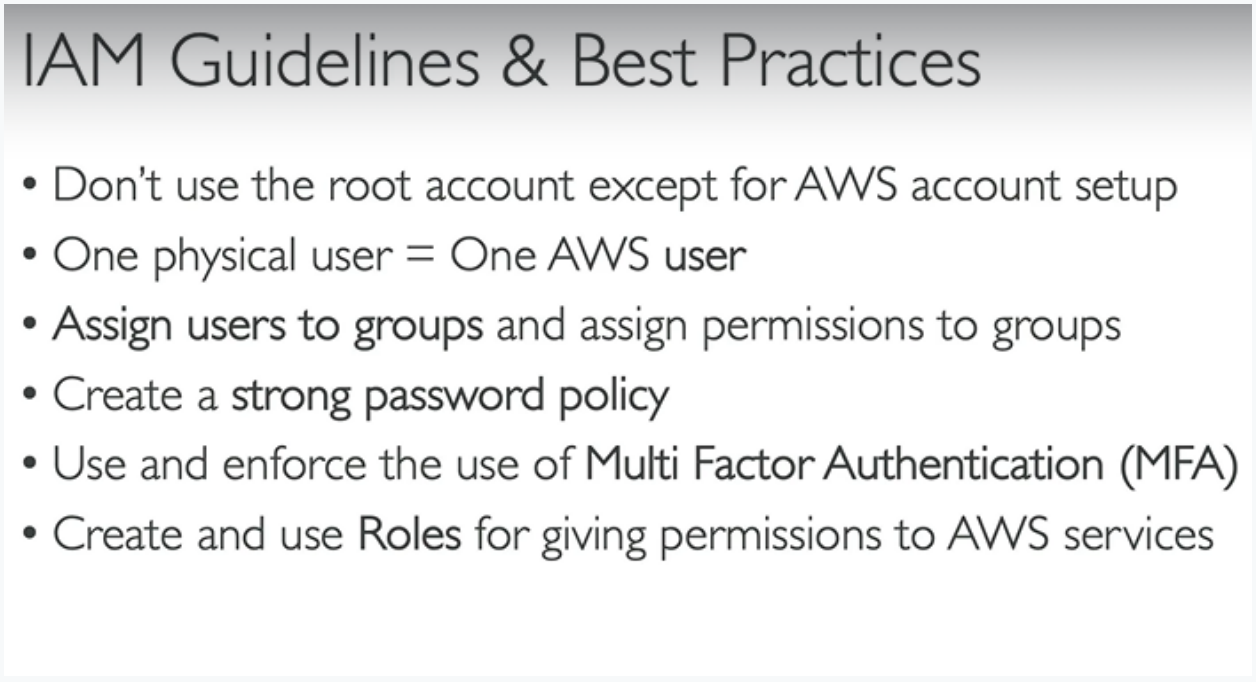


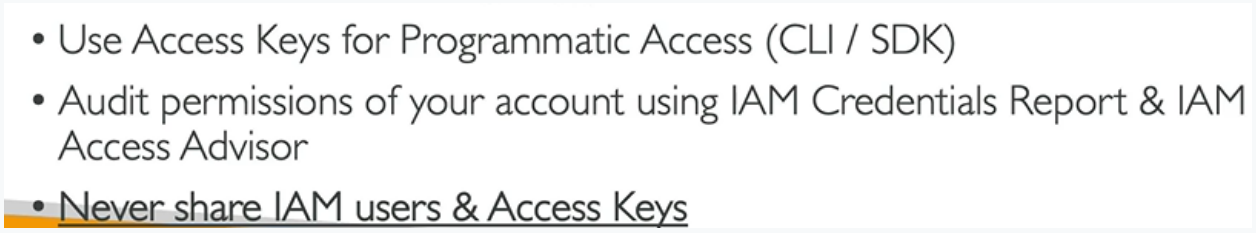
4.

IAM security Hands on:

1. On bottom left we have credential report and we can download it
2. We can access “Access Advisor” by going to user profile and click on it

