Unit 5: Disaster Recovery

Disaster Recovery Definition

Disaster Recovery (DR) is the term used to prepare for and recover from a disaster. A disaster can include anything that puts your organization's operations at risk like cyber attacks, equipment failure, natural disasters, terrorist attacks.

Traditional Disaster Recovery

Following the traditional methods of Disaster Recovery, companies duplicate their environments to another data center in a location remote to the primary site.

The remote data center infrastructure has to be purchased, installed, and maintained so it is ready whenever it is required. The Disaster Recovery site is often under-utilized or over-provisioned, and is nothing but wastage of money.

Cloud Disaster Recovery

With AWS you can bring up a Disaster Recovery site in minutes in locations all over the world and pay for it only when you are using it.

Traditional Versus Cloud

	Traditional	AWS
Facilities: Data centers, power, and cooling, and so on	User responsibility	AWS responsibility
Security to ensure the physical protection of assets	User responsibility	AWS responsibility
Suitable capacity to scale the environment	User responsibility	AWS responsibility
Support for repairing, replacing, and refreshing the infrastructure	User responsibility	AWS responsibility
Internet service provider contracts to provide bandwidth utilization for your environment under full load	User responsibility	AWS responsibility
Network infrastructure such as firewalls, routers, switches, and load balancers	User responsibility	AWS responsibility
Enough server capacity to run mission critical services	User responsibility	AWS responsibility

RTO and RPO

Recovery Time Objective (RTO)

Recovery Time Objective (RTO) is defined by the organization. RTO is the maximum acceptable delay between the interruption of service and restoration of service. This determines what is considered an acceptable time window when service is unavailable.

RTO is the duration of time it should take to **restore** all applications and systems after an outage.

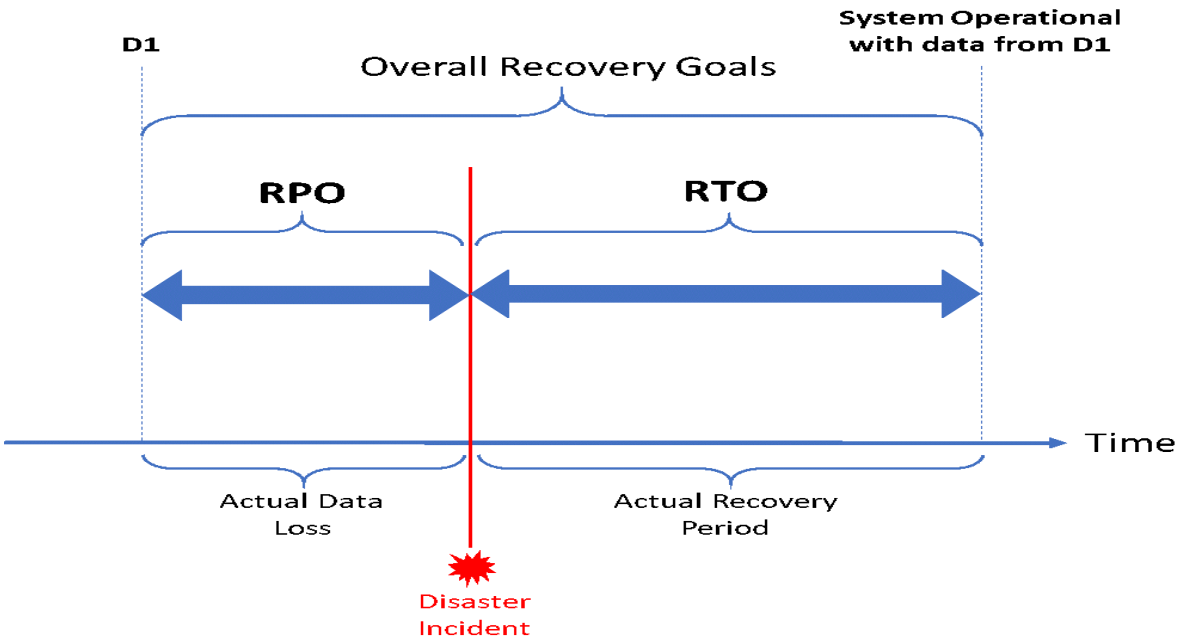
For example, if the RTO of an e-commerce site is two hours and a disaster occurs at 1pm, the DR strategy must kick-in a process to restore the site at 3pm.

