Test Project

Module Troubleshooting

Cloud Computing

Submitted by:
Amazon Web Service, Independent Test Project Designer

Contents

This Test Project consists of the following documentation/files:

- WSC2026_TP53_MA_actual (this document)

Use of AI Tools and Models

You may not use any AI tool or service to produce, debug, or explain your work at any point. This includes external services, personal accounts, and any model self-hosted on competition infrastructure. Where a task explicitly requires your solution to use an AI model, follow that task's instructions — this is part of the task, not assistance.

Description of project and tasks

Troubleshooting Module: This section evaluates your cloud computing skills through a series of challenges that replicate real-world scenarios, testing your ability to effectively diagnose issues and implement solutions in complex cloud environments.

Technical Details

Troubleshooting

Where Did My Deployment Go

Summary

Your team has inherited a completely broken CI/CD pipeline. The previous engineer, Alex, left in a hurry on Friday afternoon, leaving only a sticky note: "Sorry, tried my best but ran out of time. Good luck!" It's now Monday morning, and the product manager is waiting in the conference room for your demo.

Architecture: CodeCommit → CodeBuild → ECR → ECS (Fargate) → ALB → CodeDeploy (Blue/Green)

Important Notes:

- Each task depends on the completion of the previous one — fix them strictly in order
- Problems are layered — fixing one issue may reveal the next
- Some operations (especially ECS service deletion) usually take 5–8 minutes — DRAINING is normal; wait until the service is fully gone before recreating
- Details like container names, port numbers, and health check paths must remain consistent throughout the entire flow — including both target groups used by blue/green
- The deployment controller type of an ECS service cannot be changed after creation — choose carefully when you create or recreate the service

REMINDER

This is a Troubleshooting Challenge.

Please use us-east-1 region.

© WorldSkills International

Inventory

- CodeBuild
- CodeDeploy
- CodePipeline
- ECS
- ECR

We've Been Hacked!!!

Summary

Your company's static website is served through a CloudFront + S3 architecture. The security team received browser telemetry alerts — users are being served pages containing credit card skimming scripts (Magecart-style injection). Preliminary investigation reveals that the attacker compromised the CDN edge layer to inject malicious code, poisoned caches to persist the attack, established data exfiltration channels through S3 replication, and deliberately destroyed audit logs and security protections to cover their tracks.

Your Mission: Trace the full attack path from edge injection to evidence destruction, fix all misconfigurations, and restore the platform's security posture.

Important Notes:

- The CloudFront distribution is live — you can observe the injected responses with curl
- You must fix misconfigurations in place — do not delete the distribution or the website bucket
- Investigate CloudFront Functions, S3 bucket configurations, cache policies, and WAF associations
- All resource names are prefixed with sec-
- Some fixes require time to take effect (e.g., CloudFront cache invalidation, distribution deployment)

How to Investigate

All lab resources use the sec- name prefix. The CloudFront distribution domain and S3 bucket name are provided in the lab materials on this page as stack output values. You can also list sec-* resources in the AWS consoles. Use curl -I https://YOUR-CF-DOMAIN/ to inspect response headers and curl https://YOUR-CF-DOMAIN/ to view page content. Some hardening changes (cache invalidation, distribution updates) can take one to several minutes to take effect before re-checking progress.

REMINDER

This is a Troubleshooting Challenge.

Please use us-east-1 region.

Inventory

- CloudFront
- S3

© WorldSkills International

Dynamic Image Transformation System — Emergency Fix

Summary

AnyCompany provides real-time image resizing, cropping, and format conversion for e-commerce clients. The AWS-based **Dynamic Image Transformation (DIT)** system has three layers:

- **Image Processing**: CloudFront → ALB → ECS Fargate → Rekognition / S3
- **Management API**: CloudFront → API Gateway → Lambda → DynamoDB
- **Web Management Portal**: CloudFront → S3 static hosting + Cognito

The lead engineer resigned abruptly, leaving an architecture note and a last "quick fix" that broke several paths. Restore the system in roughly this order: **Web Portal → Authentication → API → Data / Image / Security**.

Use the us-east-1 region. S3 buckets for this system must not be publicly accessible. After CloudFront or API Gateway changes, wait until deployments finish before validating.

REMINDER

This is a Troubleshooting Challenge.

Please use us-east-1 region.

