AWS Disaster Recovery

Ransomware-resistant.
Compliance-ready.
IaC Terraform.

ISO 27001 • DORA • SOC2 • NIS2 • KRITIS

Addressing

Regulatory Requirements

ISO 27001:2022

- A.5.30: ICT Readiness for Business Continuity
- A.8.13: Information Backup
- A.8.14: Redundancy of Processing Facilities

NIS2 Directive (EU)

- Article 21: Backup & Disaster Recovery
- "Multi-layered" Cybersecurity Approach

DORA (Financial Sector)

- Articles 11-12: Backup & Recovery
- Geographic Separation mandatory

02

Finding

…the Right Strategy

Compliance frameworks demand geographic separation and point-in-time recovery capabilities.

Ransomware protection demands isolation.

Not all DR strategies deliver both.

Let's compare the most common AWS disaster recovery approaches — and see which ones actually protect your data and satisfy your auditors.

Simple Example

Let’s have a look why Multi-AZ is not enough

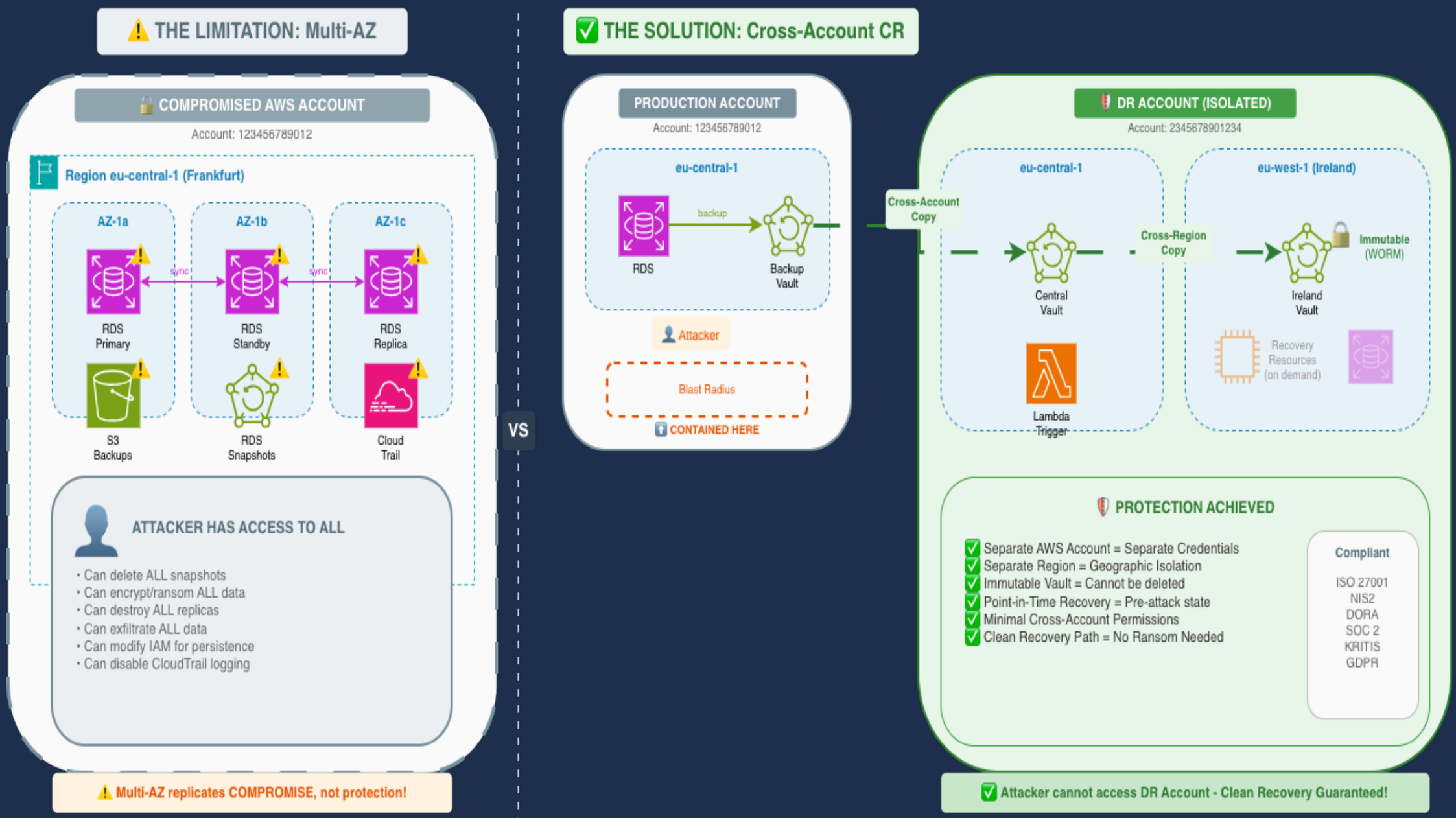
This Article describes the CACR (Cross-Account Cross-Region) - a disaster recovery architecture that goes beyond standard AWS high-availability solutions. It addresses threat scenarios against which Multi-AZ, and even Multi-Region within a single AWS account, provide no protection.

The core insight: In a security incident involving compromised credentials, your entire AWS account becomes the blast radius. Multi-AZ replicates your data across availability zones - but it also replicates attacker access. True resilience requires account isolation.

Cross Account and Cross Region (CACR)

Let’s compare Multi-AZ with CACR

Multi-AZ vs Cross-Account Cross-Region DR



Cross Account and Cross Region (CACR)

Disaster Recovery Compliance Matrix

All necessary regulatory requirements are fulfilled:

Framework	Requirement	CACR Addresses	Status
ISO 27001:2022 A.5.30	ICT Readiness for Business Continuity Redundancy of information processing facilities	Cross-Account isolation + Cross-Region redundancy Automated backup with defined RTO/RPO	✅
ISO 27001:2022 A.8.13	Information Backup Backup copies maintained and regularly tested	Daily automated backups via AWS Backup Terraform IaC for repeatable recovery tests	✅
NIS2 Directive Article 21	Backup, disaster recovery, crisis management Multi-layered cybersecurity approach	3-tier backup (Local → Central → Ireland) Account isolation as security layer	✅
DORA Articles 11-12	ICT risk management, backup policies **Geographic diversity** of backup locations	Frankfurt → Ireland (Cross-Region) Immutable vault with WORM compliance	✅
SOC 2 Type II CC7.4 / CC7.5	Recovery procedures tested Backup data stored **separately**	One-command recovery (terraform apply) Separate AWS account = air-gapped storage	✅
KRITIS / BSI IT-Grundschutz	Critical infrastructure protection Resilience against cyber attacks	Ransomware-resistant architecture KMS encryption + account isolation	✅
GDPR Article 32	Ability to restore availability and access to personal data in timely manner	RTO: ~20 minutes I RPO: < 24 hours Documented recovery procedures	✅

✅ DR Compliance Checklist

- ☑ Backups in separate AWS Account
- ☑ Geographic redundancy (Cross-Region)
- ☑ Immutable backups (Vault Lock / WORM)
- ☑ Encryption at rest (KMS CMK)
- ☑ Defined RTO (< 4 hours)
- ☑ Defined RPO (< 24 hours)
- ☑ Automated backup schedule
- ☑ Documented recovery procedures
- ☑ Infrastructure as Code (Terraform)
- ☑ Regular recovery testing
- ☑ Minimal cross-account permissions
- ☑ Audit logging (CloudTrail)

12/12

🎯 Key Compliance Principles Addressed by CACR Architecture

Account Isolation | Geographic Diversity | Immutable Storage | Encryption (KMS) | Defined RTO/RPO | Automated Backups | IaC (Terraform) | Testable Recovery | Audit Logging | Cost Efficient

ISO 27001 | NIS2 | DORA | SOC 2 | KRITIS | GDPR

The Problem:

Limitations of Traditional HA Solutions

What Multi-AZ Protects Against:

Hardware failure in one AZ	✅ Yes
Network outage in one AZ	✅ Yes
Local natural disaster	✅ Yes
Maintenance windows	✅ Yes
Power outage in one datacenter	✅ Yes

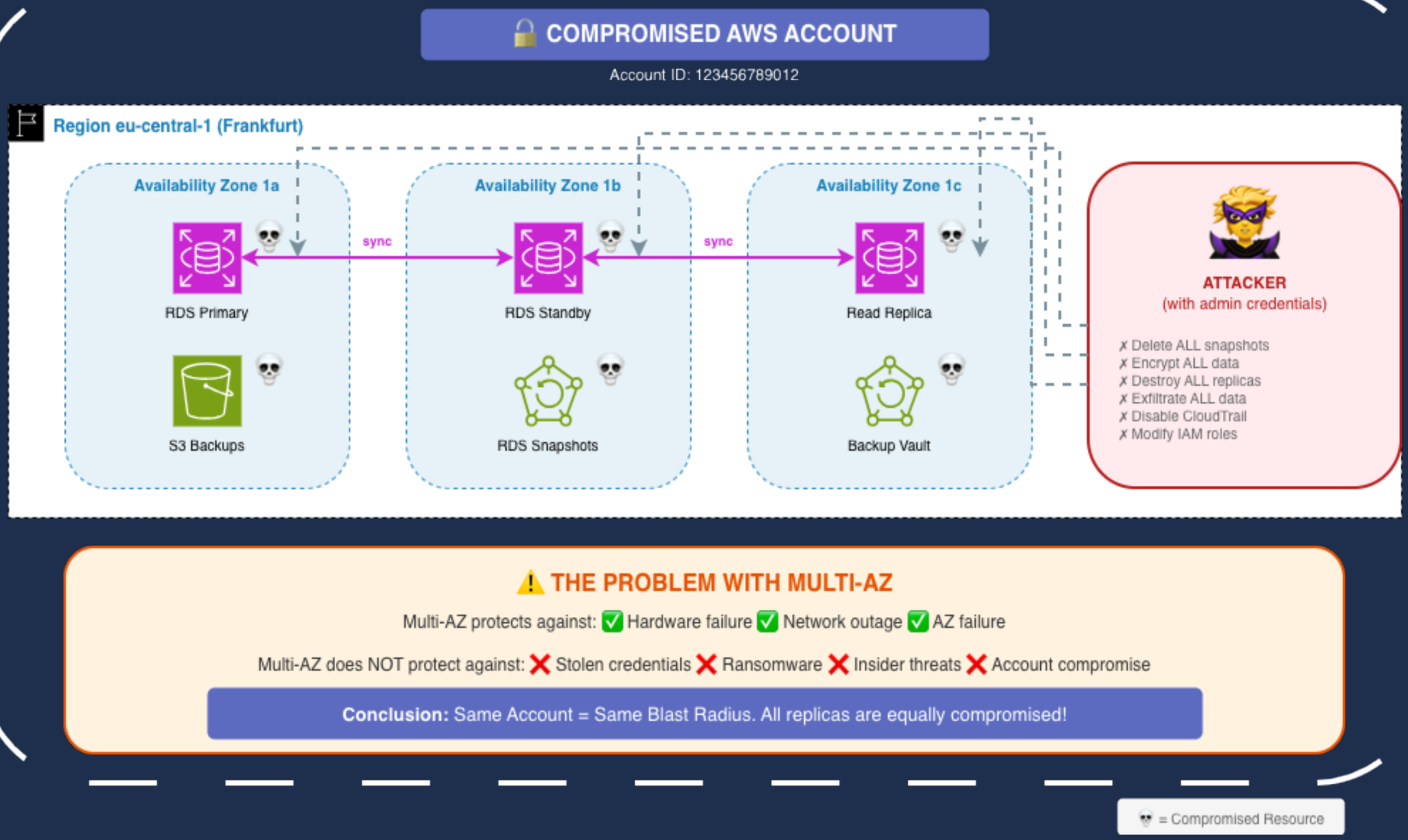
What Multi-AZ does not Protects Against:

Compromised AWS credentials	❌ No
Ransomware with admin access	❌ No
Malicious insider threat	❌ No
AWS account suspension/compromise	❌ No
Supply chain attack on CI/CD	❌ No
Sophisticated APT with persistence	❌ No
Regional AWS service outage	❌ No

The Uncomfortable Truth

When an attacker gains administrative access to your AWS account:

