QS Cloud Training

4. Permissions Perfect

3rd August 2021

Please type any questions you have in the chat. I might not be able to hear you haha.

Reading materials

- Udemy course Section 4
- Section 20 of AWS Fundamentals Learning Path
- Udemy course Section 23 and 24 (Advanced)

Agenda

What you will learn

Users

Groups

Policies

Roles

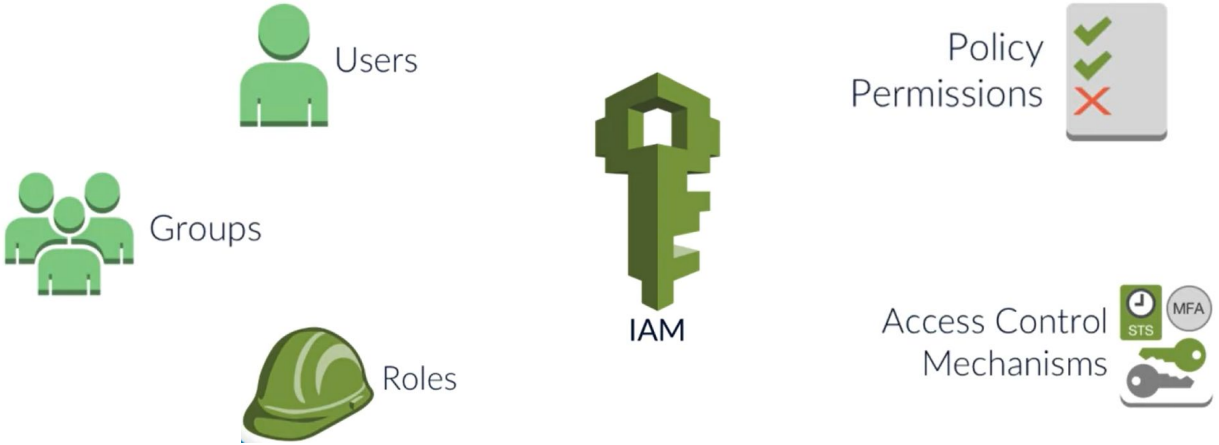
IAM Security Tools

What to expect today

As with networking, IAM can get very complicated. I will only touch on key concepts. There will be some point and click demo - please feel free to follow along.

Identity and Access Management (IAM)

- A web service that helps you securely control access to AWS resources.
- Control who is **authenticated** (signed in) and **authorized** (has permissions) to use resources.
- **Global** service



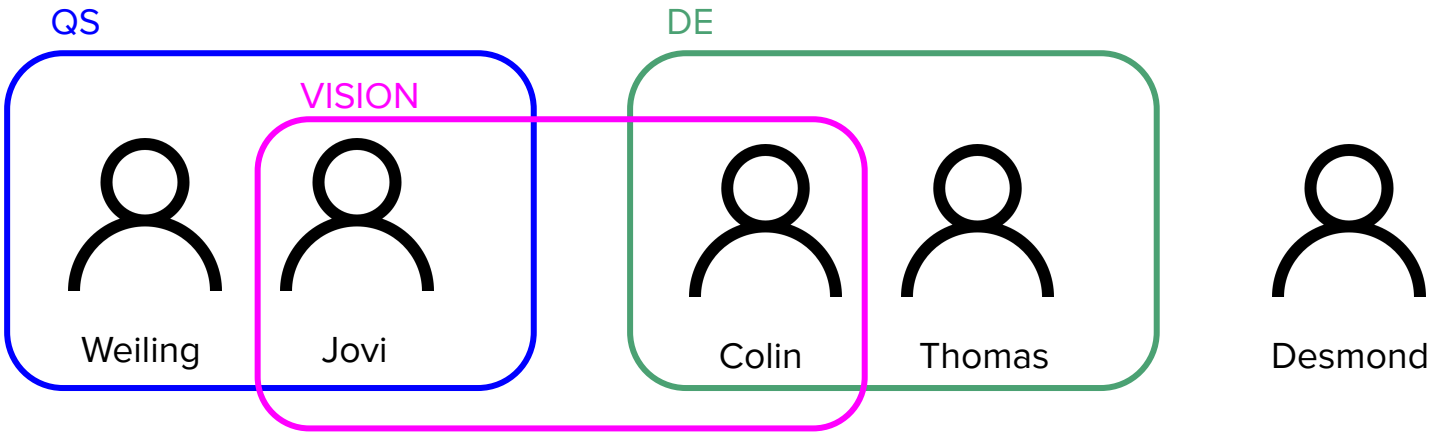
Demo: Creating users, groups, policies

On root account

- When you first create an AWS account, root account created by default.
- With **root user credentials** (email address/password), you have complete, unrestricted access to all resources in AWS account.
- Only service control policies (SCPs) in organizations can restrict permissions granted to root user.
- AWS recommends to avoid using root user credentials for everyday access.
- Instead, use IAM to create and manage identity and permissions.