Inventory

- CloudFront
- ECS
- API Gateway
- Cognito
- S3
- DynamoDB

Mist Ridge Sanctuary: The Missing Animal Records

Summary

Mist Ridge Mythical Animal Sanctuary is nestled among mountains and wetlands. It is home to more than a thousand mythical animals, and staff record their health, habitats, migration activities, and interactions every day. This information helps veterinarians arrange treatment and enables rangers to determine which areas need additional protection.

As the number of animal records and field stations grew, the sanctuary's small technical team struggled to keep up with the increasing maintenance workload. To let the team focus on conservation, the sanctuary gradually rebuilt its information platform using a serverless architecture. The platform requires no dedicated server administration, saves costs during normal operations, and scales flexibly during traffic peaks such as migration season. For years, it has quietly and reliably supported the sanctuary's daily work.

That reliability makes the current incident especially unusual. With the annual migration season approaching, animal records in the platform have become incomplete, and some features used to support animal placement and care no longer work correctly. Veterinarians cannot obtain timely information about priority animals, keepers are struggling to develop appropriate placement plans, and field patrols are also being affected.

To make matters worse, the engineer most familiar with the system is conducting a long-term survey in a remote conservation area and cannot be reached in the near future. The remaining team is reluctant to risk altering valuable historical records, so they have asked you to take over the investigation as an incident response expert. You must use the limited clues available to identify the problems, restore reliable and complete information for the staff, and ensure that recovered features do not fail again.

© WorldSkills International

REMINDER

This is a Troubleshooting Challenge.

Please use us-east-1 region.

Inventory

- CloudFront
- Database
- S3

Connected Vehicle Application: Expert CloudWatch Log Analytics Challenge

Summary

A rapidly growing connected-vehicle company provides drivers worldwide with an integrated mobile and in-vehicle application for monitoring vehicle status, planning charging, managing subscriptions and value-added services, and requesting remote control, roadside assistance, and mobility services. As its vehicle and customer base grows, the application has become a critical link between vehicles, drivers, and cloud services.

The backend consists of cooperating cloud services distributed across AWS accounts according to team ownership and business boundaries. The resulting request, business-event, and vehicle-service logs are essential both to customer experience and to reconstructing complex incidents.

The platform uses Amazon CloudWatch as its unified observability system. CloudWatch lets teams analyze large log volumes without operating a separate analytics cluster, while the new Log Analytics experience brings interactive queries, live log streams, and high-cardinality analysis into one workspace. This makes it possible to move quickly from symptoms to evidence without losing context.

During a recent traffic peak, a small number of drivers reported that service status shown in the application did not match their actual experience. The issue has no reliable reproduction path, and the partial metrics owned by individual teams do not explain the full behavior. Normal traffic continues at high volume, so manually reviewing events is no longer viable.

You are the observability and distributed-systems expert assigned to the incident. Use the available logs and controlled lab resources to establish defensible conclusions and provide a safe basis for remediation. Historical production logs are valuable evidence; unauthorized modification or destructive actions may invalidate the investigation.

General Constraints

- All resources are in us-east-1. Each task states its fixed log groups and constants; deployment-specific time windows and resource identifiers are available in the CloudFormation Stack Outputs.
- Do not delete, overwrite, or re-ingest challenge logs, and do not alter event timestamps.
- Source logs and centralized copies may both be visible; do not count the same business event more than once.
- Each task has one final objective. Standard inputs required by later tasks are supplied independently by the environment to prevent cascading score loss.
- Do not modify business data or invoke remediation APIs unless a task explicitly permits it.

REMINDER

This is a Troubleshooting Challenge.

Please use us-east-1 region.

© WorldSkills International

Inventory

- CloudWatch
- Event Bridge
- S3

Cross-Border E-Commerce Platform Multi-VPC Network & Security Troubleshooting

Summary

ShopGlobal is a cross-border e-commerce platform running its core business systems on AWS using a multi-VPC isolation architecture. The architecture consists of three VPCs interconnected via Transit Gateway: a Management VPC for the operations management plane, an Application VPC hosting the core e-commerce application, and a Shared Services VPC providing an internal compliance audit service. The core application app-server in the Application VPC depends on Secrets Manager for database credentials, S3 for product catalog configuration, a cross-VPC PrivateLink connection for compliance checks, and an ALB fronted by Network Firewall for external API access.