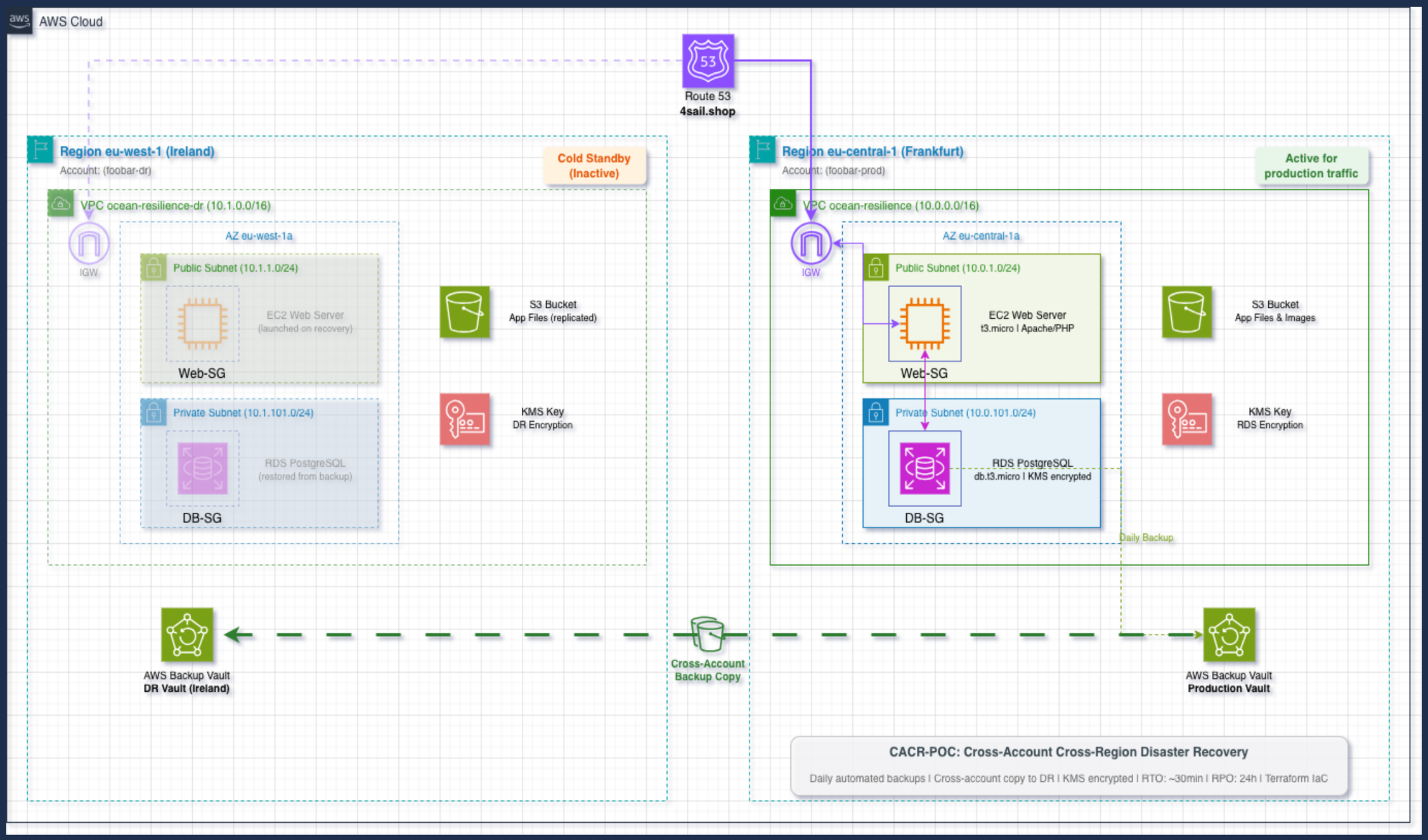
⚠️ Multi-AZ: Same Account = Same Blast Radius



The Solution:

Account Isolation Through CACR Backup and Restore

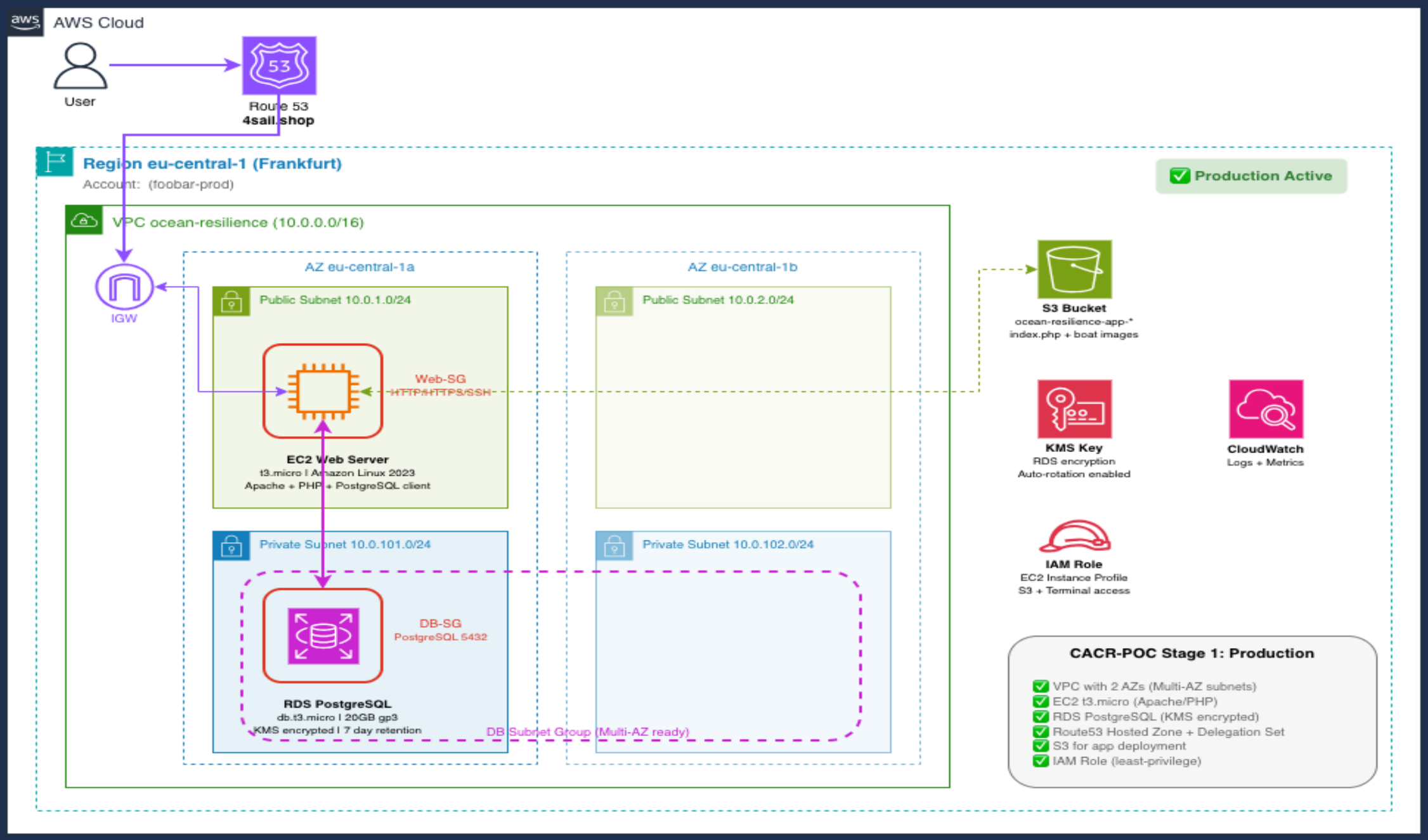
Architecture Overview:



*Requires two AWS Accounts and AWS Organizations. DR Account must be invited

Let’s create a POC:*

A simple Webserver and an RDS Database which serves as backend (PostgreSQL)



*Presuming two AWS Accounts with AWS Organizations enabled

How to setup

We choose IaC with Terraform to deploy easily and repeatable

Just simple modules*

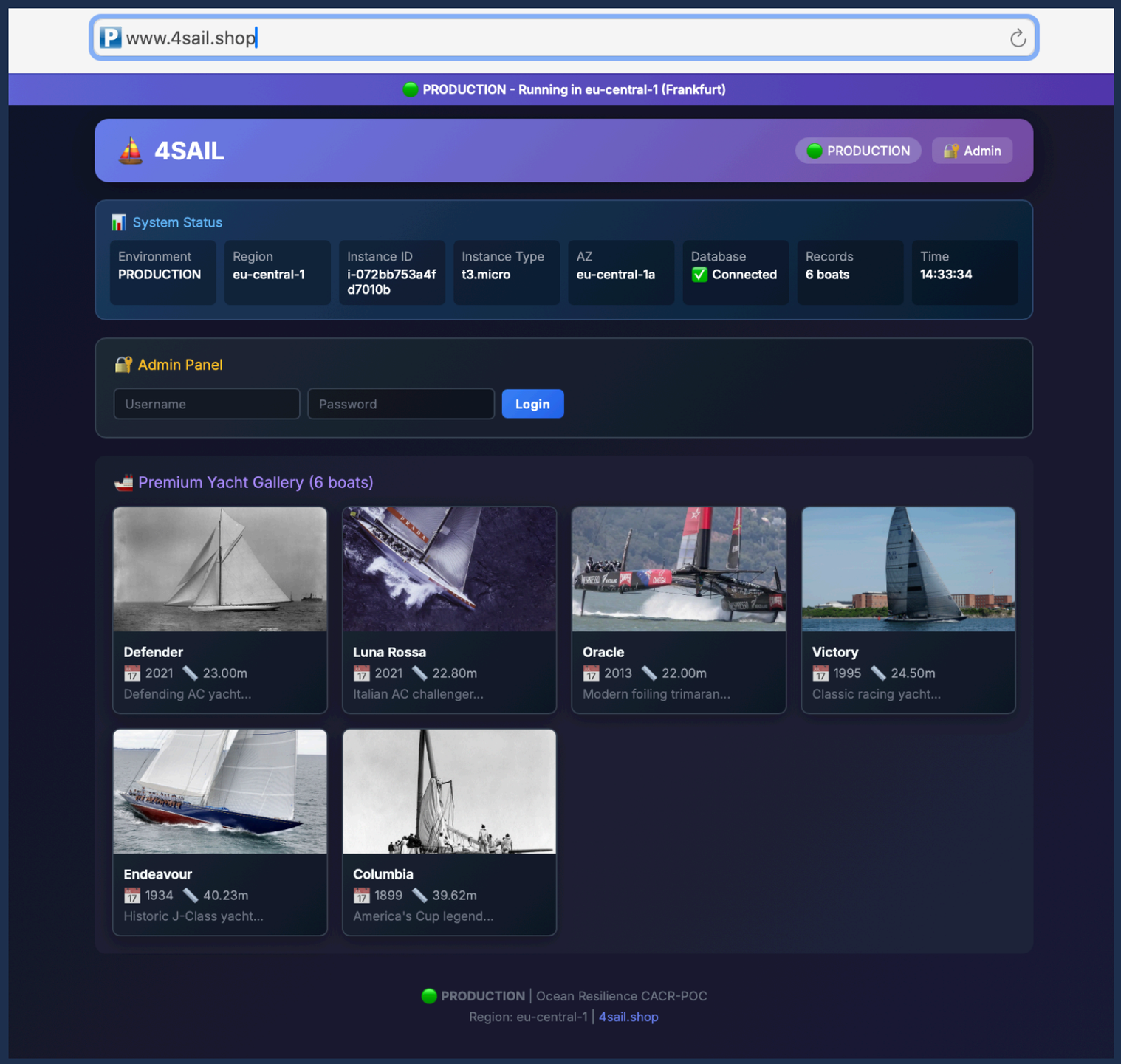
```
[devrandom:~/AWS-Projects/CACR-POC/foobar-prod #tree                                ]
.
├── app-files
│   ├── images
│   │   ├── columbia.jpg
│   │   ├── defender.jpg
│   │   ├── endeavour.jpg
│   │   ├── luna-rossa.jpg
│   │   ├── oracle.jpg
│   │   └── victory.jpg
│   └── index.php
├── backend.tf
├── backup-vault-policy-prod.tf
├── backup.tf
├── compute.tf
├── database.tf
├── DEPLOYMENT-GUIDE.md
├── dns.tf
├── dr-route53.tf
├── eventbridge.tf
├── index.php
├── kms-cross-account.tf
├── main.tf
├── networking.tf
├── outputs.tf
├── security.tf
├── stage-0
│   └── main.tf
├── storage.tf
├── terraform.tfvars
├── user-data.sh
└── variables.tf

4 directories, 27 files
devrandom:~/AWS-Projects/CACR-POC/foobar-prod #
```

*a real world scenario will be more complex.

