Different types of cloud computing models that are available:

**1)Infrastructure as a Service(IAAS):** contains the basic building blocks for cloud IT.

EG: VPC, EC2, EBS.

if we want to launch a Linux server and we want to manage that Linux server ourselves that is how we would do that as infrastructure as a service. And we would do that using the Elastic Compute Cloud or easy to service.

**2)Platform as a Service(PAAS):** Manages the underlying infrastructure usually hardware and operating systems.

Eg: RDS, EMR, Elastic Search.

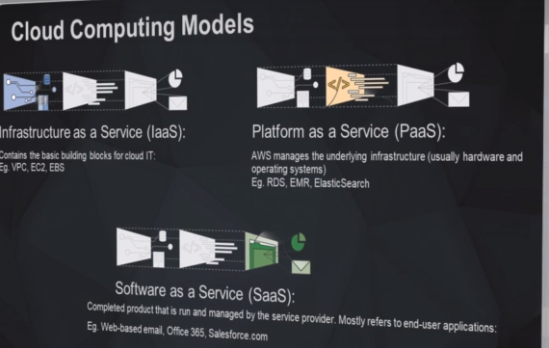
a good example that would be the relational database service. And in that service Adobe ways they provision all the operating system the server and everything to

run that.

But you still need to do the high level administration of that database.

**3)Software as a Service(SAAS):** completed product that is run and managed by the service provider that refers to end user application.

Eg: Web-based email, Office 365, Salesforce.com



**Serverless Computing:**

* Allows you to build and run applications without having to think about servers you know. You don't need to provision the server yourself. AWS would do that for you.
* It's also referred to as function as a service or abstract services.

Examples:

**Amazon Simple Storage Service(S3):** where we create a bucket and we put objects and falls into that bucket.

We don't know what's behind that bucket.

Obviously there's going to be an operating system most probably a Linux operating system a file server there's going to be hard drives.

We don't need to worry about that because AWS was they look after all that for us.

**AWS Lambda:** we can run code in the cloud again without service.

You just provide AWS with your code that AWS looks after everything for you.

**AWS Dynamo DB:** Dynamo DB is a no sequel database in the cloud as a service and Amazon simple notification service that can send out notifications to your users.

**AWS Storage Services:**

**1)Simple Storage Service(S3):**

It's designed to store and access any type of data over the Internet.

It is a serverless service and as such we don’t need to worry about what is behind it.

There is obviously a file server or an operating system or a hard drive but we don’t need to be concerned about that all.

We just need to create this thing called a bucket and then we upload objects to that bucket so that the bucket grows as we add objects to it and the size of the bucket is theoritically unlimited.

**2)Amazon Glacier:**

It is the cheapest storage option on AWS and it is used for long term archiving of data.

It is a serverless service just like Amazon S3 but it is not as ready as accessible as S3. So it should only be used for content that is to be archived.

You can also set up a lifecycle rule that will automatically migrate old data in Amazon S3 automatically over to Glacier for long term archiving.

**3)Amazon Elastic Block store(EBS):**

It is a highly available low latency block storage and its specifically for attaching to servers that are launched with the Amazon EC2.

It is similar to attaching a hard drive to your computer at home. Works in the same manner It is blocked device storage.

**4)Amazon Elastic File System(EFS):**

It is a network attached storage and it's specifically for Amazon EC2 servers because it is network attached storage.

This allows multiple servers to access a one data source. In a similar way to a NES on your network at home can be accessed by multiple computers on that network:

**AWS Storage Gateway:**

Enables hybrid storage between on premise environments and the AWS cloud.

It provides a low latency performance by casing frequently used data on premises while storing the less frequently data in Amazon cloud storage services

**Amazon Snowball:**

A snowball device is a portable petabytes scale data storage device that can be used to migrate data and large amounts of data from on premise environments over to the AWS cloud.

You simply download your data to the snowball device then you send it off to AWS who will then upload that data to an AWS storage service for you

**Storage Example:**

So let's have a look at some examples of using the AWS storage services

in orange there we've got the AWS cloud.

Now we can create a VPC inside that AWS cloud and that VPC or virtual private cloud is our own private space within the AWS cloud and that is an impenetrable fortress against attack and no one will be able to enter our own private space without us allowing that to happen.

