AWS(Amazon web services) is a cloud computing platform with an ever-expanding set of services.

Cloud architecture design principles

It is based on 6 pillars:

1. **Operational excellence** - The operational excellence pillar focuses on running and monitoring systems, and continually improving processes and procedures.

**2. Security** - The security pillar focuses on protecting information and systems.

**3. Reliability** - The reliability pillar focuses on workloads performing their intended functions and how to recover quickly from failure to meet demands.

**4. Performance efficiency -** The performance efficiency pillar focuses on structured and streamlined allocation of IT and computing resources

**5. Cost optimization -** The cost optimization pillar focuses on avoiding unnecessary costs.

**6. Sustainability -** The sustainability pillar focuses on minimizing the environmental impacts of running cloud workloads.

**• Regions:** It is a physical location spread across globe to host your data to reduce latency. In each region there will be at least two availability zones for fault tolerance

• **Availability Zone – AZ:** It is a combination of one or more data centers in each region

**• Other options-**

Local Zones | Wavelength Zones | Direct Connect Locations | Edge locations | Regional edge caches

**Ways to interact with AWS Services**

AWS Management Console | AWS SDKs (software development kit) (AWS SDK for Python (Boto3)) | AWS Command Line Interface (CLI) |REST APIs

# Boto3

To use Boto3, you must first import it and indicate which service or services you're going to use:

**import** **boto3**

s3 **=** boto3**.**resource('s3') *# Let's use Amazon S3*

Now that you have an s3 resource, you can make send requests to the service. The following code uses the buckets collection to print out all bucket names:

*# Print out bucket names*

**for** bucket **in** s3**.**buckets**.**all():

print(bucket**.**name)

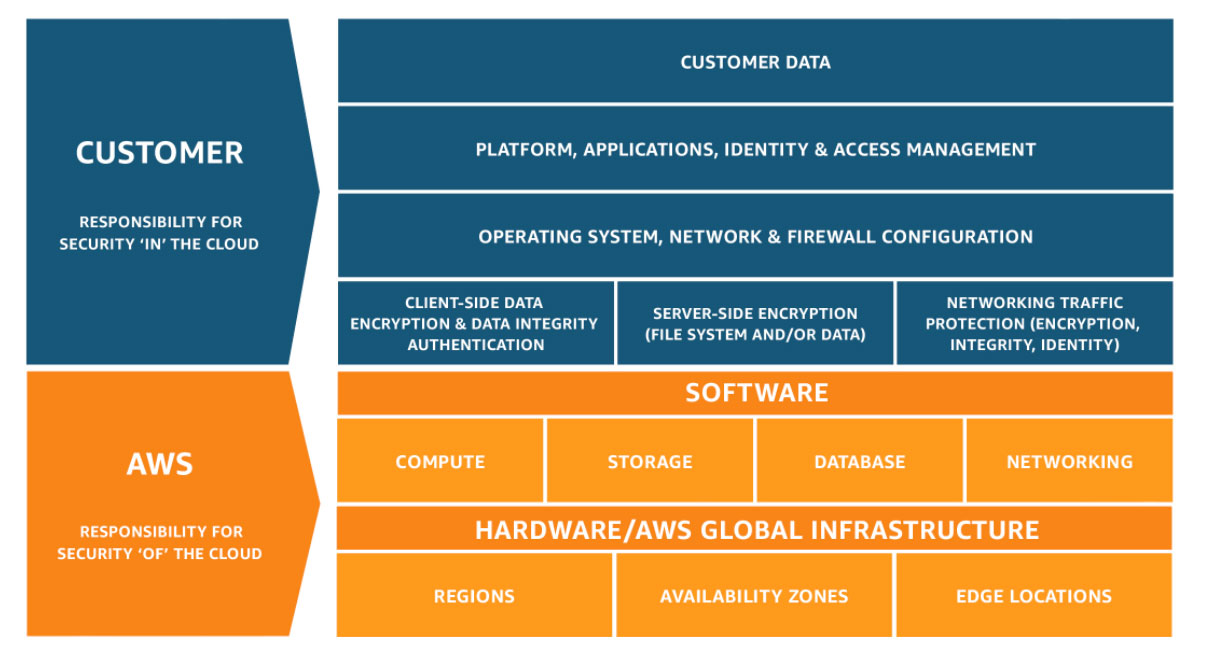
You can also upload and download binary data. For example, the following uploads a new file to S3, assuming that the bucket my-bucket already exists:

*# Upload a new file*

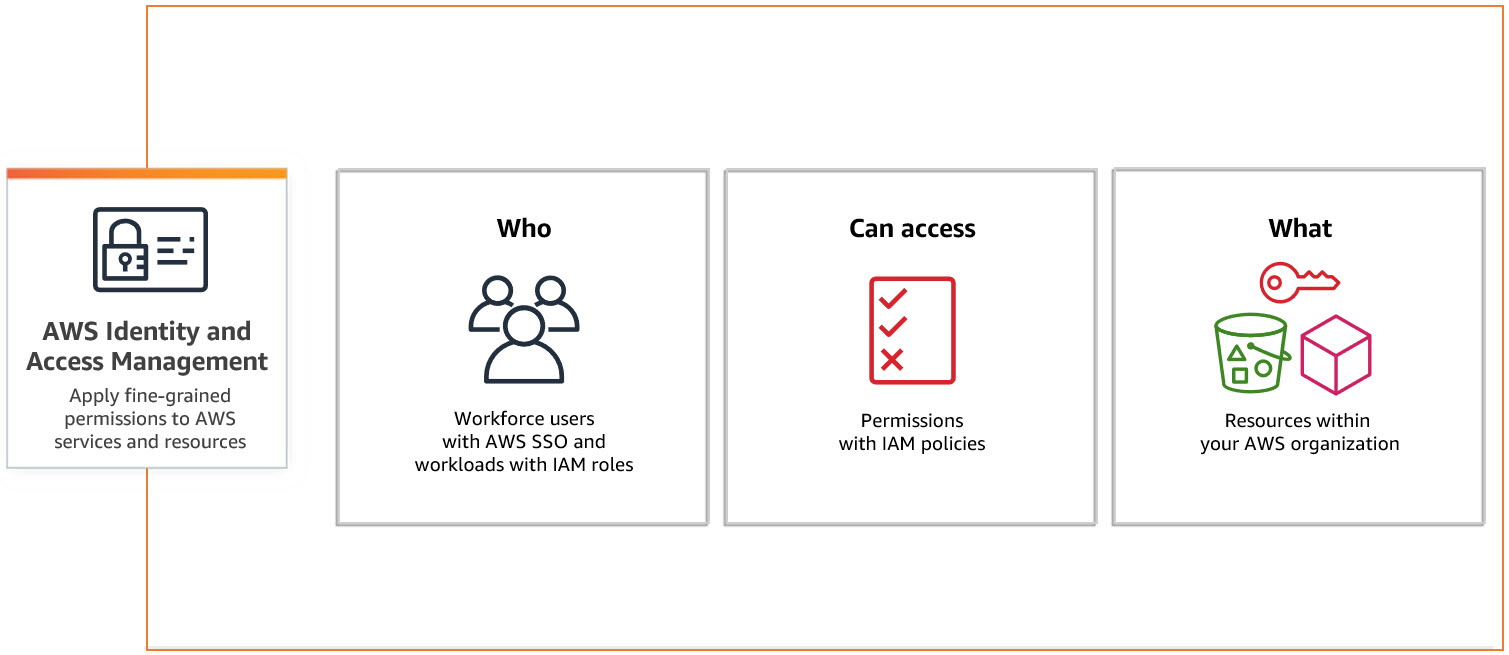
data **=** open('test.jpg', 'rb')

s3**.**Bucket('my-bucket')**.**put\_object(Key**=**'test.jpg', Body**=**data)

# AWS shared responsibility model



# IAM (Identity and access management)



# Security best practices in IAM

* [Lock away your AWS account root user access keys](https://docs.aws.amazon.com/IAM/latest/UserGuide/best-practices.html#lock-away-credentials)
* [Use roles to delegate permissions](https://docs.aws.amazon.com/IAM/latest/UserGuide/best-practices.html#delegate-using-roles)
* [Grant least privilege](https://docs.aws.amazon.com/IAM/latest/UserGuide/best-practices.html#grant-least-privilege)
* [Get started using permissions with AWS managed policies](https://docs.aws.amazon.com/IAM/latest/UserGuide/best-practices.html#bp-use-aws-defined-policies)
* [Validate your policies](https://docs.aws.amazon.com/IAM/latest/UserGuide/best-practices.html#best-practice-policy-validation)
* [Use customer managed policies instead of inline policies](https://docs.aws.amazon.com/IAM/latest/UserGuide/best-practices.html#best-practice-managed-vs-inline)
* [Use access levels to review IAM permissions](https://docs.aws.amazon.com/IAM/latest/UserGuide/best-practices.html#use-access-levels-to-review-permissions)
* [Configure a strong password policy for your users](https://docs.aws.amazon.com/IAM/latest/UserGuide/best-practices.html#configure-strong-password-policy)
* [Enable MFA](https://docs.aws.amazon.com/IAM/latest/UserGuide/best-practices.html#enable-mfa-for-privileged-users)
* [Use roles for applications that run on Amazon EC2 instances](https://docs.aws.amazon.com/IAM/latest/UserGuide/best-practices.html#use-roles-with-ec2)
* [Do not share access keys](https://docs.aws.amazon.com/IAM/latest/UserGuide/best-practices.html#sharing-credentials)
* [Rotate credentials regularly](https://docs.aws.amazon.com/IAM/latest/UserGuide/best-practices.html#rotate-credentials)
* [Remove unnecessary credentials](https://docs.aws.amazon.com/IAM/latest/UserGuide/best-practices.html#remove-credentials)
* [Use policy conditions for extra security](https://docs.aws.amazon.com/IAM/latest/UserGuide/best-practices.html#use-policy-conditions)
* [Monitor activity in your AWS account](https://docs.aws.amazon.com/IAM/latest/UserGuide/best-practices.html#keep-a-log)

IAM > Access Management

## **User groups**

A user group is a collection of IAM users. Use groups to specify permissions for a collection of users.

## **Users**

An IAM user is an identity with long-term credentials that is used to interact with AWS in an account.

## **Roles**

An IAM role is an identity you can create that has specific permissions with credentials that are valid for short durations. Roles can be assumed by entities that you trust.

## **Policies**

A policy is an object in AWS that defines permissions.

AWS has several building block infrastructure services in each of the compute, storage, database, security, management, and networking categories.

**Compute Service**

Graphical user interface, application

Description automatically generated