IAM Users and Groups

- Users are people or systems within your organization
  - IAM accounts are **not** separate accounts, they are users within your account.
  - Users can be grouped.
- Groups only contain users, not other groups.
- Users don’t have to belong to a group, and user can belong to multiple groups.



IAM groups and policies

- IAM groups allow you to specify permissions for multiple users.
- How to assign permissions to users or groups? Use **policies**.
- **Policies** are JSON documents that define permissions of users.

PutBucketPolicy

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": "s3:PutBucketPolicy",
            "Resource": "*"
        }
    ]
}
```

IAM Policies: Version

Version:

- The policy language version

Statement:

- The main elements of the policy, which includes:
  - Sid
  - Effect
  - Action
  - Resource
  - Conditions

PutBucketPolicy

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": "s3:PutBucketPolicy",
            "Resource": "*"
        }
    ]
}
```

IAM Policies: Structure

Sid: Statement ID, unique identifier within Statement array;

PutBucketPolicy

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": "s3:PutBucketPolicy",
            "Resource": "*"
        }
    ]
}
```

IAM Policies: Structure

Effect: Either set to grant or restrict access for the defined Action.

Access to resources denied by default.

```
"Effect": "Allow",
```

```
"Effect": "Deny",
```

PutBucketPolicy

IAM Policies: Structure

Action: Action that will either be allowed or denied.

Actions are API calls for different services.

```
"Action": [
        "cloudtrail:CreateTrail",
        "cloudtrail:DeleteTrail"
],
```

PutBucketPolicy

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": "s3:PutBucketPolicy",
            "Resource": "*"
        }
    ]
}
```

IAM Policies: Structure

Resource: Specifies actual resource you wish the *Action* and *Effect* to be applied to.

Use ARN to specify resources.

```
"Resource": "arn:aws:s3:::iam-course-ca",
```

PutBucketPolicy

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": "s3:PutBucketPolicy",
            "Resource": "*"
        }
    ]
}
```

Multiple SIDs, each granting a different level of access

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Stmt1494509737040",
      "Action": "cloudtrail:*",
      "Effect": "Allow",
      "Resource": "*",
      "Condition": {
        "IpAddress": {
          "aws:SourceIp": "10.10.0.0/16"
        }
      }
    },
    {
      "Sid": "Stmt1494512658702",
      "Action": "autoscaling:*",
      "Effect": "Allow",
      "Resource": "*"
    },
    {
      "Sid": "Stmt1494515449405",
      "Action": [
        "s3:CreateBucket",
        "s3:DeleteBucket"
      ],
      "Effect": "Allow",
      "Resource": "arn:aws:s3:::iam-course-ca"
    }
  ]
}
```

From Cloud Academy

What happens if there are conflicting permissions assigned to same User?

By default, all access is denied

From Cloud Academy

What happens if there are conflicting permissions assigned to same User?

By default, all access is denied

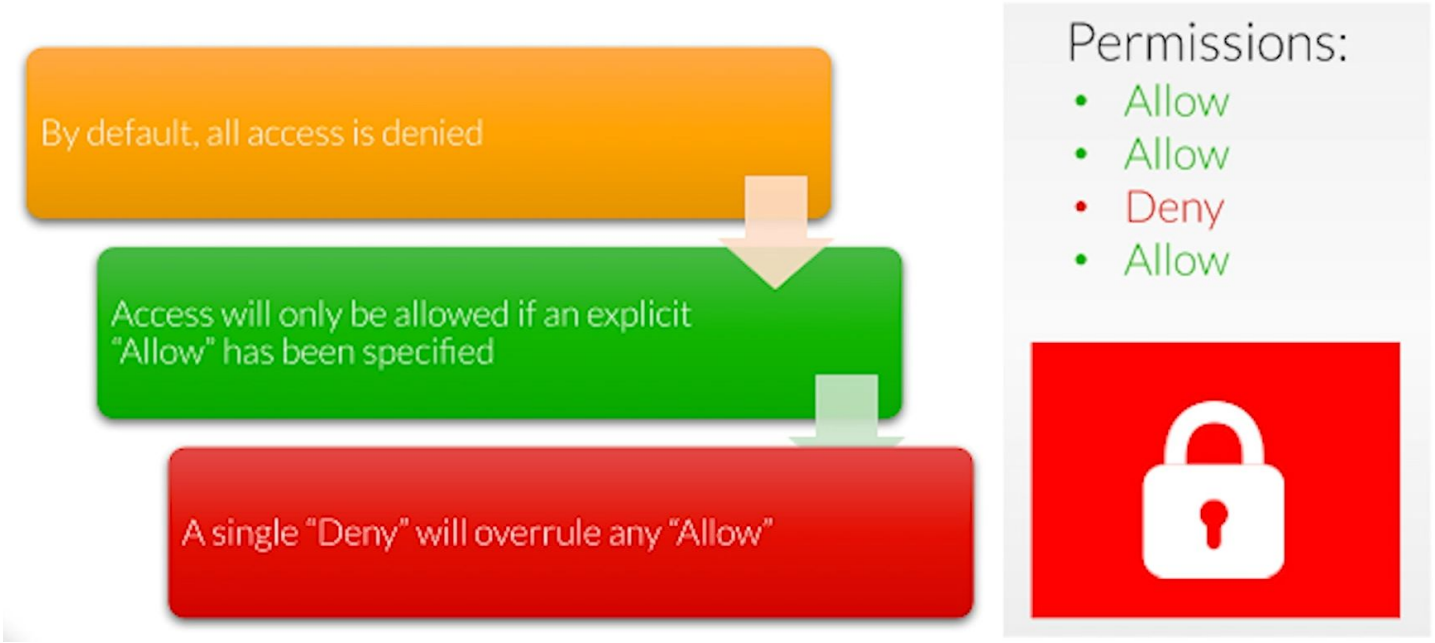
Access will only be allowed if an explicit "Allow" has been specified