The Result

A small website was created via „terraform apply“

We can see that we are currently operating in eu-central-1 (Frankfurt)



The website is reachable via Route53, the RDS DB endpoint is in eu-central-1

Key Points

Define the backup-plan, role-policy and DR-backup-vault-access-policy

```
devrandom:~ #aws backup get-backup-plan --backup-plan-id b8ce6795-e3d7-4732-81c0-7bc4a1383bcd --region eu-central-1 --profile foobar-prod --qu
ery 'BackupPlan.Rules[0].{
    Rule: RuleName,
    Schedule: ScheduleExpression,
    TargetVault: TargetVaultName,
    CopyTo: CopyActions[0].DestinationBackupVaultArn,
    Retention: Lifecycle.DeleteAfterDays,
    CopyRetention: CopyActions[0].Lifecycle.DeleteAfterDays
}' | jq -r 'to_entries[] | "\(.key): \(.value)"'
Rule: daily-1am-backup
Schedule: cron(0 1 * * ? *)
TargetVault: null
CopyTo: arn:aws:backup:eu-west-1:37[illegible]8:backup-vault:ocean-resilience-dr-vault-ireland
Retention: 3
CopyRetention: 17
devrandom:~ #aws iam get-role-policy --role-name ocean-resilience-backup-role --policy-name ocean-resilience-backup-cross-account --profile fo
obar-prod --query 'PolicyDocument.Statement[0].{
    Sid: Sid,
    Effect: Effect,
    Actions: join(`, `, Action),
    Resource: Resource
}' | jq -r 'to_entries[] | "\(.key): \(.value)"'
Sid: AllowCopyToBackupVault
Effect: Allow
Actions: backup:CopyIntoBackupVault, backup:DescribeBackupVault, backup:StartCopyJob
Resource: arn:aws:backup:*:371[illegible]:backup-vault/*
devrandom:~ #aws backup get-backup-vault-access-policy --backup-vault-name ocean-resilience-dr-vault-ireland --region eu-west-1 --profile foob
ar-dr --output json | jq -r '.Policy' | jq -r '.Statement[0] | {
    Sid: .Sid,
    Effect: .Effect,
    Principal: .Principal.AWS,
    Action: .Action
}' | jq -r 'to_entries[] | "\(.key): \(.value)"'
Sid: AllowSameAccountCrossRegionCopy
Effect: Allow
Principal: arn:aws:iam::371[illegible]:root
Action: backup:CopyIntoBackupVault
devrandom:~ #
```

The backup plan will be executed over night…

DR Account

Some small things have to be setup in the DR Account and it’s Region, such as a backup vault.

```
devrandom:~ #aws backup get-backup-plan --backup-plan-id b8ce6795-e3d7-4732-81c0-7bc4a1383bcd --region eu-central-1 --profile foobar-prod --qu
ery 'BackupPlan.Rules[0].{
    Rule: RuleName,
    Schedule: ScheduleExpression,
    TargetVault: TargetVaultName,
    CopyTo: CopyActions[0].DestinationBackupVaultArn,
    Retention: Lifecycle.DeleteAfterDays,
    CopyRetention: CopyActions[0].Lifecycle.DeleteAfterDays
}' | jq -r 'to_entries[] | "\(.key): \(.value)"'
Rule: daily-1am-backup
Schedule: cron(0 1 * * ? *)
TargetVault: null
CopyTo: arn:aws:backup:eu-west-1:37[illegible]8:backup-vault:ocean-resilience-dr-vault-ireland
Retention: 3
CopyRetention: 17
devrandom:~ #aws iam get-role-policy --role-name ocean-resilience-backup-role --policy-name ocean-resilience-backup-cross-account --profile fo
obar-prod --query 'PolicyDocument.Statement[0].{
    Sid: Sid,
    Effect: Effect,
    Actions: join(`, `, Action),
    Resource: Resource
}' | jq -r 'to_entries[] | "\(.key): \(.value)"'
Sid: AllowCopyToBackupVault
Effect: Allow
Actions: backup:CopyIntoBackupVault, backup:DescribeBackupVault, backup:StartCopyJob
Resource: arn:aws:backup:*:371[illegible]:backup-vault/*
devrandom:~ #aws backup get-backup-vault-access-policy --backup-vault-name ocean-resilience-dr-vault-ireland --region eu-west-1 --profile foob
ar-dr --output json | jq -r '.Policy' | jq -r '.Statement[0] | {
    Sid: .Sid,
    Effect: .Effect,
    Principal: .Principal.AWS,
    Action: .Action
}' | jq -r 'to_entries[] | "\(.key): \(.value)"'
Sid: AllowSameAccountCrossRegionCopy
Effect: Allow
Principal: arn:aws:iam::371[illegible]:root
Action: backup:CopyIntoBackupVault
devrandom:~ #
```

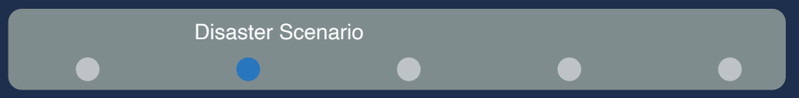
This is called a cold standby environment. No compute resources are up

Checking the backup plan next day

Let’s see if the backup-plan copied the RDS DB backup over to Ireland

```
devrandom:~ #echo "BACKUP JOBS:"
aws backup list-backup-jobs --region eu-central-1 --profile foobar-prod --max-results 1 --query "BackupJobs[0].[CreationDate,State,BackupVaultName]"
 --output table

echo -e "\n COPY JOBS:"
aws backup list-copy-jobs --region eu-central-1 --profile foobar-prod --max-results 1 --query "CopyJobs[0].[CreationDate,State,DestinationBackupVaul
tArn]" --output table

echo -e "\n RECOVERY POINTS in (Ireland):"
[aws backup list-recovery-points-by-backup-vault --backup-vault-name ocean-resilience-dr-vault-ireland --region eu-west-1 --profile foobar-dr --max-r]
esults 1 --query "RecoveryPoints[0].[CreationDate,Status]" --output table
BACKUP JOBS:
-------------------------------------
|          ListBackupJobs           |
+-----------------------------------+
|  2026-01-16T02:00:00+01:00        |
|  COMPLETED                        |
|  ocean-resilience-production-vault|
+-----------------------------------+

 COPY JOBS:
-----------------------------------------------------------------------------------------
|                                    ListCopyJobs                                       |
+---------------------------------------------------------------------------------------+
|  2026-01-16T02:30:49.632000+01:00                                                     |
|  COMPLETED                                                                            |
|  arn:aws:backup:eu-west-1:371[redacted]:backup-vault:ocean-resilience-dr-vault-ireland|
+---------------------------------------------------------------------------------------+

 RECOVERY POINTS in (Ireland):
--------------------------------
|ListRecoveryPointsByBackupVault|
+--------------------------------+
|  2026-01-16T02:00:00+01:00     |
|  COMPLETED                     |
+--------------------------------+
devrandom:~ #
```

Perfect! So we got an recovery point in eu-west-1 in the DR account



Now the worst case happens!

Compromised data!

Possible Scenarios:

- Compromised AWS credentials
- Ransomware with admin access
- Malicious insider threat
- Accidental account-wide deletion
- AWS account suspension/compromise
- Supply chain attack on CI/CD
- Sophisticated APT with persistence
- Regional AWS service outage

How can we get back to work BUSINESS CRITICAL WORKLOADS by keeping the possibility to perform a deep attack path analysis in the compromised environment?

Simple Restore with traditional DR under Time Pressure?

Why not:

- Evidence destroyed during recovery
- Rush job = missed IOCs (Indicators of Compromise)
- Attacker persistence may survive hasty remediation
- No time for proper root cause analysis

So, let’s have a closer look to the possibilities with CACR and the Attack Path Analysis

*Attack Path Analysis (APA)**

Overview Traditional DR and CACR

The CACR Advantage: Forensics-First Recovery

Traditional DR (Single Account)

COMPROMISED ACCOUNT
Account: 123***

↓

Rush Restore
Evidence destroyed

↓

Blind Recovery
Attacker may persist
No forensic analysis
Risk of re-infection

Time pressure prevents proper investigation

CACR DR (Cross-Account)

COMPROMISED ACCOUNT
Account: 123***

↓

Freeze Production Account
Preserve CloudTrail evidence

↓

DR Account Active (234***)
Business operations continue
No time pressure

↓

CloudTrail Analysis (Parallel)
- Initial Access - How did they get in?
- Privilege Escalation - How gain control?
- Defense Evasion - What did they hide?
- Exfiltration - What was stolen?

Forensics-first approach enables complete attack understanding

↓

Clean Rebuild
Attack path eliminated
Attacker persistence removed

Different situations may require different reactions in a disaster scenario.

But anyway: the Recovery Point Objective (RPO) has to be defined!

CloudTrail Analysis

Cloudtrail logs are the AWS sources

Key Attack Path Categories

Analyze CloudTrail logs to reconstruct attack path

Initial Access

ConsoleLogin
PasswordRecoveryRequested
GetPasswordData

Privilege Escalation

CreateRole
PutRolePolicy
AddUserToGroup

Defense Evasion

StopLogging
DeleteTrail
DeleteFlowLogs

Exfiltration

CreateSnapshot
SharedSnapshotCopy
GetSecretValue

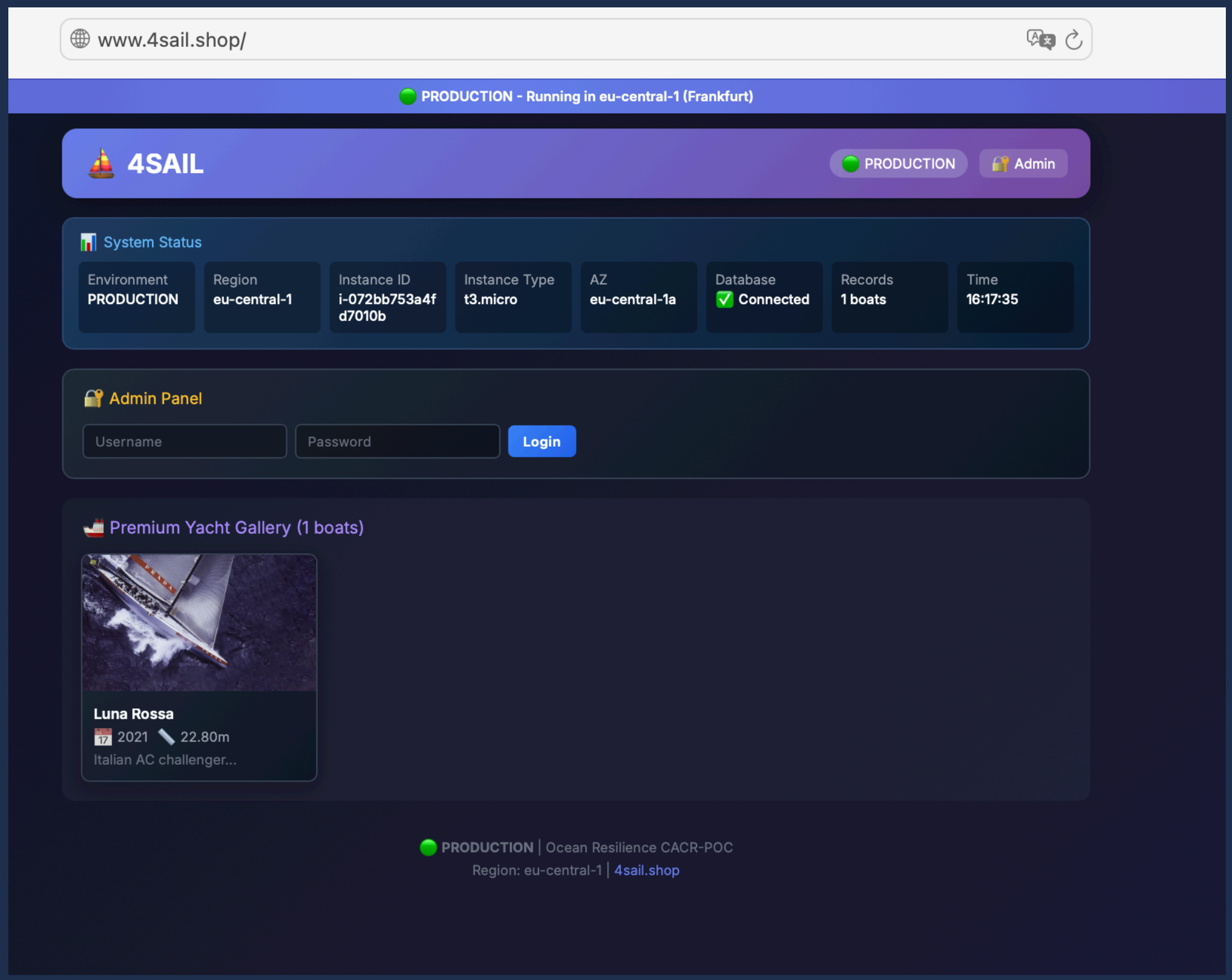
Result: Clean recovery with eliminated persistence

Detailed methods of APA goes beyond this POC

The Disaster

This is just a simulation

But let’s imagine data in the RDS Database is lost. Most boat data is gone:



We even noticed, that all backup-data in eu-central-1 has been deleted or compromised!