IAM guidelines and best practices

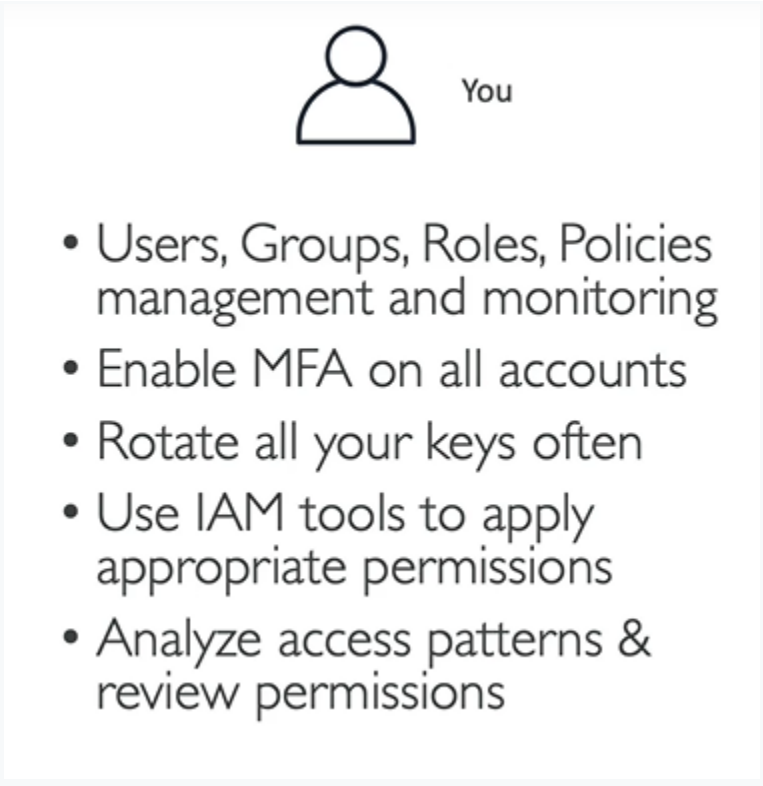
1. 

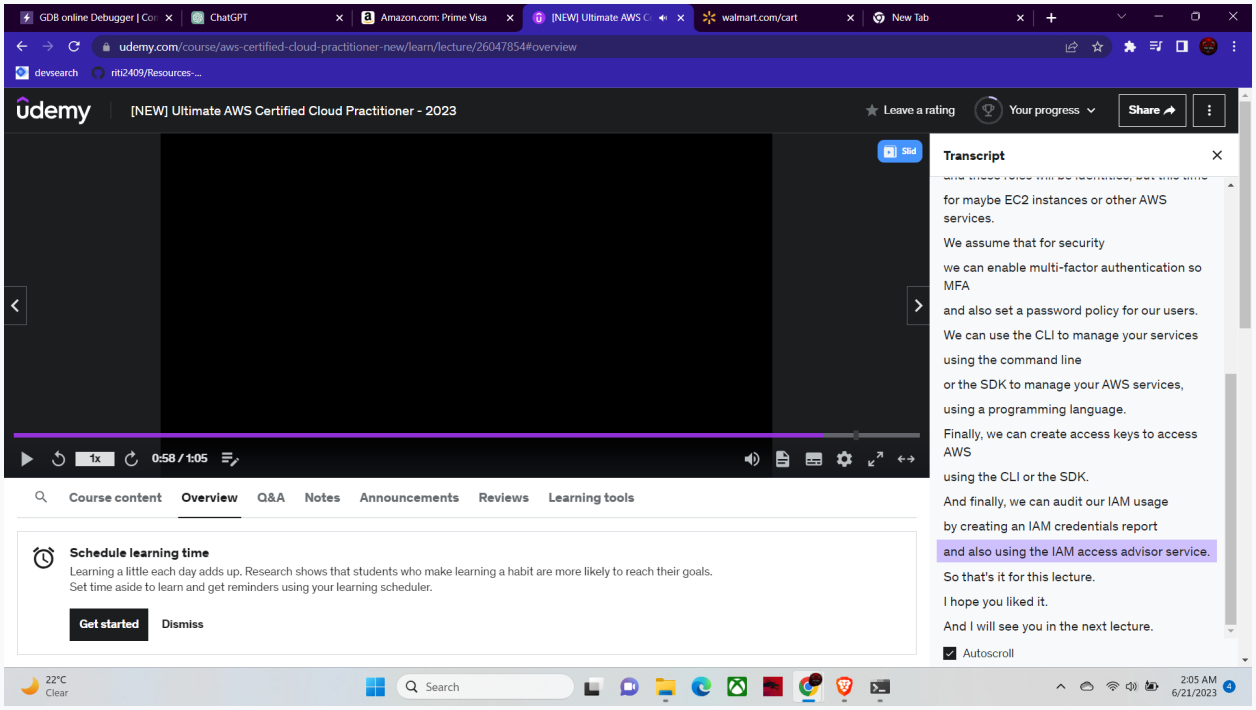
2. 

Shared Responsibility Model for IAM

1. Aws is responsible for everything they do like infrastructure , configuration and vulnerability analysis
2. So AWS is responsible for everything that they do, for example, their infrastructure
3. and their global network security, the configuration and vulnerabilty analysis of the services they offer, and any sign of compliance that they are responsible for.
4. You are responsible for users,...

5. 
Shared Responsibility Model for IAM
aws
You
• Infrastructure (global network security)
• Configuration and vulnerability analysis
• Compliance validation
• Users, Groups, Roles, Policies management and monitoring
• Enable MFA on all accounts

6. 
You
• Users, Groups, Roles, Policies management and monitoring
• Enable MFA on all accounts
• Rotate all your keys often
• Use IAM tools to apply appropriate permissions
• Analyze access patterns & review permissions

7. 

IAM Section Summary:

1.