Last Friday the deployment team finished setting up the infrastructure and left for the weekend. On Monday morning, the system is completely non-functional: operations cannot remotely connect to the instance, the application cannot read its configuration or credentials, cross-VPC service calls fail, and external access times out. As the on-call engineer, you need to systematically troubleshoot the network, permission, and security configuration issues layer by layer to restore the system within 1 hour.

The challenge is structured as a progressive troubleshooting exercise across four layers: cross-VPC management connectivity, VPC Endpoint service access, PrivateLink cross-VPC service calls, and Network Firewall deployment with external access routing.

REMINDER

This is a Troubleshooting Challenge.

Please use us-east-1 region.

Inventory

- EC2
- Transit Gateway
- VPC
- Secrets Manager
- S3

© WorldSkills International

EchoCast Media Video-to-Podcast — Serverless Audio/Video Processing Pipeline

Summary

EchoCast Media is a content company that has accumulated a large library of online seminars (webinars) recorded in MP4 over the years. The marketing team wants to "repurpose" these videos: first convert them into audio that can be listened to on podcast platforms, then further generate text transcripts and AI summaries for search, SEO, and multi-channel distribution to reach more listeners.

The tech lead has decided to automate this with a fully Serverless pipeline, and hands the first version to you to implement:

"The assets and base environment operations are already set up — networking, storage, a build host, and the execution role are all in place. But the pipeline itself is still empty. First get the 'video-to-audio' stage working, then progressively wire up playback, transcription, and summarization."

Environment:

- Region: us-east-1
- S3 bucket serverless-storage-bucket-<account-id>, pre-populated with: mp4/webinar-1.mp4 (source video), mp4/probe.mp4 (a few-seconds short test clip, for quick verification), bin/ffmpeglib.zip (pre-compiled FFmpeg binary package)
- EC2 serverless-storage-linux: a Linux instance in the environment, connected via Session Manager (no SSH), with python3.13 / pip / aws / jq installed
- Lambda execution role serverless-storage-role: starts with minimal permissions; you need to attach AWS managed policies to it as each task requires. This environment is a sandbox, and only the following three managed policies may be attached, with no support for creating your own policies (inline or customer-managed): AmazonS3FullAccess, AWSLambda_FullAccess, AmazonBedrockFullAccess (note that AWSLambda_FullAccess has an underscore in the middle).

Evaluation (E2E Runner):

All three tasks are evaluated through a single E2E Runner (the Lambda function gameXXX-e2e-runner). It performs a real end-to-end test (actually invoking the functions / APIs you build, actually transcribing, etc.); the slower operations all run inside the Runner, and it securely writes the results to the platform.

Two-step submission (can be run in CloudShell or on serverless-storage-linux, region us-east-1):

1. Run the Runner (replace taskN with task1 / task2 / task3):

```
aws lambda invoke --function-name gameXXX-e2e-runner \
  --cli-binary-format raw-in-base64-out --cli-read-timeout 0 \
  --payload '{"test_phase":"taskN"}' out.json && cat out.json
```

2. After confirming "all_passed": true in the output, click Check Progress to complete the task (Check Progress only reads the result the Runner just wrote, and returns within seconds).

Example output:
{"test_phase":"task1","all_passed":true,"checks":[{"name":"...","passed":true,"message":"ok"}]}

Notes:

- Always run the Runner first, then click Check Progress; otherwise you will get "Test result not found".
- If all_passed is false, the Runner only gives a generic message and will not point out the specific cause — debug and fix it yourself, then re-run the Runner before submitting.
- Some operations run inside the Runner may take a while; be sure to keep --cli-read-timeout 0 and wait patiently.

© WorldSkills International

Tasks must be completed in order: each stage's output is the next stage's input.

REMINDER

This is a Troubleshooting Challenge.

Please use us-east-1 region.

Inventory

- Lambda
- S3
- EFS
- FFmpeg

SwiftLogistics Real-Time Package Tracking Platform — CloudFormation & Custom Resource Challenge

Summary

You are a newly hired cloud platform engineer at SwiftLogistics, a logistics company. The company is building a next-generation real-time package tracking system that reports package location and status in real time via IoT devices.

The previous engineer, Kevin, left the company two months ago, leaving behind a defective CloudFormation template and an unfinished Custom Resource Lambda. The CTO explains the situation to you:

"The CloudFormation code Kevin left behind is riddled with problems, and that Lambda responsible for managing route configuration and database schema has never run successfully. Singles' Day is coming, and we must get the system live within two weeks. You take care of it."