Disaster Recovery

Terraform IaC has been prepared and luckily tested upfront during regular disaster simulations

```
[devrandom:~/AWS-Projects/CACR-POC/foobar-dr/dr-recovery #tree
.
├── app-files
│   └── index.php
├── backend.tf
├── bin
├── compute.tf
├── database-restore.tf
├── dns-failover.tf
├── dr-recovery-helper.sh
├── main.tf
├── networking.tf
├── README.md
├── secrets.auto.tfvars
├── secrets.auto.tfvars.template
├── security.tf
├── storage.tf
├── terraform.tfvars
├── terraform.tfvars.bak
├── terraform.tfvars.template
├── user-data.sh
└── variables.tf

3 directories, 18 files
devrandom:~/AWS-Projects/CACR-POC/foobar-dr/dr-recovery #
```

This will recover the Workload in the DR Account in the DR Region

Choosing the recovery point

We configure now the recovery point with a small helper script

This script writes the recovery point ARN into the terraform.tfvars

```
[devrandom:~/AWS-Projects/CACR-POC/foobar-dr/dr-recovery #./dr-recovery-helper.sh

   🚨  DR RECOVERY POINT HELPER  🚨

   Ocean Resilience - Cross-Account Cross-Region DR


📦 Ireland DR Vault (for Recovery)
   Vault: ocean-resilience-dr-vault-ireland | Region: eu-west-1

   1 recovery points found:

   IDX    CREATED              STATUS
   ------------------------------------------------------------
   [0]  2026-01-16 02:00:00  COMPLETED
        arn:aws:rds:eu-west-1:37[illegible]3:snapshot:awsbackup:copyjob-da84b655-c6c7-862f-8903-da49d0e7b657


🎯 SELECT RECOVERY POINT

   Enter index (0 = newest, q = quit): 0

✅ Selected:

   arn:aws:rds:eu-west-1:37[illegible]3:snapshot:awsbackup:copyjob-da84b655-c6c7-862f-8903-da49d0e7b657

   📋 Copied to clipboard!

📝 What would you like to do?

   [1] Write to terraform.tfvars
   [2] Run terraform plan
   [3] Run terraform apply
   [q] Exit

   Selection: 1
✅ terraform.tfvars updated:
   recovery_point_arn = "arn:aws:rds:eu-west-1:37[illegible]3:snapshot:awsbackup:copyjob-da84b655-c6c7-862f-8903-da49d0e7b657"

   Backup: ./terraform.tfvars.bak
devrandom:~/AWS-Projects/CACR-POC/foobar-dr/dr-recovery #
```

Let’s proceed to check everything prior to restore…

Checking the terraform plan

Let’s double-check if we are in line with our recovery plan and have a closer look to the outputs of this terraform plan

```
+ app_bucket_name         = (known after apply)
+ dns_failover_command    = (known after apply)
+ dns_status              = (known after apply)
+ dr_info                 = <<-EOT
      ┌──────────────────────────────────────────┐
      │           DR RECOVERY STATUS             │
      ├──────────────────────────────────────────┤
      │ DR Account:        3717[redacted]
      │ DR Region:         eu-west-1
      │ DR Vault:          ocean-resilience-dr-vault-ireland
      │ Vault ARN:         arn:aws:backup:eu-west-1:371[redacted] backup-vault:ocean-resilience-dr-vault-ireland
      │
      │ Production Account: 567[redacted]
      │ Domain:            4sail.shop
      │ Hosted Zone ID:    Z045316260B6HTHDH3BC (automatically discovered ✓)
      ├──────────────────────────────────────────┤
      │ Recovery Point:    arn:aws:rds:eu-west-1:3717[redacted] snapshot:awsbackup:copyjob-da84b655-c6c7-862f-8903-da49d0e7b657
      │ Discovery Mode:    INACTIVE - Recovery running!
      │ DNS Update:        YES - DNS will be updated!
      ├──────────────────────────────────────────┤
      │
      │ LIST RECOVERY POINTS (execute locally):
      │
      │ aws backup list-recovery-points-by-backup-vault \
      │   --backup-vault-name ocean-resilience-dr-vault-ireland \
      │   --region eu-west-1 \
      │   --profile foobar-dr \
      │   --query 'RecoveryPoints[*].[RecoveryPointArn,CreationDate,Status]' \
      │   --output table
      │
      └──────────────────────────────────────────┘
  EOT
+ dr_website_url          = (known after apply)
+ ec2_instance_id         = (known after apply)
+ ec2_public_ip           = (known after apply)
+ hosted_zone_id          = "Z045316260B6HTHDH3BC"
+ public_subnet_ids       = [
    + (known after apply),
    + (known after apply),
  ]
+ recovery_point_arn      = "arn:aws:rds:eu-west-1:37[redacted]:snapshot:awsbackup:copyjob-da84b655-c6c7-862f-8903-da49d0e7b657"
+ restored_db_address     = (known after apply)
+ restored_db_endpoint    = (known after apply)
+ restored_db_identifier  = (known after apply)
+ vault_arn               = "arn:aws:backup:eu-west-1:37[redacted]:backup-vault:ocean-resilience-dr-vault-ireland"
+ vpc_id                  = (known after apply)

────────────────────────────────────────────────────────────────────────────

Note: You didn't use the -out option to save this plan, so Terraform can't guarantee to take exactly these actions if you run "terraform apply" now.
devrandom:~/AWS-Projects/CACR-POC/foobar-dr/dr-recovery #
```

Everything seems fine. Webserver, RDS Database and Route53 will be created

Executing the terraform plan

Now, we will perform the disaster recovery (terraform apply)

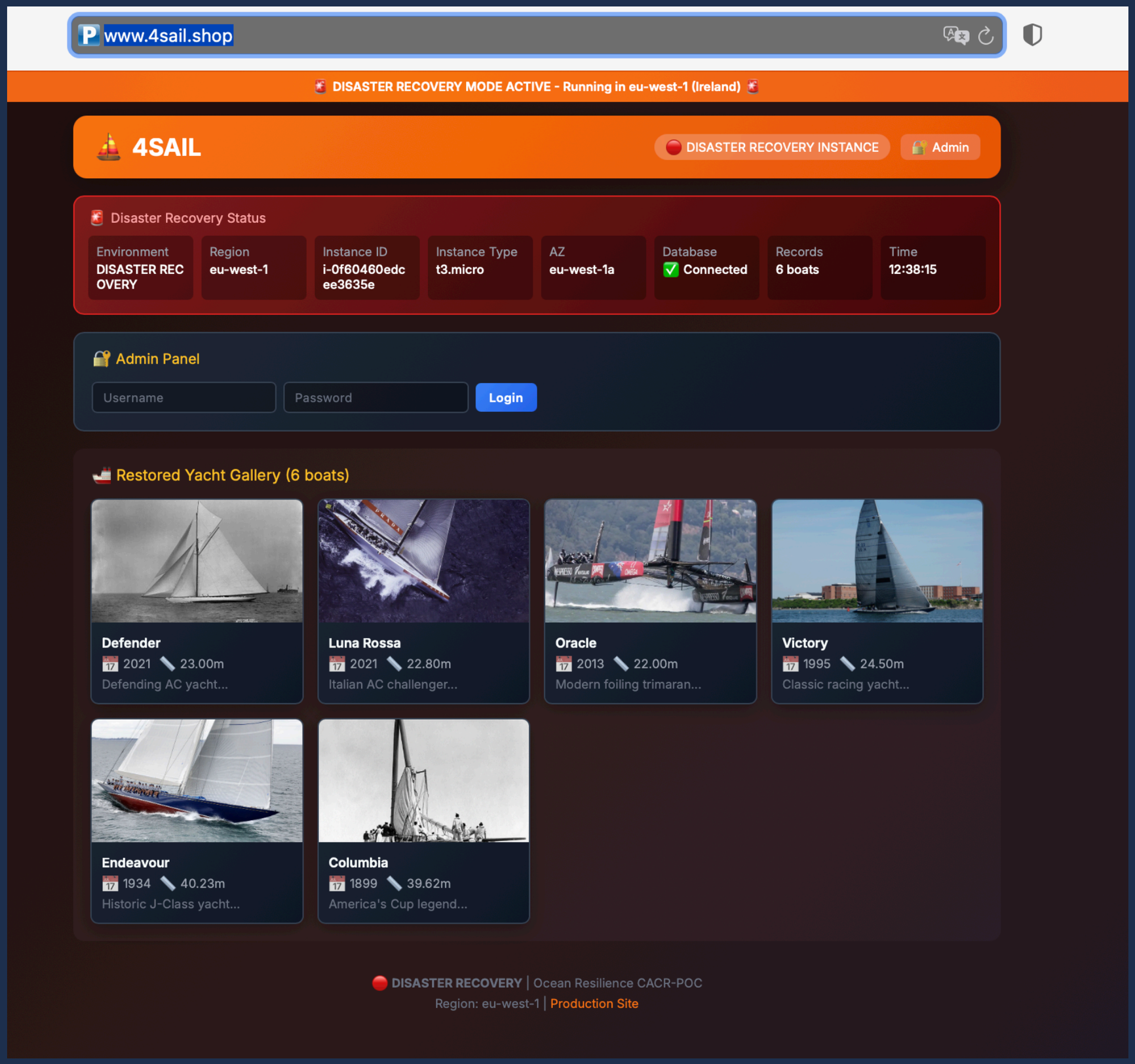
This will take some minutes …

```
    ]
recovery_point_arn = "arn:aws:rds:eu-west-1:371[illegible]:snapshot:awsbackup:copyjob-da84b655-c6c7-862f-8903-da49d0e7b657"
restored_db_address = "ocean-resilience-dr-db-20260119121243.cnqok6ckiiw4.eu-west-1.rds.amazonaws.com"
restored_db_endpoint = "ocean-resilience-dr-db-20260119121243.cnqok6ckiiw4.eu-west-1.rds.amazonaws.com:5432"
vault_arn = "arn:aws:backup:eu-west-1:371[illegible]:backup-vault:ocean-resilience-dr-vault-ireland"
dns_failover_command = <<-EOT
        # Check DNS propagation:
        dig 4sail.shop +short

        # Should show: 52.18.168.27

        # Check DR Status TXT Record:
        dig _dr-status.4sail.shop TXT +short
    EOT
dns_status = "✅ DNS FAILOVER ACTIVE - 4sail.shop points to 52.18.168.27"
ec2_public_ip = "52.18.168.27"
app_bucket_name = "ocean-resilience-dr-app-3717[illegible]"
ec2_instance_id = "i-0f60460edcee3635e"
restored_db_identifier = "ocean-resilience-dr-db-20260119121243"
app_bucket_arn = "arn:aws:s3:::ocean-resilience-dr-app-371[illegible]"
dr_info = <<-EOT

        ╔═══════════════════════════════════════════════════════════════════════════╗
                                DR RECOVERY STATUS
        ╠═══════════════════════════════════════════════════════════════════════════╣
          DR Account:        371[illegible]
          DR Region:         eu-west-1
          DR Vault:          ocean-resilience-dr-vault-ireland
          Vault ARN:         arn:aws:backup:eu-west-1:3717[illegible]backup-vault:ocean-resilience-dr-vault-ireland

          Production Account: 563[illegible]
          Domain:            4sail.shop
          Hosted Zone ID:    Z045316260B6HTHDH3BC (automatically discovered ✓)
        ╠═══════════════════════════════════════════════════════════════════════════╣
          Recovery Point:    arn:aws:rds:eu-west-1:371[illegible]:snapshot:awsbackup:copyjob-da84b655-c6c7-862f-8903-da49d0e7b657
          Discovery Mode:    INACTIVE - Recovery running!
          DNS Update:        YES - DNS will be updated!
        ╠═══════════════════════════════════════════════════════════════════════════╣

          LIST RECOVERY POINTS (execute locally):

          aws backup list-recovery-points-by-backup-vault \
            --backup-vault-name ocean-resilience-dr-vault-ireland \
            --region eu-west-1 \
            --profile foobar-dr \
            --query 'RecoveryPoints[*].[RecoveryPointArn,CreationDate,Status]' \
            --output table

        ╚═══════════════════════════════════════════════════════════════════════════╝

    EOT
hosted_zone_id = "Z045316260B6HTHDH3BC"
devrandom:~/AWS-Projects/CACR-POC/foobar-dr/dr-recovery #
```

