Seeking Security and Identity:

Data Protection:

1. Amazon Macie:
   - Discover and protect your sensitive data

2. AWS Key Management Service:
   - Store and manage encryption keys

3. AWS CloudHSM:
   - Hardware based Key Storage

4. AWS Certificate Manager:
   - Provision, manage and deploy SSL and TLS security certificates

5. AWS Secret Manager:
   - Rotate, manage and retrieve secrets

Infrastructure Protection:

1. AWS Shield:
   - Denial of service protection

2. AWS Web Application Firewall:
   - Filter malicious website traffic

3. AWS Firewall Manager:
   - Centrally manage firewall rules

Threat Detection:

1. Amazon GuardDuty:
   - Automatically detect threats

2. Amazon Inspector
   - Analyse application security

3. AWS Config:
   - Record and evaluation configuration of your AWS resources

4. AWS CloudTrail:
   - Track use activity and API usage

Identity Management:

1. AWS IAM:
   - Securely manage access to AWS account services and resources

2. AWS Single Sign-on:
   - Implement cloud single sign-on

3. Amazon Cognito:
    - Manage identity inside applications

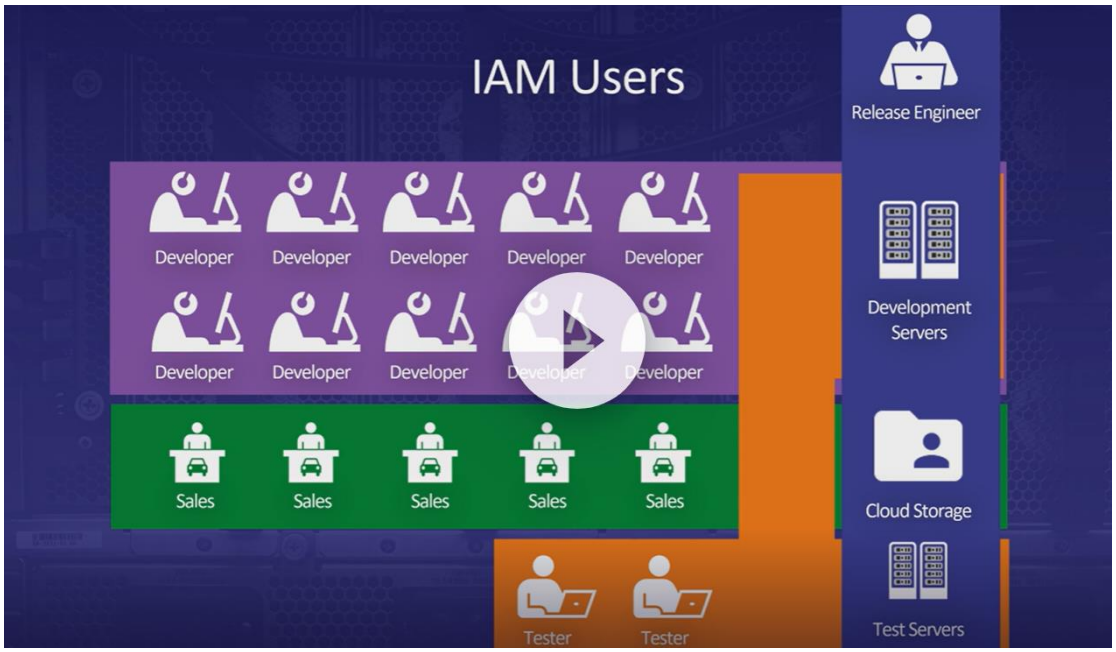
4. AWS Directory Service:
    - Implement and manage Microsoft active directory

5. AWS Organizations:
    - Centrally govern and manage multiple AWS accounts in one place

AWS Identity and access management (IAM):

- Manage who can access what in your AWS accounts
- Create users and groups
- Allow or deny access via policies
- Free service in all accounts

IAM Users:



IAM Policy examples:

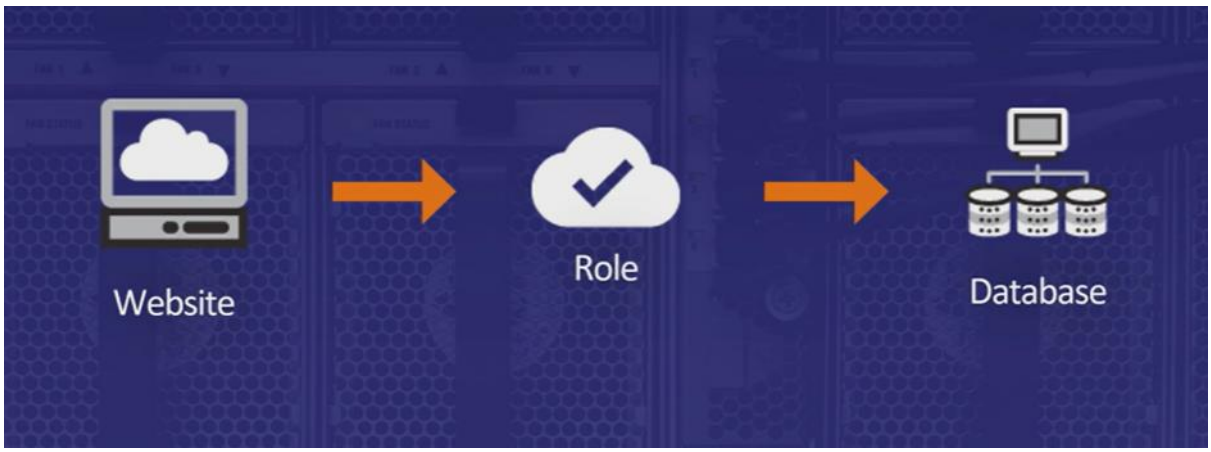
1. Administrator user

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

2. S3 bucket access

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Effect": "Allow",
         "Action": [
            "s3:ListBucket"
         ],
         "Resource": "arn:aws:s3:::bucket-name"
      },
      {
         "Effect": "Allow",
         "Action": [
            "s3:PutObject",
            "s3:GetObject"
         ],
         "Resource": "arn:aws:s3:::bucket-name/*"
      }
   ]
```

IAM Roles:



Summarizing Secret Manager:

- Protects the secrets required to access your resources
- Rotates automatically
- Stores passwords, keys and tokens

mySQL Secret Manager Example:

```
import mysql.connector

connection = mysql.connector.connect(
    host='localhost',
    database='apetguru',
    user='root',
    password=get_secret_value_response['SecretString']
)
```

Two main features:

- Saving time
- Securing application and rotating your keys properly.