So let's just say we launch two servers in our VPC.

Now we want these servers to have access to data and somewhere to store that data.

And so in a normal environment you would just add a harddrive to that server.

So in the same way we can attach an Amazon Elastic block storage device to our servers.

So that's great.

We've now got high speed access to our data but what if we want that data to be available to both of those servers. So here we've only got all we've got EBS volumes.

What do we want that data to be on one volume only?

So as we know it on our computer at home we can't attach a harddrive or block device harddrive to multiple computers. It just doesn't work like that.

So in a situation like that in your home network at home you would just go out and purchase a NES a network attached storage device you would attach it to your network you would set up your operating system in your desktop computers to have a mount target for that network attached storage.

So when you go to your G drive or whatever it is or e drive or f drive whatever it is that will point

to that network attached storage. In the same way that we can do that with our network at home we can do the same thing with a AWS elastic file system is network attached storage and so that with a mount target can enable multiple servers to access the one data source.

Now what if we don't want to worry about Mount targets and block devices and all the sort of stuff we just want somewhere we can upload objects to in a similar way do we do with Google Drive or something like that.

And we also want to have an automated solution that over time migrates that data over to something more low cost and more long term for archiving.

Now that is where Amazon S3 comes in.

And so we can use Amazon S3 to create a bucket, store objects in that bucket, delete objects do whatever we want with it and we can also set up a lifecycle rule on that bucket so that over a period of time as objects age they can be migrated over to an Amazon Glacier vault for long term archiving it will still be accessible.

It just won't be as readily accessible as the three bucket.

But the advantage is that we'll be using the lowest cost storage that's available on AWS.

Now that S3 bucket will be located in the AWS cloud.

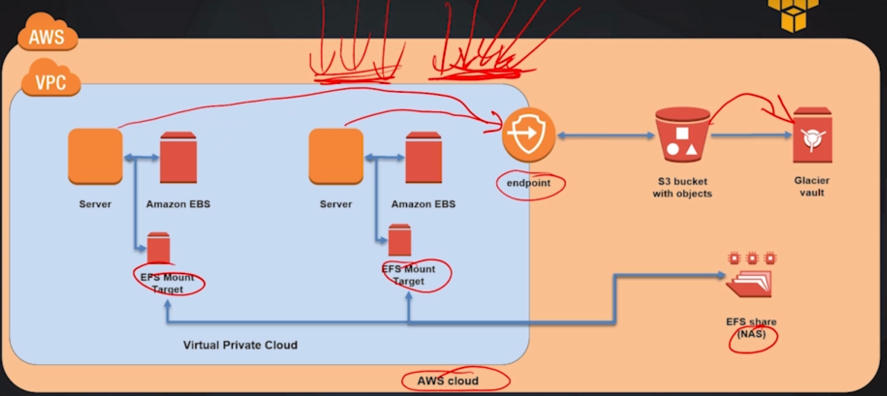
It's not located in our VPC.

So remember we said the VPC is our private space within the AWS cloud and nothing gets through

it without us allowing it to come through.

So that is where the VPC end point comes in.

So we can create one of those and that will allow traffic to flow in and out of our VPC specifically for that is three service



so let's have a look at a hybrid storage example where we've got onsite storage

in a corporate data center and we've also got that stored in the IDB cloud in Amazon S three.

Why would we do that.

Well it's great for a disaster recovery solution because it provides high speed access to our data in our corporate data center.

And at the same time we're taking advantage of the durability and availability of Amazon S3 as a disaster recovery solution.

So the first problem that we're going to encounter is that these corporate data center will have petabytes of data and to transfer that over via the Internet to the AWS cloud is not going to be practical so AWS they can send out to us a snowball device.

And that is a high capacity device they can store petabytes of data.

And so we can upload when we receive that snowball device from AWS.

We can upload our data to that and then we can send that back to AWS and they will upload that for us into the Amazon S3 bucket.

And so that solves that problem for us.

So then we've got to find a solution for making sure that the data in our corporate data center is synched

with our AWS cloud.

Now that's where the AWS Gateway comes in.

And that will orchestrate all of that for us.