Webserver, RDS Database and DNS Records have been created successfully

Restore success

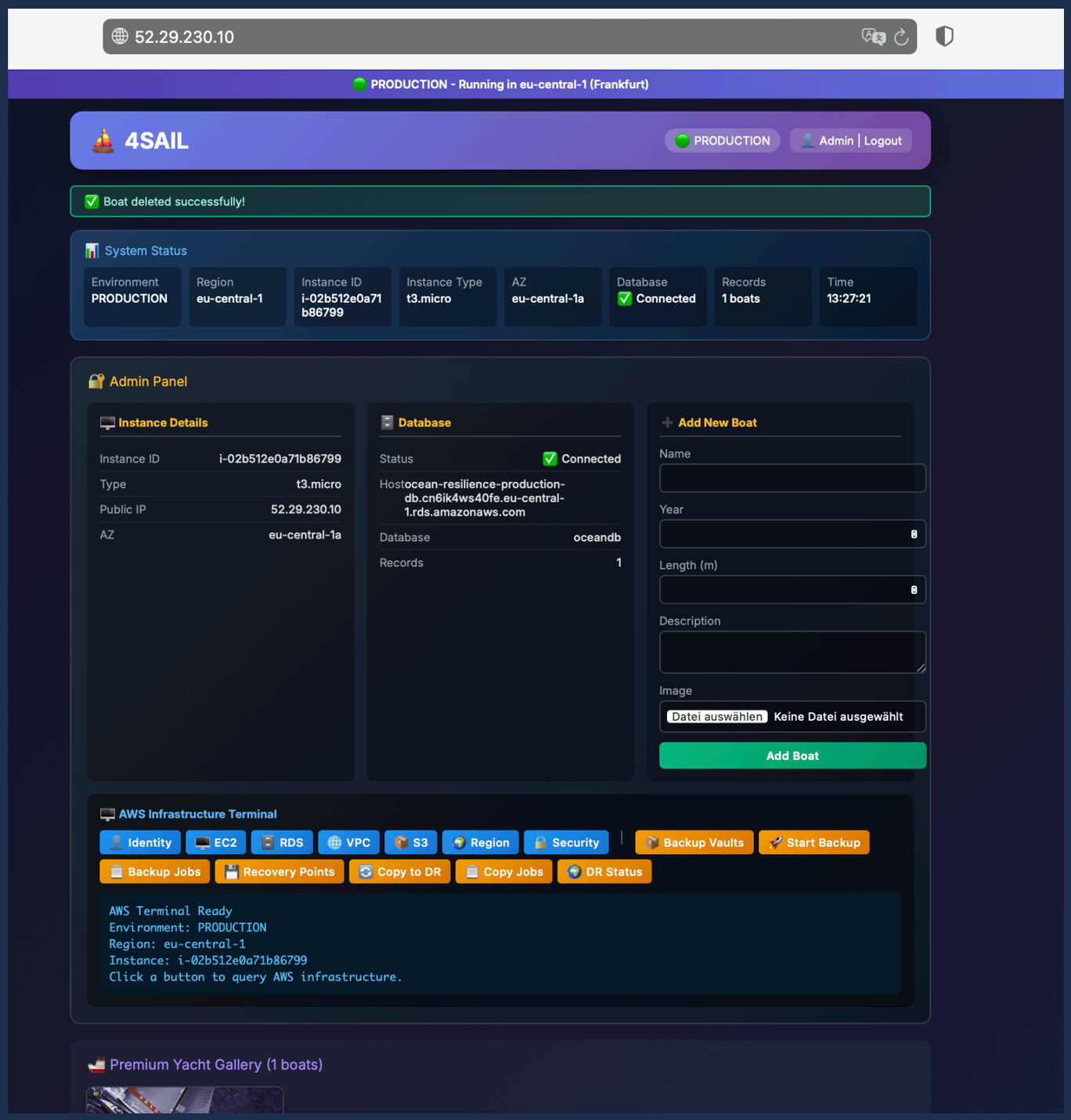
Our website is up again. It took just a few minutes



Note that the complete Workflow in now located in Region eu-west-1

Forensics

And the compromised Environment is still running in eu-central-1

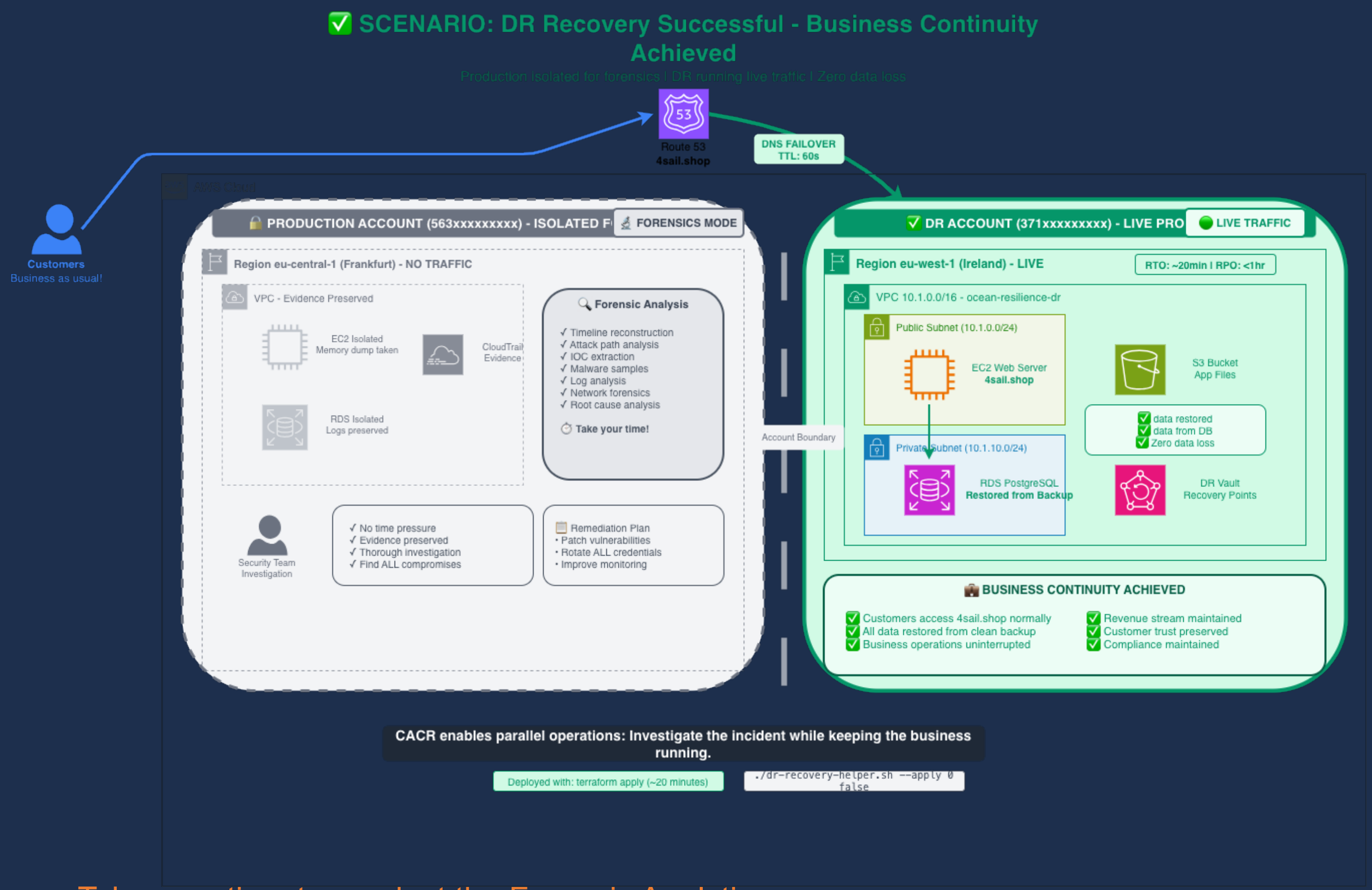


That’s an important advantage of CACR!

Forensic Analytics can be done while the Production runs in the DR Account

Overview

This is the situation we have achieved



Take your time to conduct the Forensic Analytics …

Summary

This was quite easy

Key Takeaways from the POC Implementation Results:

- Cross-Account isolation for separate blast radius
- Cross-Region redundancy for geographic separation
- Point-in-Time recovery to pre-attack state
- Forensics-First approach enabling clean recovery

Technical Metrics:

- Implementation: ~2000 lines Terraform (Multi-AZ Instance)
- Backup storage cost: ~$8-14/month (100GB database)

Recovery Metrics:

- RTO achieved: < 30 minutes (terraform apply)
- RPO achieved: 24 hours (daily backups)
- Forensics time: Unlimited (production already recovered)

We consider some best practices now in regards to terraform

Considerations

About terraform state

Why distinct state files for production and disaster recovery?

Independent Lifecycle:

- Production changes isolated from DR infrastructure
- DR testing without production impact
- Faster recovery (no state conflicts during disaster)
- Team separation (SecOps can manage DR independently)

Further:

- State Locking: Built-in via Terraform Cloud
- Version Control: Automatic state versioning
- Encryption: At-rest and in-transit (SOC2 compliant)

But what about unsupported environments to a single backup-plan?

CACR with Multi-AZ Cluster

Multi-AZ Clusters are not eligible to a simple AWS backup-plan

Multi-AZ Cluster: Requires Lambda + EventBridge

„RDS multi availability zone (Multi-AZ) database instances can be copied, but Multi-AZ clusters do not currently support any copy operations."