Permissions:

- Allow

From Cloud Academy

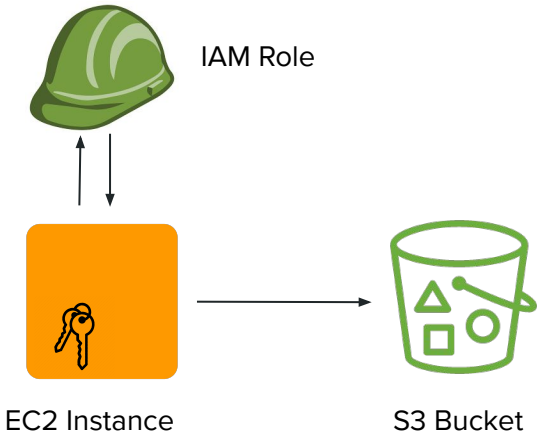
What happens if there are conflicting permissions assigned to same User?



Demo: AWS Service Role

IAM Roles

- An IAM identity with specific permissions. Similar to IAM user, except that it is **not** associated with one person. Instead, it is *assumable* by anyone who needs it.
- Provides temporary security credentials for role session.



Demo: Roles for Cross-Account Access

Roles for Cross-Account Access

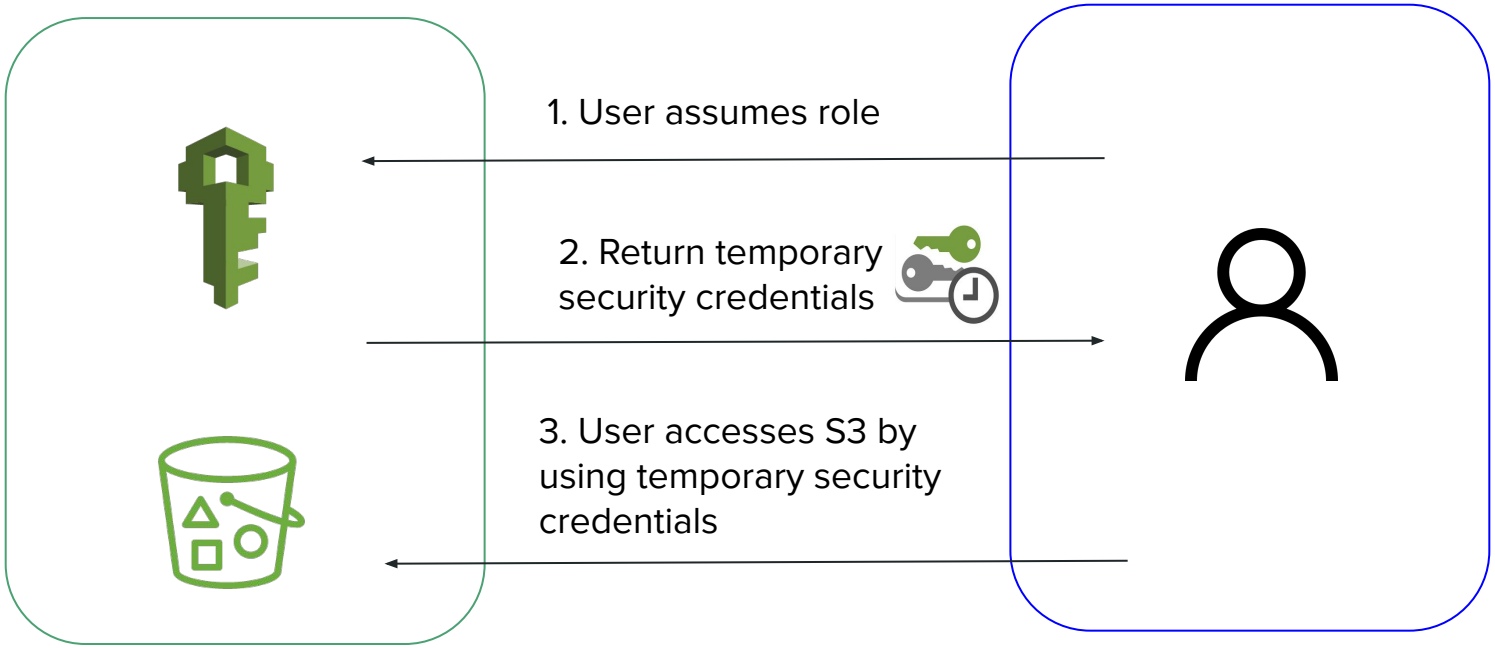
Trusting Account (Account A) which has the resources that need to be accessed.