E2E Test Instructions:

Task 2-3 are verified using the E2E Runner (task2-e2e-runner). The Runner creates a real CloudFormation Stack to invoke your Lambda and returns the result synchronously (about 25-30 seconds).

Submission Steps:

Step 1: Package the Lambda code

zip -r lambda.zip index.py

Step 2: Update the Lambda function code

aws lambda update-function-code \

  --function-name swift-logistics-test-route-config \

  --zip-file fileb://lambda.zip

Step 3: Run the E2E test (replace <phase> with the test phase given in each task)

aws lambda invoke --function-name task2-e2e-runner \

  --cli-binary-format raw-in-base64-out \

  --payload '{"test_phase":"<phase>"}' \

  /tmp/result.json && cat /tmp/result.json

Step 4: After confirming all_passed=true, click Check Progress

Important Notes:

© WorldSkills International

- Resources deployed by Task 1 use the prefix swift-logistics-dev-*
- Resources operated on by Task 2-3 use the prefix swift-logistics-test-* (pre-provisioned by the environment)
- The two sets of resources do not affect each other. When debugging Task 2-3, check the DynamoDB tables and Aurora data with the test- prefix
- Task 2-3 must be completed in order

REMINDER

This is a Troubleshooting Challenge.

Please use us-east-1 region.

Inventory

- CloudFormation
- Custom Resource
- Lambda
- Aurora Serverless v2

CloudVote Platform Operations Challenge

Summary

CloudVote Inc. is migrating its voting application onto EKS, using KRO (Kube Resource Orchestrator) to compose ACK-managed AWS resources (RDS / ElastiCache / DynamoDB / Lambda, etc.) together with native K8s resources into high-level abstractions (ResourceGraphDefinition, RGD for short). This challenge contains three independent tasks.

Environment Overview:

- EKS cluster
- Participant EC2: ParticipantInstance
- Challenge assets are located under /opt/jam/assets/ on the ParticipantInstance

Submission and Grading:

After completing each task, you must submit in the following order:

1. Trigger the corresponding CodeBuild project (run on the ParticipantInstance):

Task 1

aws codebuild start-build --project-name jam-validator-jam-eks-kro-t1 --region us-east-1

Task 2

aws codebuild start-build --project-name jam-validator-jam-eks-kro-t2 --region us-east-1

Task 3

aws codebuild start-build --project-name jam-validator-jam-eks-kro-t3 --region us-east-1

2. Wait for CodeBuild to finish (about 2-5 minutes)

3. Return to the challenge page and click Check Progress

⚠ Note: Make sure you trigger the correct CodeBuild project (t1/t2/t3 map to different tasks). If you click Check Progress without running CodeBuild first, the grading result will be the previous cached one and will not reflect your latest changes.

REMINDER

© WorldSkills International

This is a Troubleshooting Challenge.

Please use us-east-1 region.

Inventory

- EKS
- KRO (Kube Resource Orchestrator)
- ACK (AWS Controllers for Kubernetes)
- OPA Gatekeeper

Agentic Snake Cup — GenAI Agent Challenge

Summary

SnakeTech is hosting an internal AI programming competition — the Agentic Snake Cup. Contestants must use Amazon Bedrock foundation models and AgentCore Runtime to develop and deploy a self-deciding Snake AI Agent. The competition platform (Engine) calls your Agent every tick to get a movement decision, driving the snake across the grid to eat food and avoid obstacles.

As a contestant, you need to complete the following three stages of the challenge:

- Infrastructure Setup: Deploy the competition's public-facing site
- Agent First Attempt: Develop and deploy your Snake Agent, achieving a WIN at the entry difficulty
- Agent Advanced: Achieve a WIN at a higher difficulty

Environment Notes:

- Engine Platform: A web console exposed via an ALB, used to submit matches, view results, and watch replays
- S3 Resource: The competition rules site code is stored at s3://gameXXX-assets-{AccountId}-us-east-1/player_input.zip
- Pre-provisioned IAM Roles: The Runtime Role and CodeBuild Role required for AgentCore deployment are pre-provisioned; contestants do not need to create any IAM resources
- Bedrock Models: Only Amazon Nova series foundation models in the us-east-1 region may be used (e.g. Nova Micro / Lite / Pro)

REMINDER

This is a Troubleshooting Challenge.

Please use us-east-1 region.

Inventory

- EC2
- Application Load Balancer
- CloudFront
- Amazon Bedrock AgentCore

© WorldSkills International