Recovery Point Objective (RPO)

Recovery Point Objective (RPO) measures the acceptable amount of data loss measured in time.

Recovery Point Objective (RPO) is defined by the organization. RPO is the maximum acceptable amount of time since the last data recovery point. This determines what is considered an acceptable loss of data between the last recovery point and the interruption of service.

For example, if the RPO of a financial trading system is 30 minutes and a disaster occurs at 2pm, the DR strategy must kick-in a process to recover all of the data before 1:30pm. Data loss can only span between 1:30pm and 2pm.



Disaster Recovery Methodologies

- Backup and restore
- Pilot light
- Warm standby
- Multi-region (multi-site) active-active

Backup and restore (RPO in hours, RTO in 24 hours or less):

This is a traditional method of Disaster Recovery where data is backed up at regular intervals and in the event of a disruption or disaster, data is restored from the backup location. Given the lead time in transferring and restoring data from backup, this method is only suitable where hours of RTO and RPO are acceptable. In the event of a disaster, data is recovered from the backup tapes and restored; but it can take a long time.

Figure below shows the backup of data to S3 from either on-premise infrastructure or from AWS. Workloads running in AWS are backed up by taking snapshots of the storage disks which are natively supported for a majority of the services and stored in S3. For on-premise backup and restore, data can be transferred to and from AWS using the following services:

- AWS Direct Connect or Virtual Private Network (VPN)
- AWS Snowball
- AWS Storage Gateway
- Amazon S3 Transfer Acceleration

To facilitate quick recovery in the event of a disaster, the application is pre-configured on an EC2 instance and converted to Amazon Machine Image (AMI) stored in S3.
In the event of a disaster, data is retrieved from S3 and a pre-configured AMI is used to recreate the production environment in AWS.

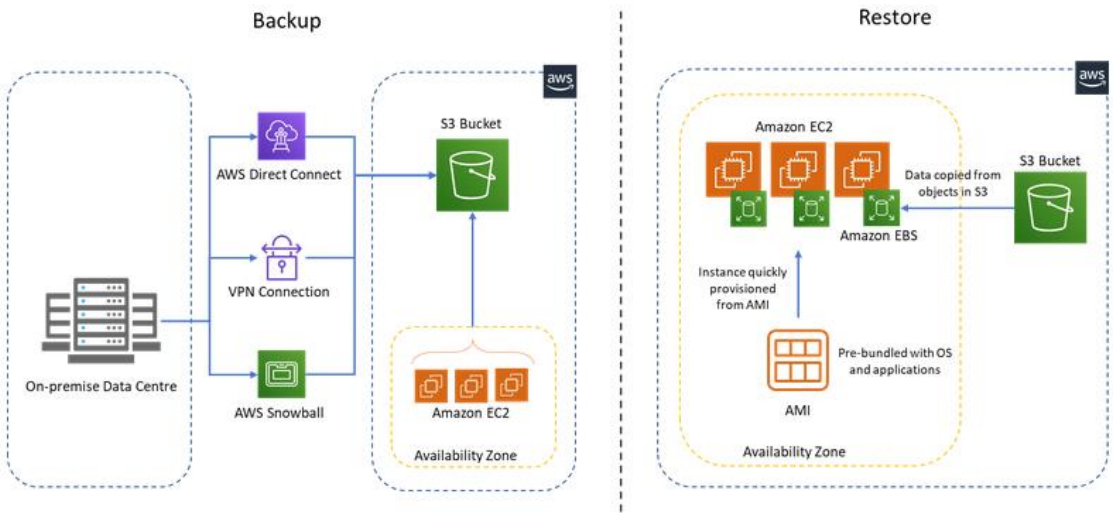


Figure 2: Backup and Restore Implementation on AWS

Backup and Restore Key Steps

The key steps to configure backup and restore are as follows:

- Select an appropriate tool or method to back up the data into AWS
- Ensure you have an appropriate retention policy for this data
- Ensure appropriate security measures are in place for this data
- Regularly test the recovery of this data and the restoration of your system

Pilot light (RPO in minutes, RTO in hours):

Pilot light allows you to run a minimal version of your environments in the cloud. In case of a disaster, you can scale up the standby copies of your core systems up to production capacity. In pilot light the RTO is medium and the RPO is low-to-medium depending on the frequency of replication.