Trusted Account (Account B) which contains users that need to access the resources in the Trusting Account.

1. A role is created in the Trusting Account `cross-account-role-demo`
2. A Trust is established with the role by the AWS Account number of the Trusted Account.
3. Permissions applied to the role via policies.
4. Users in the Trusted Account have a policy attached to *AssumeRole*.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "sts:AssumeRole",
    "Resource": "arn:aws:iam::AccountAID:role/cross-account-role-demo"
  }
}
```

How does it work?



Demo: IAM Security Tools

IAM Security Tools

- IAM Credentials Report (account-level)
  - A report that lists all users and the status of their various credentials.

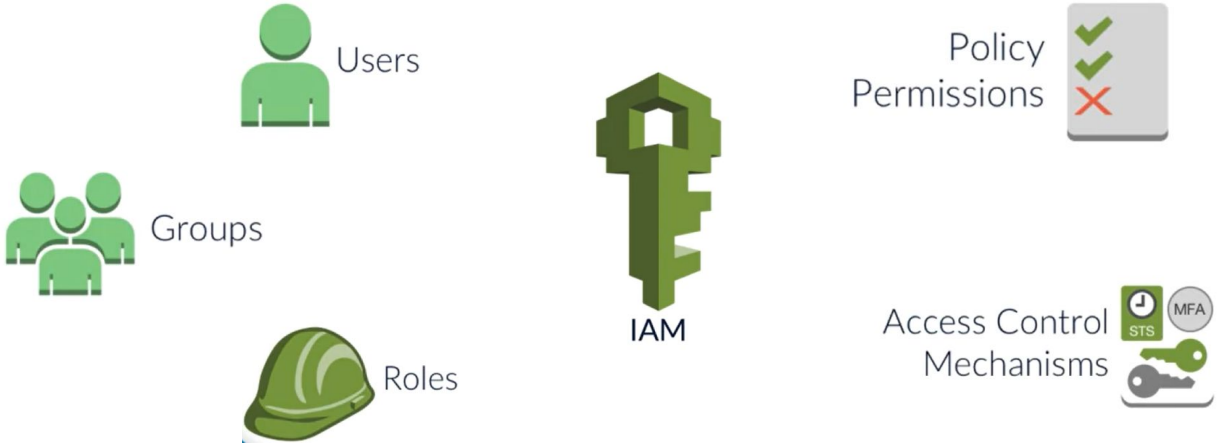
- IAM Access Advisor (user-level)
  - Shows service permissions granted to user and when those services were last accessed.
  - You can use this information to revise your policies.

From Stephen Maarek Slides

IAM Guidelines and Best Practices

- Don't use the root account except for AWS account setup
- One physical user = One AWS **user**
- **Assign users to groups** and assign permissions to groups
- Create a **strong password policy**
- Use and enforce the use of **Multi Factor Authentication (MFA)**
- Create and use **Roles** for giving permissions to AWS services
- Use Access Keys for Programmatic Access (CLI / SDK)
- Audit permissions of your account with the IAM Credentials Report
- Never share IAM users & Access Keys

Summary


Users
Groups
Roles
IAM
Policy
Permissions
Access Control
Mechanisms
STS
MFA

There are other IAM concepts not covered today

- Identity federation and Cognito
- Organizations
- Service control policies
- AWS SSO