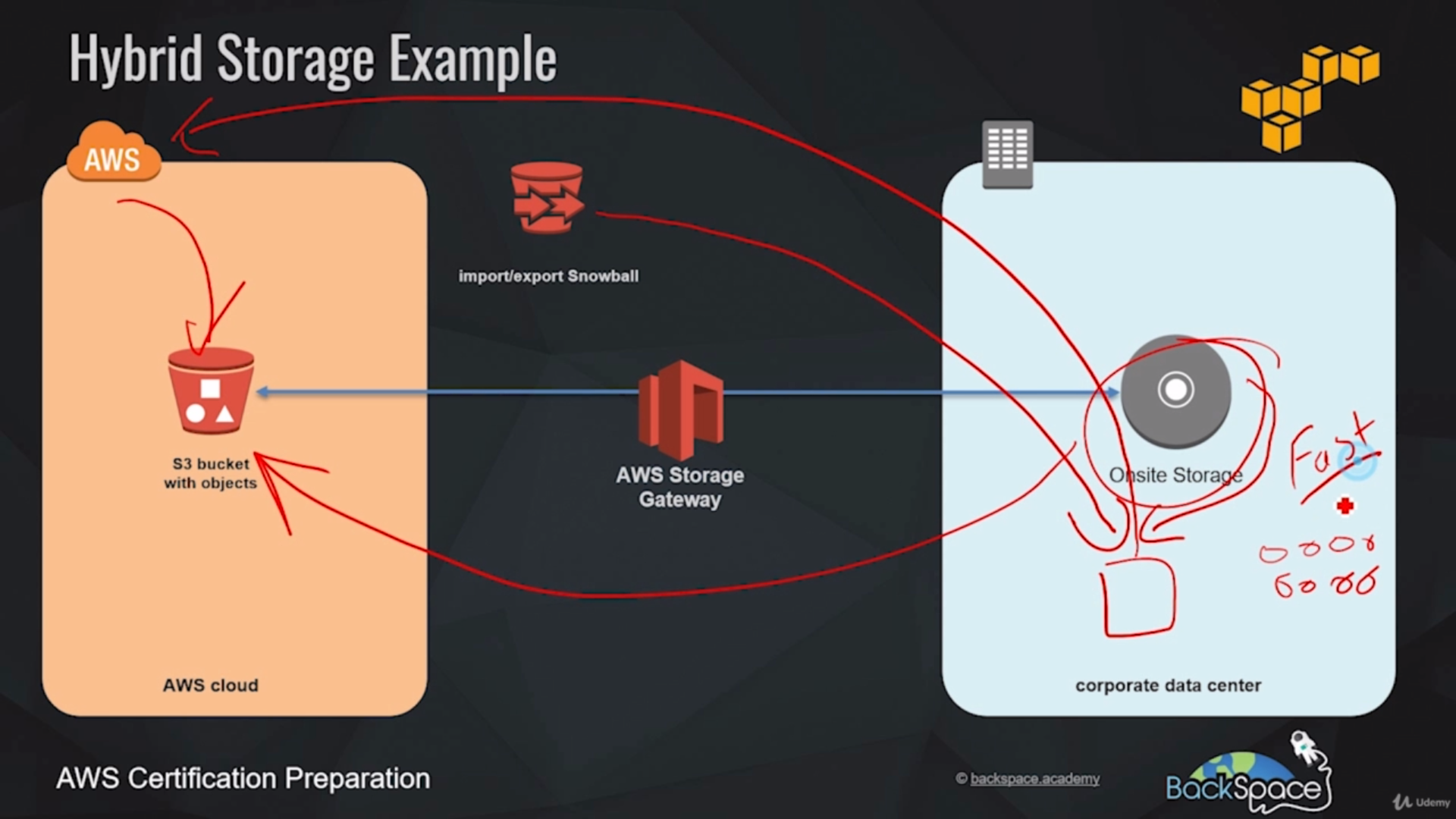
And so if you have got a high speed link between your corporate data center and the AWS cloud which you can with the AWS direct connect service you can have the AWS storage gateway to orchestrate and manage that all for you.

And what it will do it will get your popular content your content that is frequently accessed and it will store copies of that on site in your onsite storage.

But at the same time it will store all of that data in an Amazon S3 bucket for you.