Pilot Light strikes an excellent balance between affordability and reliability. One of the key differences between Backup and Restore and Pilot Light is that Pilot Light will always have core functionality running in the cloud. For instance, in Backup and Restore your data will be synced in an S3 bucket for retrieval in the event of a disaster. With Pilot Light, the data is synced with a database replica that is always on and ready to go.

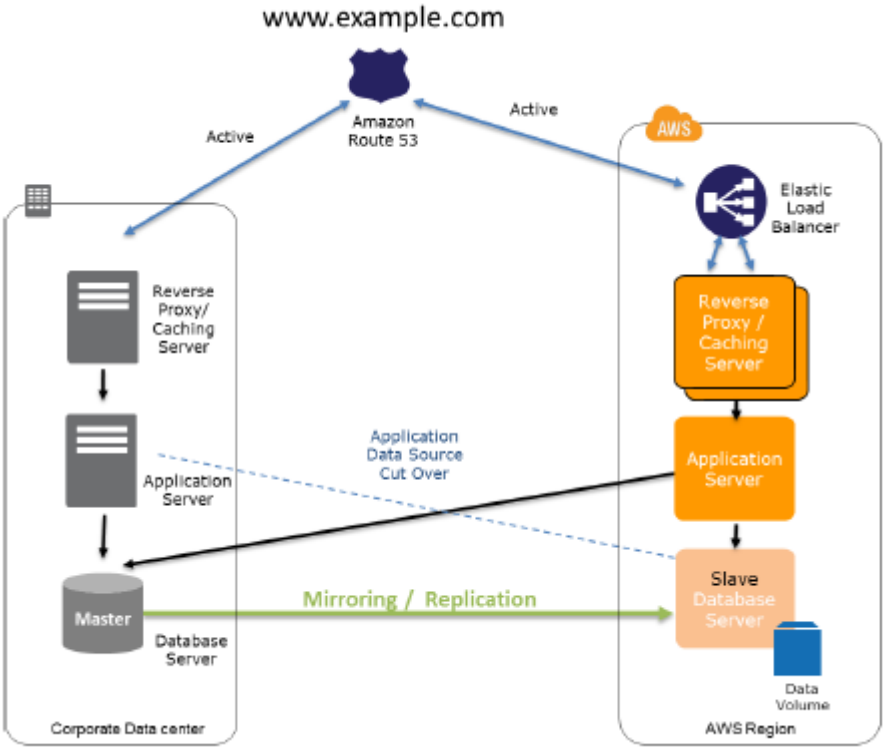
Another great feature of Pilot Light — and the subsequent DR methods — is that it's largely automated. On the AWS console, an administrator can create a health check to verify the accessibility of a particular URL. For example, the homepage of the website you intend to

backup. Then, in the event that this health check fails, the Pilot Light environment will be switched on.

One quick way that this could be done is for the health check to send an SNS (Simple Notification Service) message to a lambda function. If the SNS message describes a health check failure, the lambda will send a command to begin the Pilot Light EC2 Instance.

AWS Pilot Light Failover

Preparation Phase:



The following figure shows the preparation phase, in which you need to have your regularly changing data replicated to the pilot light, the small core around which the full environment will be started in the recovery phase. Your less frequently updated data such as operating systems and applications can be periodically updated and stored as Amazon Machine Images (AMIs).

Recovery Phase:

The diagram represents a scenario of a failover in the primary site, the pilot light environment is brought online.

To recover the remainder of the environment around the pilot light, you would start your systems from the Amazon Machine Images (AMIs) in minutes on the appropriate instance

types. For your dynamic data servers, you can resize them to handle production volumes as needed or add capacity accordingly. Horizontal scaling, if possible, is often the most cost effective way and scalable approach to add capacity to a system, however, it's also possible to pick larger EC2 instance types and thus scale vertically. From a networking perspective, any required DNS updates can be done in parallel.

