How can you handle Disaster Recovery for CI/CD pipeline ?

In a high-availability DevOps environment, the CI/CD pipeline must be resilient to regional outages, network failures, or service-specific disruptions. For our mission-critical applications, we designed a cross-region disaster recovery strategy for our entire CI/CD pipeline using AWS-native tools like **CloudFormation, S3 replication, Lambda, and Route 53.** Here's how we implemented it.

🏗️ **1. Cross-Region Pipeline Infrastructure via CloudFormation**

We used Infrastructure as Code (IaC) to deploy identical pipeline stacks across two AWS regions:

- Primary Region: Mumbai (ap-south-1)
- Disaster Recovery Region: Singapore (ap-southeast-1)

⚙️ Tools:

- AWS CloudFormation templates stored in Git
- CDK or CI job to deploy and update in both regions

bash

```
aws cloudformation deploy \
 --template-file pipeline.yaml \
 --region ap-south-1 \
 --stack-name ci-pipeline-primary

aws cloudformation deploy \
 --template-file pipeline.yaml \
 --region ap-southeast-1 \
 --stack-name ci-pipeline-dr
```

✓ This ensures identical infrastructure and allows for instant promotion of the DR region when needed.

---

2. Cross-Region S3 Artifact Replication

Build artifacts (e.g., .zip, .jar, Docker images) are stored in S3 and replicated to the DR region.

Setup:

- Enable **S3 Cross-Region Replication (CRR)** with versioning

- **Replicate artifacts from primary (Mumbai) to secondary (Singapore)**

```
json
{
  "Rules": [{
    "Status": "Enabled",
    "Prefix": "artifacts/",
    "Destination": {
      "Bucket": "ci-artifacts-dr",
      "StorageClass": "STANDARD"
    }
  }]
}
```

✓ Both pipelines access the same deployables — ensuring no rebuild needed in case of failover.

🧠 3. Daily Backup of CodePipeline Configuration

We wrote a Lambda function scheduled via EventBridge to export the CodePipeline structure as JSON daily:

```
python
import boto3, json

client = boto3.client('codepipeline')

def lambda_handler(event, context):
    pipelines = client.list_pipelines()['pipelines']
    for pipeline in pipelines:
        name = pipeline['name']
        definition = client.get_pipeline(name=name)
        s3 = boto3.client('s3')
        s3.put_object(
            Bucket='pipeline-backups',
            Key=f'{name}.json',
            Body=json.dumps(definition)
        )
```

✓ This provides recoverability in case of accidental deletion or corruption.

🌐 4. Health Check & Route 53 Failover

We added **a /health endpoint on our CI/CD orchestration service** (custom API Gateway or webhook manager).

Setup:

- Route 53 health checks monitor /health
- If primary pipeline API fails beyond a threshold:
    - DNS automatically routes to Singapore-based pipeline

✓ Ensures zero manual intervention for switching to DR.

🧪 5. Monthly DR Drills (Failover Simulation)

To validate DR readiness, we run automated simulation drills using Lambda:

python
# Simulates Mumbai outage
def lambda_handler(event, context):
  # Disable primary (simulate failure)
  # Trigger DR pipeline
  boto3.client('codepipeline', region_name='ap-southeast-1')\
    .start_pipeline_execution(name='ci-pipeline-dr')

DR Drill Steps:

- Triggered monthly via EventBridge cron
- Verifies pipeline readiness and DR deployment functionality
- Logs success/failure in CloudWatch Logs

✓ **Helps meet audit and compliance standards (e.g., SOC 2, ISO 27001).**

Mumbai Outage Recovery

During a connectivity disruption in ap-south-1, our health check failed, triggering:

- Route 53 failover
- Singapore-based pipeline started automatically
- Hotfix deployment went live without any human intervention

✅ **Result: Zero downtime, no deployment rollback, and 100% confidence in our DR workflow.**

Summary of Key DR Components

Component	Tool Used	Purpose
Pipeline Infra	CloudFormation	Cross-region reproducibility
Artifact Replication	S3 CRR	Shared artifacts in DR
Config Backups	Lambda + S3	Recover CodePipeline structure
DNS Failover	Route 53	Auto switch to DR region
DR Simulation	Lambda	Monthly failover drills
Logs & Alerts	CloudWatch, SNS	Monitor drill success and failures

---

🧠 **Takeaway**

"Disaster recovery for CI/CD is not just about backup—it's about continuous readiness, automation, and zero manual reliance. With this cross-region setup, our pipelines are resilient, auditable, and capable of surviving regional AWS failures without affecting release timelines."

Multi-Region Pipeline Availability Strategy

🔁 **1. Active-Passive Architecture**

- Primary region (e.g., Mumbai) handles builds/deployments.
- Secondary region (e.g., Singapore) is standby for failover.
- Identical infra in both via CloudFormation/CDK.

🔄 **2. Sync Across Regions**

- Artifacts: S3 Cross-Region Replication
- Build metadata: DynamoDB Global Tables
- Config: SSM Parameter Store or GitOps
- **Release tags: From source control (Git)**

🌍 **3.** Region-Specific Builds (Active-Active)

- us-east-1: Frontend (US team)
- us-west-2: Backend (India team)
- eu-west-1: Europe product builds
- Reduces latency, avoids region congestion.

🧰 **Tools Used**

- CDK/CloudFormation: Infra setup
- S3: Artifact storage + CRR
- DynamoDB Global Tables: Metadata
- Route 53: Health checks
- EventBridge + Lambda: Triggers
- CodeBuild: Scalable compute
- Regional CodePipeline: Fast routing

Real-Life Outcome

- Separate build start for US team
- 40% faster backend builds
- DR drills passed with no regression

Backup and Restore of Pipeline Configs

♻️ **1. Nightly Backups via Lambda**

- Lambda runs nightly via EventBridge.
- Uses get-pipeline to save JSON config to S3.
- S3 versioning enabled.

🛡️ **2. Parameter Store Backups**

- Fetches SSM Parameter Store values.
- Stores them in S3 for environment recovery.

🏷️ **3. Git Commit Tagging**

- S3 object tagging with commit ID and timestamp.
- Useful for audit, rollback, traceability.

🧩 **4. One-Command Restore**

- Use update-pipeline CLI to restore JSON.
- Restore SSM parameters via put-parameter.

🔁 **5. Weekly Restore Drill**

- Restore to staging every Friday.
- Run build to validate integrity.
- Logs + SNS alerts if it fails.

🧪 **Real Incident Recovery**

- Production pipeline accidentally deleted.
- Restored in 15 mins from last night's backup.
- No impact, no manual effort.

🧰 **Tools Used**

- Lambda + CodePipeline API: Backup logic
- S3 + Versioning: Backup storage
- SSM Parameter Store: Runtime config
- update-pipeline CLI: Fast restore
- EventBridge: Scheduled tasks
- CloudWatch Logs + SNS: Alerts
-

What is the role of the secondary region in an Active-Passive multi-region pipeline setup?

A) Load balancing
B) Build scheduling
C) Failover during primary region failure
D) Backup storage only

✔ **Answer: C**

What is the benefit of region-specific builds in an Active-Active setup?

A) Reduces compute cost
B) Ensures all builds happen in one region
C) Minimizes latency and avoids congestion
D) Uses less secure routing

✔ **Answer: C**