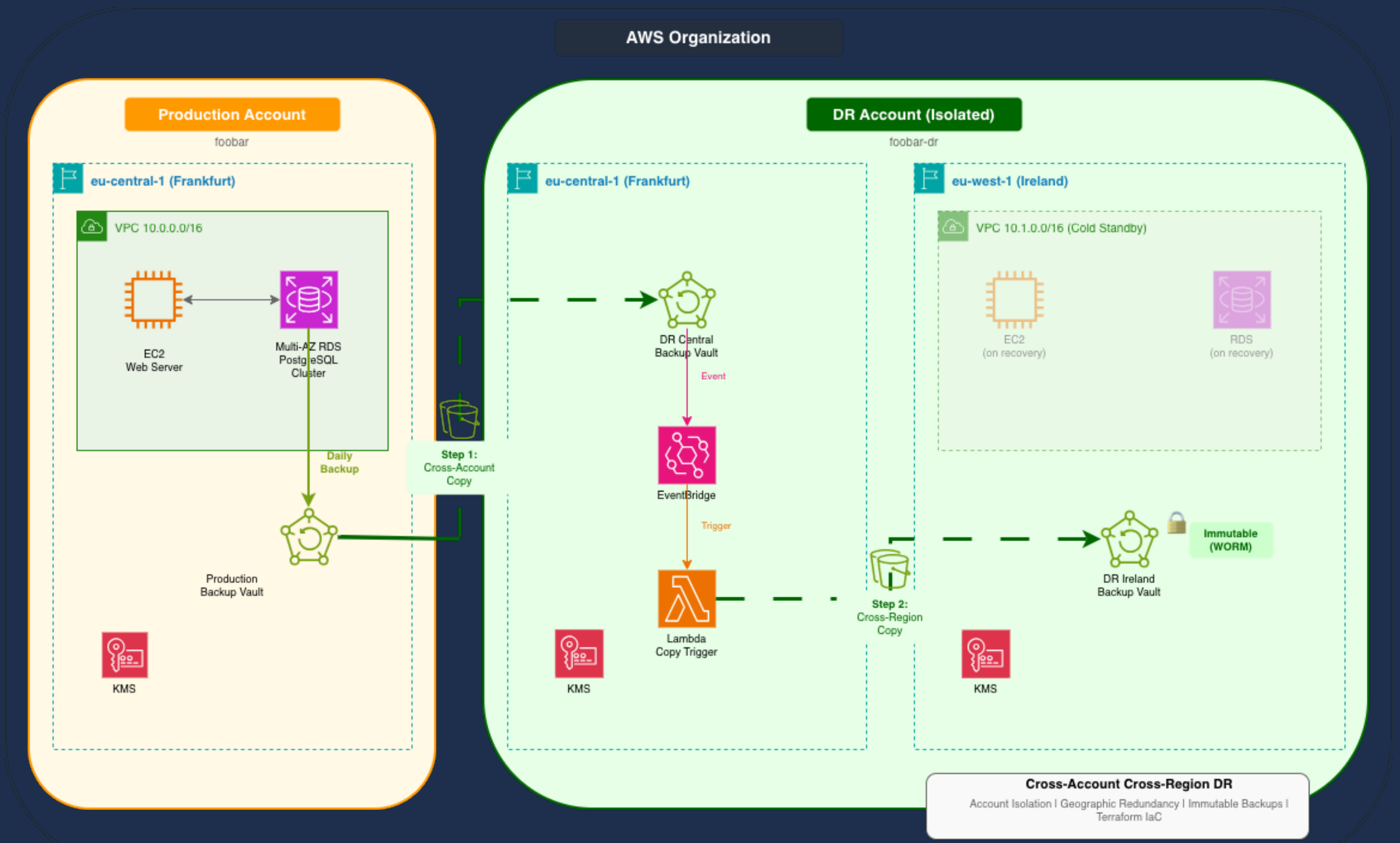
Source: https://docs.aws.amazon.com/aws-backup/ latest/devguide/backup-feature-availability.html

That’s an important information, as AWS recommends Cluster since 2021

CACR for RDS Multi-AZ Clusters

We need a more complicated setup which includes Eventbridge and Lambda

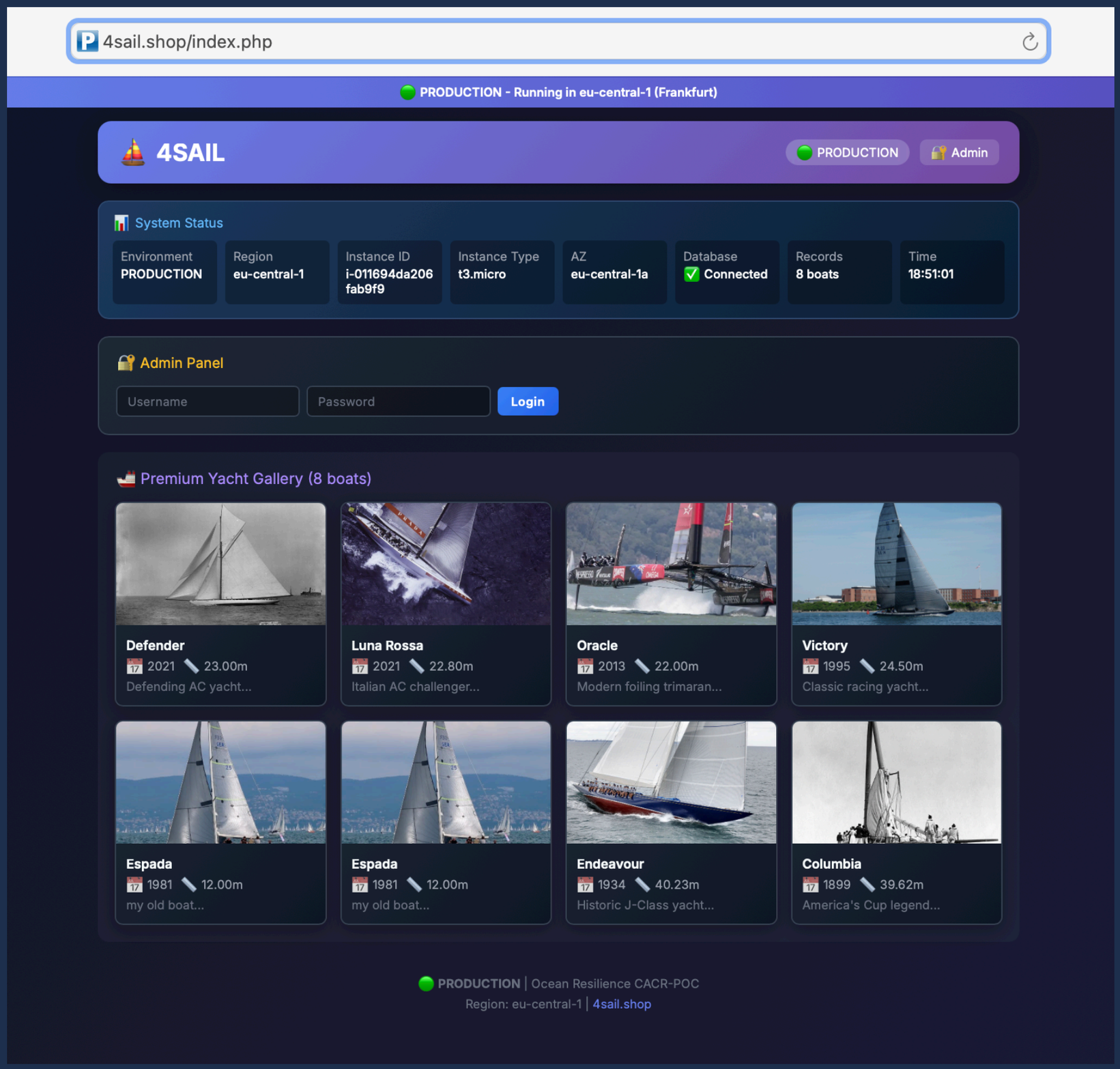
Take a look to the infrastructure as shown here:



This is a more complex setup including backup-roles and some policies

Setup

In Principle we deploy like in the first case a RDS PostgeSQL DB and Webinstance and the basic DR Infrastructure as Cold Pilot



With Terraform we get the environment in short time up and running

Setup

The Difference is that backup data is copied by a Lambda Function into the DR region which is in the DR account, as we are using an Aurora Cluster-DB.

```
devrandom:~ #aws rds describe-db-clusters --profile foobar-prod --region eu-central-1 --query 'DBClusters[?contains(DBClu]
sterIdentifier,`ocean-resilience`)].{Cluster:DBClusterIdentifier,Engine:Engine,EngineVersion:EngineVersion,MultiAZ:MultiA
Z,Status:Status,Endpoint:Endpoint,ReaderEndpoint:ReaderEndpoint,Port:Port,BackupRetention:BackupRetentionPeriod,Preferred
BackupWindow:PreferredBackupWindow,Encrypted:StorageEncrypted,KmsKey:KmsKeyId,AvailabilityZones:AavailabilityZones,DBClust
erMembers:DBClusterMembers[*].DBInstanceIdentifier}' --output table
-------------------------------------------------------------------------------------------------------
|                                              DescribeDBClusters                                     |
+----------------------+------------------------------------------------------------------------------+
|  BackupRetention     |  7                                                                           |
|  Cluster             |  ocean-resilience-production-cluster                                         |
|  Encrypted           |  True                                                                        |
|  Endpoint            |  ocean-resilience-production-cluster.cluster-cn6ik4ws40fe.eu-central-1.rds.amazonaws.com |
|  Engine              |  aurora-postgresql                                                           |
|  EngineVersion       |  15.8                                                                        |
|  KmsKey              |  arn:aws:kms:eu-central-1:563[illegible]:key/658[illegible]-b3b7-bc9eebbd9d1a |
|  MultiAZ             |  False                                                                       |
|  Port                |  5432                                                                        |
|  PreferredBackupWindow|  03:00-04:00                                                                |
|  ReaderEndpoint      |  ocean-resilience-production-cluster.cluster-ro-cn6ik4ws40fe.eu-central-1.rds.amazonaws.com |
|  Status              |  available                                                                   |
+----------------------+------------------------------------------------------------------------------+
||                                          AvailabilityZones                                        ||
|+---------------------------------------------------------------------------------------------------+|
||  eu-central-1a                                                                                    ||
||  eu-central-1c                                                                                    ||
||  eu-central-1b                                                                                    ||
|+---------------------------------------------------------------------------------------------------+|
||                                          DBClusterMembers                                         ||
|+---------------------------------------------------------------------------------------------------+|
||  ocean-resilience-cluster-writer                                                                  ||
|+---------------------------------------------------------------------------------------------------+|
devrandom:~ #
```

AWS command describing the cluster

Setup

Excerpt: Backup-Plan, Eventbridge and Backup-Role

```
devrandom:~ #aws backup get-backup-plan --backup-plan-id 937674cf-c465-4e80-91d0-8409a6ea7e3e --profile foobar-
prod --region eu-central-1 --query 'BackupPlan.Rules[0].{Rule:RuleName,Schedule:ScheduleExpression,SourceVault:
TargetBackupVaultName,CopyTo:CopyActions[0].DestinationBackupVaultArn,Retention:CopyActions[0].Lifecycle.Delete
AfterDays}' --output table

aws events describe-rule --name ocean-resilience-dr-backup-copy-complete --profile foobar-dr --region eu-centra
l-1 --query '{Name:Name,State:State,Description:Description}' --output table

aws iam list-roles --profile foobar-dr --query 'Roles[?contains(RoleName,`ocean-resilience`)].[RoleName,CreateD
ate]' --output table
-------------------------------------------------------------------------------
|                                GetBackupPlan                                |
+-------------+-----------------------------------------------------------------+
|  CopyTo     |  arn:aws:backup:eu-central-1:3717[illegible]:backup-vault:ocean-resilience-dr-vault-central   |
|  Retention  |  90                                                             |
|  Rule       |  daily_backup_with_cross_account_copy                           |
|  Schedule   |  cron(0 15 * * ? *)                                             |
|  SourceVault|  ocean-resilience-production-vault                              |
+-------------+-----------------------------------------------------------------+
-------------------------------------------------------------------------------
|                                DescribeRule                                 |
+-----------------------------------------+-------------------------------------------+----------+
|                Description              |                   Name                    |  State   |
+-----------------------------------------+-------------------------------------------+----------+
|  Trigger Lambda when backup copy job completes |  ocean-resilience-dr-backup-copy-complete  |  ENABLED |
+-----------------------------------------+-------------------------------------------+----------+
-----------------------------------------------------------------
|                           ListRoles                           |
+---------------------------------------+-----------------------+
|  ocean-resilience-backup-copy-job-role |  2026-02-05T08:39:26+00:00  |
|  ocean-resilience-lambda-backup-copy   |  2026-02-05T08:39:26+00:00  |
+---------------------------------------+-----------------------+
devrandom:~ #
```