Once recovered, you should ensure that redundancy is restored as quickly as possible. While a failure of your DR environment shortly after your production environment failed is unlikely, you need to be aware of this risk. Continue to take regular backups of your system and consider additional redundancy at the data layer.

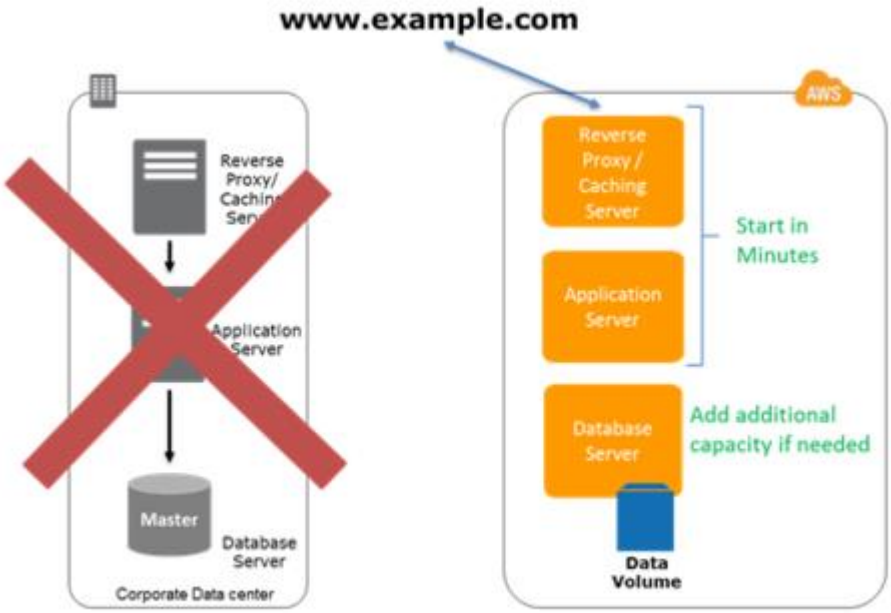


Figure 4: The recovery phase of the Pilot light scenario.

Pilot Light Key Steps

The key steps to configure pilot light are as follows:

- Ensure you have all supporting custom software packages available in AWS
- Create and maintain AMIs of key servers where fast recovery is required
- Regularly run the servers, test them, and apply any software updates and configuration changes.
- When using Amazon RDS turn on multi Availability Zones to improve resilience
- Use Route 53 to point traffic at the Amazon EC2 servers

Warm Standby

It extends the idea of pilot light by running a scaled down version of a fully functional environment always running in the Cloud.

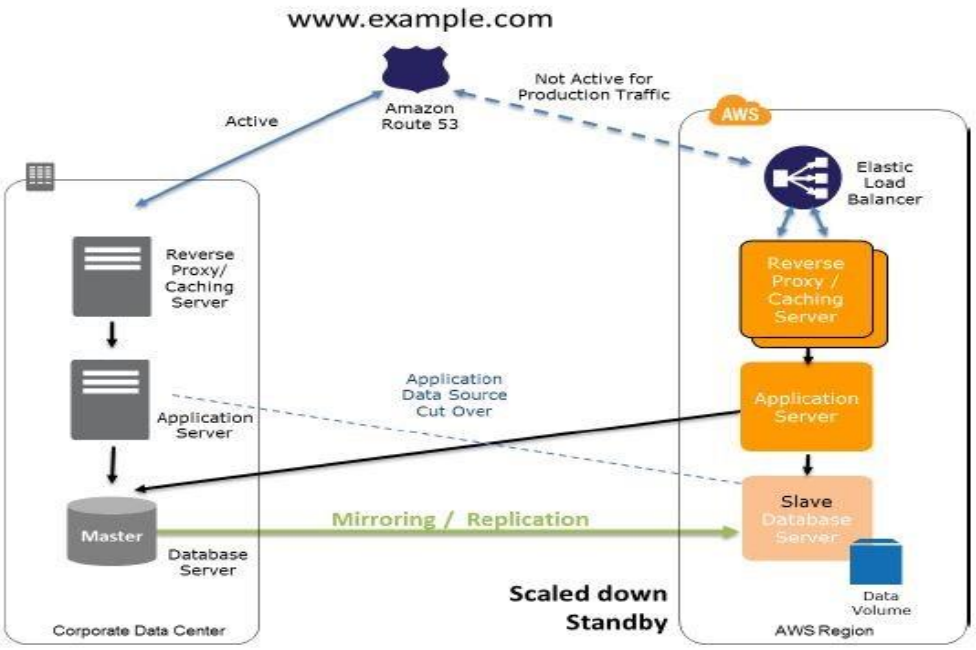
The method decreases recovery time as the critical systems are already operational. The warm standby site can either be on instances ready for production or on lower sized instances that can be scaled as required.