And so then you've got the advantage of having all of the all of the durability and availability of Amazon S3 as a disaster recovery solution.

But at the same time you've got high speed access to your data which is cased on the corporate data center.



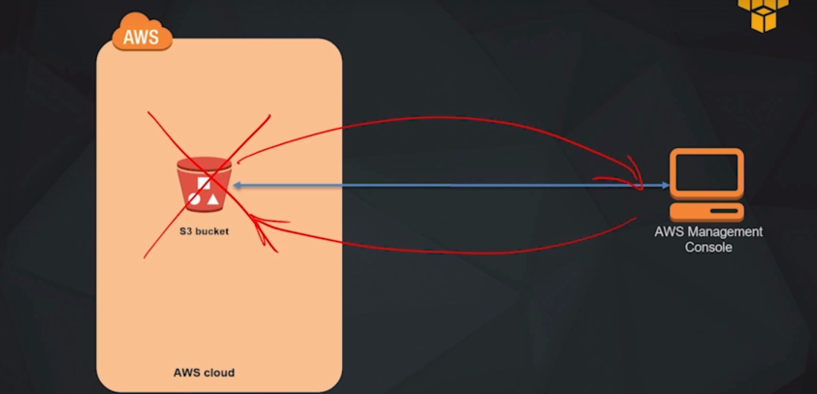
Let's have a go at using the Amazon S3 service.

So what we're going to do now is that we're going to use the AWS management console to connect into

the AWS cloud and then we're going to create an Amazon S3 bucket.

We're going to upload files to that bucket and then we're going to download files from that bucket.

And then finally we're going to empty and delete that bucket.



Introduction to Database services:

Different options that are available for running databases on AWS

**Relational Database services:**

1. The relational database service or RDS for short is a fully managed database service that makes it easy to launch database servers in the AWS cloud and scale them when required
2. The RDS service can launch service for mySQL including variations of the mySQL database engine with MariaDB and Amazon's own enterprise version of mySQL.
3. Amazon Aurora, standard postgre SQL is also available and also available as Amazon's Enterprise Aurora postgre SQL, Microsoft SQL server and oracle are also available

**Amazon Dynamo DB:**

1. Amazon DynamoDB is AWS's noSQL database as a service.
2. It's a serverless service like Amazon s3 and as such you don't need to worry about the underlying infrastructure behind it. AWS takes care of everything for you and it provides high speed extremely low latency performance

**Amazon Redshift:**

1. Amazon redshift is a fast fully managed petabyte scale data warehouse that is based upon the postgre SQL database engine.
2. If you're looking for a big data storage solution redshift is perfect for this.

**Amazon ElastiCache:**

* Amazon ElastiCache is an in-memory data store or cache in the cloud.
* it allows you to retrieve information from fast fully managed in-memory caches instead of relying for slower disk based databases.

**AWS Database migration service:**

* The AWS database migration service orchestrates a migration of databases over to AWS easily and securely
* It can also migrate data from one database engine type to another totally different database engine type
* For example you can use it to migrate from Oracle over to Amazon Aurora

**Amazon Neptune:**

* Amazon Neptune is a fast reliable fully managed graph database service
* It has a purpose-built high performance graph database engine optimized for storing billions of relationships and clearing the graph with millisecond latency.

**How to use these Database services:**

Example:

let's say we have an on site Oracle relational database and we want to migrate that over to Amazon Aurora on the AWS cloud

The first thing we could do is to launch an RDS instance in our virtual private cloud and remember VPC i.e. it's our own private space within the AWS cloud and no one can enter it without us allowing them to enter

we could use a database migration service to migrate that data in that on-site Oracle database over to a target RDS Amazon Aurora server.

Now let's say our new database is becoming overwhelmed with requests for frequently accessed data and we would like a high-speed way of accessing that frequently accessed data and this is where ElasticCache can help us.

We can put an ElasticCache node in front of that our RDS instance and that will cache our frequently accessed data and because it's delivering that data from memory and it's not delivering it from a hard drive it will be delivered with very low latency and at the same time the load on our database will be massively reduced and any requests for anything that is not in the elastication will be simply forwarded to the RDS instance and that way we have high-speed access to our frequently accessed data while at the same time we can still access our less frequently accessed data directly from our RDS instance.