AWS commands

kenneth-he.de

Recovery Points

Our recovery helper gives us next day the opportunity to chose the recovery point of interest. We promote it into the terraform.tfvars file

```
Searching Production Account (eu-central-1)...
✓ Found production vault: ocean-resilience-production-vault
Searching DR Account Central (eu-central-1)...
✓ Found DR central vault: ocean-resilience-dr-vault-central
Searching DR Account Ireland (eu-west-1)...
✓ Found DR ireland vault: ocean-resilience-dr-vault-ireland


  🚨  DR RECOVERY POINT HELPER  🚨

  Cross-Account Cross-Region DR (DYNAMIC VAULT DISCOVERY)


📦 Ireland DR Vault (for Recovery)
   Vault: ocean-resilience-dr-vault-ireland | Region: eu-west-1

   4 recovery points found:

   IDX    CREATED              STATUS
   ──────────────────────────────────────────────────────────────────────
   [0]  2026-02-08 03:00:00  COMPLETED
        arn:aws:rds:eu-west-1:37[illegible]:cluster-snapshot:awsbackup:copyjob-3f4c4016-7484-4010-96ab-e5030c8d3f84

   [1]  2026-02-07 03:00:00  COMPLETED
        arn:aws:rds:eu-west-1:371[illegible]:cluster-snapshot:awsbackup:copyjob-9a9a6291-fdbf-4e06-95c4-af3259cde924

   [2]  2026-02-06 16:00:00  COMPLETED
        arn:aws:rds:eu-west-1:371[illegible]:cluster-snapshot:awsbackup:copyjob-b9a42f3e-2198-4387-9ac4-96bf1d81ca8c

   [3]  2026-02-05 23:57:32  COMPLETED
        arn:aws:rds:eu-west-1:371[illegible]cluster-snapshot:awsbackup:copyjob-2e61ef5f-2893-41db-ac95-62485ef15e77


🎯 SELECT RECOVERY POINT

   Enter index (0 = newest, q = quit): 2
```

Next step is to recover by invoking „terraform apply“

Disaster recovery

We confirmed to apply …

```
aws_vpc.dr[0]: Still creating... [10s elapsed]
aws_route53_record.dr_status[0]: Still creating... [10s elapsed]
aws_route53_record.dr_root[0]: Still creating... [10s elapsed]
aws_route53_record.dr_www[0]: Still creating... [10s elapsed]
aws_vpc.dr[0]: Creation complete after 13s [id=vpc-06623a463be9ac11f]
aws_internet_gateway.dr[0]: Creating...
aws_subnet.private[1]: Creating...
aws_subnet.private[0]: Creating...
aws_subnet.public[0]: Creating...
aws_subnet.public[1]: Creating...
aws_security_group.web[0]: Creating...
aws_internet_gateway.dr[0]: Creation complete after 0s [id=igw-062d8ec837623e385]
aws_route_table.public[0]: Creating...
aws_subnet.private[1]: Creation complete after 1s [id=subnet-0760da20cd6b293e6]
aws_subnet.private[0]: Creation complete after 1s [id=subnet-04cf1c634461b38cc]
aws_db_subnet_group.dr[0]: Creating...
aws_route_table.public[0]: Creation complete after 2s [id=rtb-07dc316755dbb3603]
aws_security_group.web[0]: Creation complete after 3s [id=sg-01cd4b63de781272c]
aws_security_group.database[0]: Creating...
aws_db_subnet_group.dr[0]: Creation complete after 2s [id=ocean-resilience-dr-db-subnet-group]
aws_security_group.database[0]: Creation complete after 3s [id=sg-013bc65c12cebba22]
aws_rds_cluster.restored_cluster[0]: Creating...
aws_route53_record.dr_status[0]: Still creating... [20s elapsed]
aws_route53_record.dr_www[0]: Still creating... [20s elapsed]
aws_route53_record.dr_root[0]: Still creating... [20s elapsed]
aws_subnet.public[0]: Still creating... [10s elapsed]
aws_subnet.public[1]: Still creating... [10s elapsed]
aws_subnet.public[0]: Creation complete after 11s [id=subnet-02eac38c8370eadb4]
aws_subnet.public[1]: Creation complete after 11s [id=subnet-0e70b436c7576029d]
aws_route_table_association.public[1]: Creating...
aws_route_table_association.public[0]: Creating...
aws_route_table_association.public[0]: Creation complete after 1s [id=rtbassoc-025e460f07eb8361d]
aws_route_table_association.public[1]: Creation complete after 1s [id=rtbassoc-048b932930a415441]
aws_rds_cluster.restored_cluster[0]: Still creating... [10s elapsed]
aws_route53_record.dr_status[0]: Still creating... [30s elapsed]
aws_route53_record.dr_root[0]: Still creating... [30s elapsed]
aws_route53_record.dr_www[0]: Still creating... [30s elapsed]
aws_route53_record.dr_status[0]: Creation complete after 31s [id=Z08333383EH5YOF3C5NLV__dr-status.4sail.shop_TXT]
aws_route53_record.dr_www[0]: Creation complete after 31s [id=Z08333383EH5YOF3C5NLV_www.4sail.shop_A]
aws_route53_record.dr_root[0]: Creation complete after 31s [id=Z08333383EH5YOF3C5NLV_4sail.shop_A]
aws_rds_cluster.restored_cluster[0]: Still creating... [20s elapsed]
aws_rds_cluster.restored_cluster[0]: Still creating... [30s elapsed]
aws_rds_cluster.restored_cluster[0]: Still creating... [40s elapsed]
```

This may take a while…

Disaster recovery done

The entire workload has been successfully recovered in our DR account and region

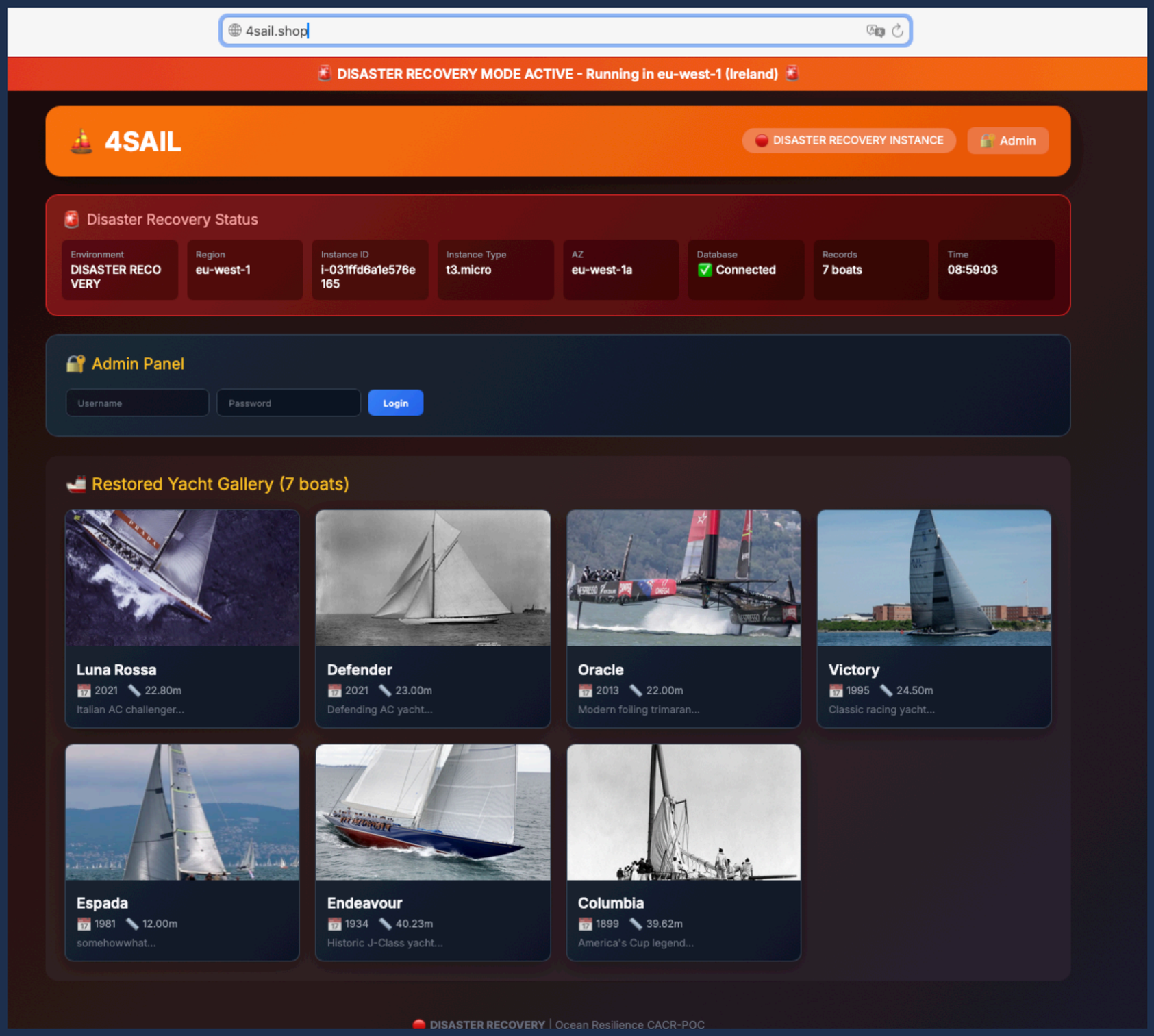
```
    EOT
dr_website_url = "http://34.254.105.94"
ec2_instance_id = "i-031ffd6a1e576e165"
restored_db_reader_endpoint = "ocean-resilience-dr-cluster-20260208083511.cluster-ro-cnqok6ckiiw4.eu-west-1.rds.amazonaw
s.com"
restored_db_type = "Aurora Cluster"
vpc_id = "vpc-06623a463be9ac11f"
public_subnet_ids = [
        "subnet-02eac38c8370eadb4",
        "subnet-0e70b436c7576029d",
    ]
restored_db_endpoint = "ocean-resilience-dr-cluster-20260208083511.cluster-cnqok6ckiiw4.eu-west-1.rds.amazonaws.com"
restored_db_identifier = "ocean-resilience-dr-cluster-20260208083511"
app_bucket_arn = "arn:aws:s3:::ocean-resilience-dr-app-
dns_failover_command = <<-EOT
        # Check DNS propagation:
        dig 4sail.shop +short

        # Should show: 34.254.105.94

        # Check DR Status TXT Record:
        dig _dr-status.4sail.shop TXT +short
    EOT
ec2_public_ip = "34.254.105.94"
hosted_zone_id = "Z08333383EH5YOF3C5NLV"
app_bucket_name = "ocean-resilience-dr-app-
recovery_point_arn = "arn:aws:rds:eu-west-1:            :cluster-snapshot:awsbackup:copyjob-b9a42f3e-2198-4387-9ac4-96bf
1d81ca8c"
restored_db_address = "ocean-resilience-dr-cluster-20260208083511.cluster-cnqok6ckiiw4.eu-west-1.rds.amazonaws.com"
restored_db_port = "5432"
vault_arn = "arn:aws:backup:eu-west-1:371        :backup-vault:ocean-resilience-dr-vault-ireland"
devrandom:~/AWS-Projects/CACR-POC/foobar-dr/dr-recovery #dig 4sail.shop +short
34.254.105.94
devrandom:~/AWS-Projects/CACR-POC/foobar-dr/dr-recovery #dig _dr-status.4sail.shop TXT +short
"dr-active;region=eu-west-1;timestamp=20260208083511"
devrandom:~/AWS-Projects/CACR-POC/foobar-dr/dr-recovery #
```

Aurora Cluster is working. Sure, we lost all Data since the snapshot was created. But better than having lost everything

Route53 points the A-Record now to Ireland

Check

Workload Up and Running as expected.