The RTO is low to medium and the RPO is low.

In a Pilot Light scenario, only an EC2 Instance and a DynamoDB may be running. In Warm Standby, however, everything is running — just in a much smaller capacity. This means the load balancer; gateways, databases, all subnets, and everything else are ready to go on a moment's notice.

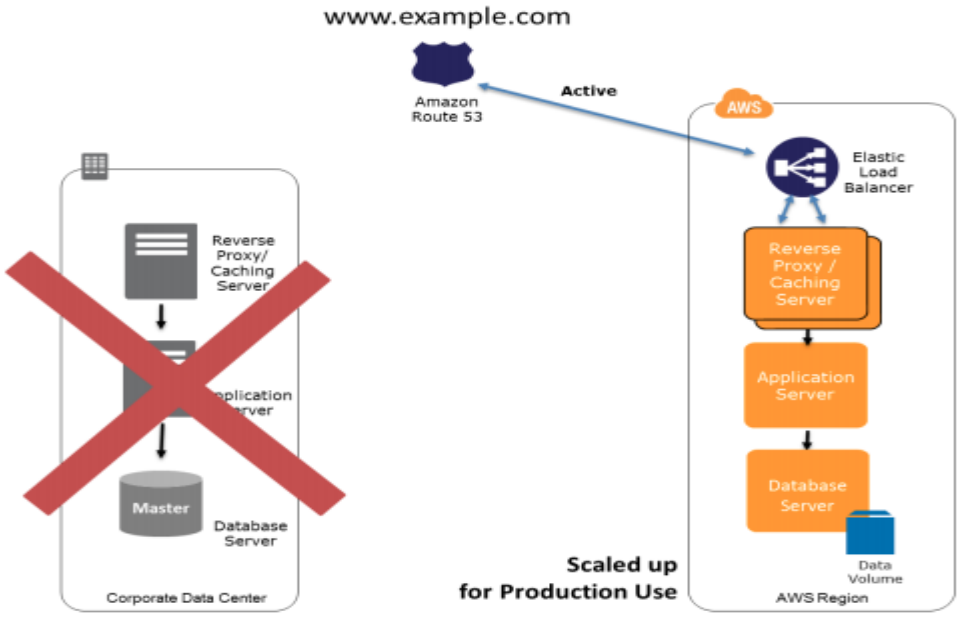
Preparation Phase:

The following diagram shows the preparation phase for a warm standby solution, in which an on-site and an AWS solution run side by side.



Recovery Phase:

The diagram represents a scenario of a failover. In case of a failover at the primary site you can just simply reroute traffic using Amazon Route 53 to the AWS site.



Warm Standby Key Steps

The key steps to configure warm standby are as follows:

- Set up Amazon EC2 instances to replicate or mirror data
- Run your application using a minimal footprint of AWS infrastructure and scale in a DR situation
- Patch, update, and change configuration files in line with your live environment
- Use Route 53 to point traffic at the Amazon EC2 servers
- Use Auto Scaling to right-size your resources to accommodate the increased load

