DISASTER RECOVERY WITH IBM CLOUD VIRTUAL SERVERS

PHASE 4: Development Part 2

Continue building the disaster recovery plan by configuration replication and testing recovery procedures.

1)Set Up Data Replication:

*Identify critical data and systems to replicate.

* replication mechanisms such as AWS S3 Cross-Region Replication.

2)Create Automated Backups:

*Use automated backup services like AWS RDS automated backups for databases.

3) Implement Disaster Recovery Testing:

*Create scripts or automation for testing recovery procedures.

4) Regularly Test Failover:

*Schedule regular tests of failover to ensure the process works.

Example of scheduling an AWS Lambda function for failover testing

Aws events put-rule –name “FailoverTestRule” –schedule-expression “rate(1 day)”

Aws lambda create-function –function-name FailoverTest –runtime nodejs12.x –handler index.handler –zip-file fileb://function.zip

Aws events put-targets –rule FailoverTestRule –targets “Id”=”1”,”Arn”=”arn:aws:lambda:us-east-1:123456789012:function:FailoverTest”

5) Monitor and Alerting:

*Implement monitoring and alerting for any failures in the replication and recovery process.

Example of creating an AWS CloudWatch alarm for RDS instance status

Aws cloudwatch put-metric-alarm –alarm-name RDSInstanceStatusAlarm –alarm-description “Alarm for RDS instance status” –actions-enabled –alarm-actions arn:aws:sns:us-east-1:123456789012:MySNSTopic –metric-name DBInstanceStatus –namespace AWS/RDS –statistic Average –period 300 –threshold 1 –comparison-operator GreaterThanThreshold

6) Documentation and DR Runbooks:

*Maintain detailed documentation and runbooks for recovery procedures.

Implement replication of data and virtual machine images from on-premises to IBM cloud virtual servers.

1)Set up the IBM Cloud Environment:

First, ensure you have an IBM Cloud account.

Create the necessary virtual servers on IBM Cloud that you want to replicate to.

2)Connectivity:

Establish a secure connection between your on-premises environment and IBM Cloud. This may involve setting up a VPN, direct link, or using IBM Cloud Direct Link.

3)Data Replication:

Choose a data replication method, depending on your needs (e.g., real-time, periodic, one-time).

You can use tools like rsync, SCP, or IBM Cloud’s Object Storage service to transfer data.

4)Virtual Machine Replication:

For virtual machine replication, consider using IBM Cloud’s services such as IBM Cloud Virtual Servers or IBM Cloud VMware Solutions, which allow you to create virtual machines in the cloud.

5)Automation:

To automate this process, you can use scripting languages like Bash, PowerShell, or Python.

IBM Cloud CLI or SDKs for various programming languages can be helpful.

```
Import ibm_boto3
From ibm_botocore.client import CConfi
Cos_credentials = {
   “IAM_SERVICE_ID”: “YOUR_SERVICE_ID”,
   “IBM_API_KEY_ID”: “YOUR_API_KEY”,
   “ENDPOINT”: https://s3.us-south.cloud-object-storage.appdomain.cloud,
   “IBM_AUTH_ENDPOINT”: https://iam.cloud.ibm.com/identity/token,
   “BUCKET_NAME”: “your-bucket-name”
}

Cos = ibm_boto3.client(“s3”,
   Ibm_api_key_id=cos_credentials[“IBM_API_KEY_ID”],
   Ibm_service_instance_id=cos_credentials[“IAM_SERVICE_ID”],
   Ibm_auth_endpoint=cos_credentials[“IBM_AUTH_ENDPOINT”],
   Config=Config(signature_version=”oauth”),
   Endpoint_url=cos_credentials[“ENDPOINT”]
)

Virtual_server_params = {
   “name”: “my-vm”,
   “profile”: “cx2-2x4”,
   “datacenter”: “dal13”,
}

Response = cos.create_instance(virtual_server_params)

Print(“Virtual server created:”, response)
```

Conduct recovery tests to ensure that the disaster recovery plan works as intended.Simulate a disaster scenerio and practice recovery procedures.

1)Define Your Disaster Recovery Plan:

First, you need to have a clear and well-documented disaster recovery plan. This plan should outline the steps to take in the event of various disaster scenarios.

2)Set Up a Separate Test Environment:

Create a separate environment in the cloud for testing and recovery purposes. You can use infrastructure as code (IaC) tools like Terraform or AWS CloudFormation to provision this environment.

3)Data Backup and Replication:

Ensure that your data is regularly backed up and replicated to the test environment. Use cloud storage services like Amazon S3, Azure Blob Storage, or Google Cloud Storage for this purpose.

4)Simulate a Disaster Scenario:

Write code to simulate a disaster scenario. For example, you might script the deletion of critical resources, data corruption, or downtime. Be very cautious when doing this to avoid affecting your production environment.

5)Automate Testing:

Create scripts or use automation frameworks like Ansible or Puppet to orchestrate the testing process. These scripts should trigger the disaster simulation, initiate recovery, and report on the results.

6)Monitoring and Validation:

During and after the recovery process, set up monitoring and validation mechanisms to ensure that the recovery is successful. Use tools like AWS CloudWatch or Azure Monitor.

```
Import boto3

Ec2 = boto3.client('ec2', region_name='us-east-1')

Instance_id = 'i-0123456789abcdef0'  # Replace with your instance ID

Ec2.terminate_instances(InstanceIds=[instance_id])


New_instance = ec2.run_instances(
   ImageId='ami-12345678',
   InstanceType='t2.micro',
   MinCount=1,
   MaxCount=1
)
```

This is a basic example, and a real-world disaster recovery scenario may involve more complex recovery procedures, multiple services, and comprehensive monitoring and validation steps.