Easy, but still a lot of work to catch up data loss

kenneth-he.de

Takeaways

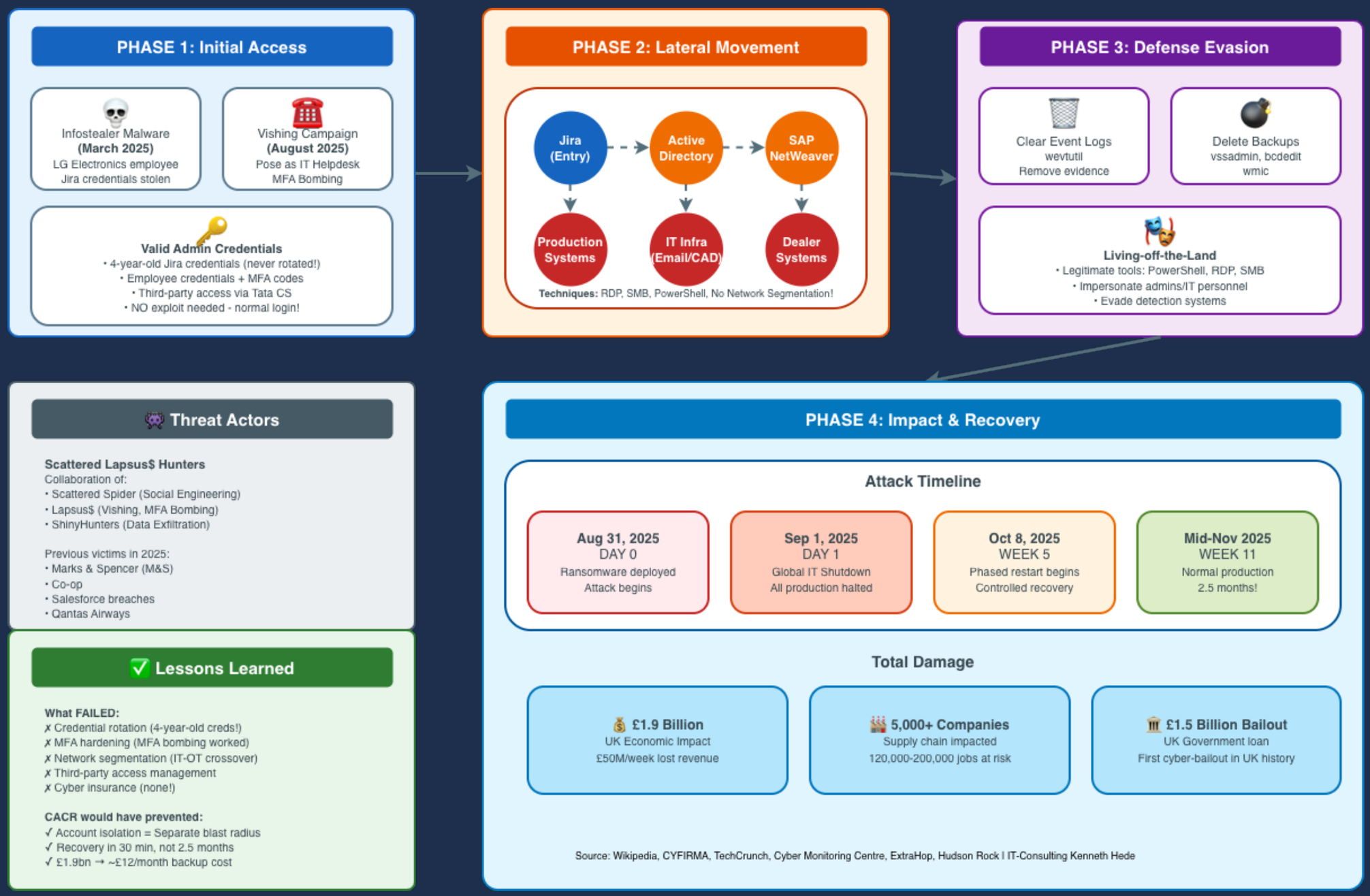
1. Multi-AZ != Security Control — Geographic redundancy without account isolation is insufficient
2. Compliance Frameworks Are Converging — ISO 27001, DORA, NIS2 all mandate separation
3. Cost is Negligible — to prevent £1.9 billion scenarios
4. Recovery Speed Matters — Hours/Days vs. 2.5 months changes everything
5. Forensics Capability is Critical — Clean recovery requires preserved evidence

Now, we have a closer Look to a Scenario which we never want to have

The Reality Check: When Disaster Recovery Fails

Jaguar Land Rover Attack Case Study

A quick look to the Attack Path:



Note that the initial Footprint was months upfront the final attack

The Numbers:

- £1.9 billion total UK economic damage
- 2.5 months until normal production resumed
- 5,000+ companies affected in supply chain
- 120,000-200,000 jobs at risk
- £1.5 billion UK government bailout (first cyber-bailout in history)

The Question:

Could Cross-Account Cross-Region (CACR) disaster recovery have prevented this?

How They Got In

- **March 2025 - The Setup:**

- Infostealer malware infects LG Electronics employee
- Jira credentials harvested (third-party with JLR access)
- Credentials from 2021 - never rotated for 4 years!
- HELLCAT ransomware group exfiltrates 700+ documents

- **August 2025 - The Main Attack:**

- Vishing campaign: Attackers pose as IT Helpdesk
- Social engineering: "Technical issues require immediate action"
- MFA Bombing: Flood authentication requests until employee approves
- Result: Valid admin credentials + MFA codes obtained

- **The Entry Point:**

- Atlassian Jira instance
- Tata Consulting Services (outsourced IT provider)
- Same entry point used in M&S and Co-op breaches

No sophisticated exploit needed - just social engineering

Spreading Through The Network

Attack Path:

1. Jira (initial entry)
2. Active Directory compromise
3. SAP NetWeaver exploitation
4. Production systems access
5. IT infrastructure (Email, CAD/PLM)
6. Dealer ordering systems

- **Techniques Used:**

- T1021.001: Remote Desktop Protocol (RDP)
- T1021.002: SMB/Windows Admin Shares
- T1027: Living-off-the-Land (PowerShell, legitimate tools)

- **The Critical Failure:**

- **NO network segmentation**
- **IT-OT convergence = Production systems reachable**
- **Single account = entire infrastructure accessible**

Result: Complete compromise in days, not weeks

Covering Their Tracks

Evasion Tactics:

1. Clear Event Logs
- Tool: wevtutil (Windows Event Utility)
- Result: Forensic evidence destroyed

2. Delete Backups
- Tools: vssadmin, bcdedit, wmic
- Target: Volume Shadow Copies
- Result: Recovery impossible from local backups

3. Living-off-the-Land
- Use legitimate tools: PowerShell, RDP, SMB
- Impersonate administrators and IT personnel
- Evade detection systems (no custom malware signatures)

The Detection Gap:
- Weak monitoring/alerting
- No anomaly detection for lateral movement
- Third-party access poorly tracked
- Delayed incident response

First signs detected: Only when production systems failed

The Devastating Impact

Timeline:

Day 0 (Aug 31, 2025):
- Ransomware deployed across global infrastructure
- Attack begins

Day 1 (Sep 1, 2025):
- JLR proactively shuts down all IT systems globally
- Production halted at: UK, Slovakia, India, Brazil, China
- Dealer systems offline
- Email and CAD/PLM systems down

Week 5 (Oct 8, 2025):
- Phased restart begins
- Controlled, cautious recovery
- Forensic investigation ongoing

Week 11 (Mid-November 2025):
- Normal production finally resumed
- 2.5 months total disruption

Financial Damage £50 million per week lost revenue

What If They Had Cross-Account Cross-Region DR?

When Attack Happens:

- Production account compromised ✓
- Attacker has admin access ✓
- All production data encrypted ✓

But With CACR:

- DR Account is ISOLATED
- Attacker cannot access separate AWS account and data
- Yesterday's snapshots in DR vault are SAFE
- CloudTrail logs preserved for forensics

Recovery Timeline:	
Traditional DR	**With CACR approach**
2.5 months (11 weeks)	Hours/Days
£50M/week lost	24 hours data loss
5,000+ companies impacted	Supply chain protected
£1.9 billion total damage	Additional monthly backup cost

What If They Had Cross-Account Cross-Region DR?

CACR Recovery Steps:
1. Freeze production account (preserve evidence)
2. Switch to DR account
3. terraform apply (restore infrastructure)
4. Update DNS to DR region
5. Business operational in < 1 hour

While Production Runs in DR:
- Forensics team analyzes attack path in production account
- No time pressure
- Clean recovery guaranteed
- CloudTrail analysis identifies attack vector

Data Loss Reality:
- 24 hours acceptable for ransomware scenario
- Recent data is potentially compromised anyway
- PITR available in source account for operational errors
- Trade-off: Isolation vs. Precision

Financial Damage £50 million per week lost revenue

Thinking beyond Disaster Recovery

Once CACR Disaster Recovery has avoided the Doomsday:

1. Implement Disaster Recovery towards another Region, because you don’t know how long your production will have to continue in your DR Account and Region. In Case you’re going to be attacked again - you have a solution.

2. Think about Reconstitution, because you’d probably good reason having chosen your initial Region for your workloads (latency?). Reverse the process, you already mastered.

CONSULTING NEEDED?

Contact me: https://kenneth-he.de

Key Takeaways

1. Multi-AZ ≠ Security Control
- Protects hardware failures, not credential compromise
- Same account = same blast radius
- Geographic redundancy without account isolation is insufficient

2. Third-Party Risk Is Real
- Supply chain security is critical
- Vendor access needs isolation

3. Traditional DR Is Too Slow
- 2.5 months recovery time = business extinction
- Manual rebuilds are error-prone
- Infrastructure-as-Code enables rapid recovery

4. Account Isolation Is Mandatory
- Separate AWS accounts = separate credentials
- CACR provides true isolation
- Compliance requirement (ISO 27001, DORA, NIS2)

5. The Stakes Are Existential
- £1.9 billion damage
- 120,000+ jobs at risk
- Government intervention required
- Some suppliers went bankrupt

CACR isn't about preventing attacks. It's about surviving them.

Sources & Attribution #1

Incident Reports:

1. Cyber Monitoring Centre (CMC) - Official UK Assessment
- URL: cybermonitoringcentre.com/2025/10/22/cyber-monitoring-centre-statement-on-the-jaguar-land-rover-cyber-incident-october-2025/
- Key Data: £1.9 billion damage estimate, 5,000+ organizations affected

2. Wikipedia - Jaguar Land Rover Cyberattack
- URL: en.wikipedia.org/wiki/Jaguar_Land_Rover_cyberattack
- Timeline and government response

3. Bank of England - Monetary Policy Report (November 2025)
- GDP impact statement

Technical Analysis:

4. CYFIRMA - Investigation Report
- URL: cyfirma.com/research/investigation-report-on-jaguar-land-rover-cyberattack/
- Technical details on Jira compromise, HELLCAT tactics

5. Hudson Rock - Infostealer Analysis
- URL: infostealers.com/article/jaguar-land-rover-breached-by-hellcat-ransomware
- Credential theft methodology, 4-year-old credentials

6. Treblle - JLR Breach Breakdown
- URL: treblle.com/blog/jlr-breach-breakdown-analysis
- Social engineering details, vishing campaign

7. ExtraHop - Ransomware Analysis
- URL: extrahop.com/blog/ransomware-hits-jlr-a-supply-chain-under-siege
- MITRE ATT&CK mapping, lateral movement techniques

Thanks to all creators and contributors of these sources!

Sources & Attribution #2

Media Coverage:

8. TechCrunch - UK Government Bailout
- URL: techcrunch.com/2025/09/29/uk-government-bails-out-jaguar-land-rover
- £1.5 billion loan details

9. NBC News - GDP Impact
- URL: nbcnews.com/tech/security/jaguar-land-rover-hack-hurt-uk-gdp
- Economic consequences, Bank of England statement

10. Computer Weekly - Supply Chain Impact
- URL: computerweekly.com/news/366633395/Jaguar-Land-Rover-attack-to-cost-UK-19bn
- Human impact, job losses

AWS Documentation:

11. AWS Backup Feature Availability
- URL: docs.aws.amazon.com/aws-backup/latest/devguide/backup-feature-availability.html
- Multi-AZ Cluster limitation documentation

12. AWS Announcement - Single-Action Copy
- URL: aws.amazon.com/about-aws/whats-new/2025/10/aws-backup-single-action-database-snapshot-copy-regions/
- October 2025 feature release

13. AWS Official Blog Post (November 2025 Update):
- URL: /aws.amazon.com/blogs/storage/protecting-encrypted-amazon-rds-instances-with-cross-account-and-cross-region-backups/

Thanks for reading!

kenneth-he.de

Next Steps

Let’s assess your disaster recovery posture in a free 30-minute session:

1. Gap analysis of your current AWS backup strategy. Compliance mapping for ISO 27001, DORA, NIS2 and KRITIS. ROI calculation for CACR implementation. Custom architecture recommendations for your environment.

2. Available resources: DORA/NIS2 Compliance Checklist, Terraform State Management Guide, Attack Path Analysis with Amazon Athena. Request access via the contact link below.

FREE DISCOVERY SESSION

Get in touch: https://kenneth-he.de