Multi-Site

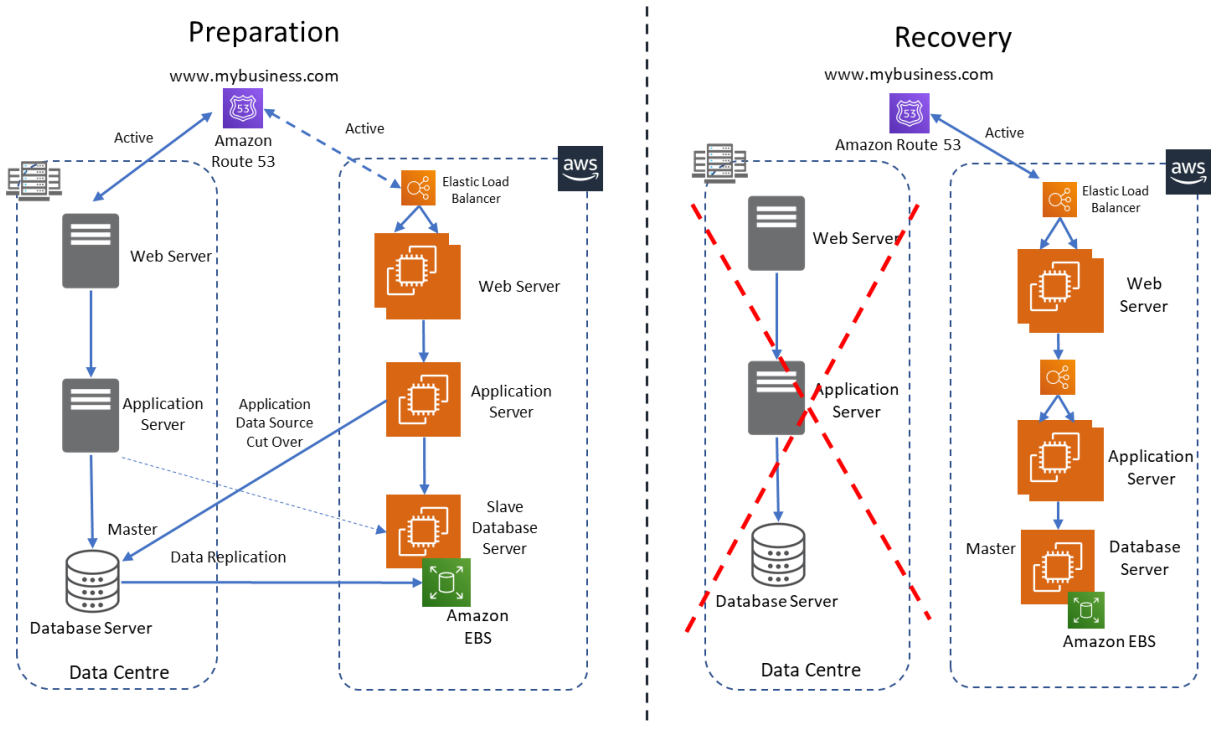
In the multisite scenario the infrastructure runs both in AWS and in existing on-premise sites in an active- active configuration.

Users can use both sites. You can direct the user traffic via Route 53 weighted routing. In this case the RTO is short and the RPO is very low.

A Multi-Site operation, also known as Hot Standby is a one-for-one replication of your production environment. It is a truly fault-tolerant system. As you can imagine, this is a very, very expensive operation. Let's take a look at a couple reasons why.

One reason is that the EC2 instances are constantly scaled out, and constantly on. This means that the company is paying for all these resources that they are not actually using. In addition, all lambda functions would have to operate in real time for both environments, which would get expensive. Lastly, multi-site requires constant testing and configuration to ensure the operation is seamless in real-life situations.

In the event of a disaster, the routing policy can be updated to send all traffic to AWS where a pre-configured autoscaling group will scale up the EC2 instances to cater for the increased production traffic. A logic will be required to detect the disaster and upgrade the slave database to the master with the database credentials updated in the application layer to reference the database on AWS.



Multi-Site Key Steps

The key steps to configure multi-site availability are as follows:

- Configure AWS environment to duplicate your production environment
- Configure DNS weighting to distribute requests to both sites and redirect traffic away from sites in Disaster Recovery situation
- Use failover application logic to use the local AWS database servers for all queries
- Use Auto Scaling to automatically right-size the AWS fleet