Terraform Debugging Journey: Private S3 + CloudFront OAC

Introduction

This document summarizes the debugging journey of deploying a private S3 bucket behind CloudFront with Origin Access Control (OAC) using Terraform. We encountered multiple issues, investigated root causes, and applied fixes to finally achieve a successful deployment.

Initial Setup

We structured our Terraform project into environments (staging/prod) and modules (cloudfront_oac). The staging environment invoked the module with variables for project name, domain, and CDN domain. Providers were configured for us-east-1 because ACM certificates for CloudFront must be issued in that region.

Errors Encountered

1. **Undefined Provider (aws.us_east_1):** Terraform complained about a missing provider alias. We fixed this by explicitly declaring an aliased provider `aws.us_east_1` in the staging environment. 2. **Typos and Syntax Errors in Module:** The initial module had several typos like `resoure` instead of `resource`, missing quotes, and misnamed attributes. We replaced the module with a clean, validated version. 3. **ACL Error with CloudFront Logs:** When creating the CloudFront distribution, Terraform failed with the error: 'The S3 bucket specified for CloudFront logs does not enable ACL access.' This happened because the logs bucket was configured with `BucketOwnerEnforced`, which disables ACLs. CloudFront requires ACLs to deliver logs.

Debugging and Fixes

For the provider issue, we added an explicit alias for `us_east_1` and passed it to the module. For the module errors, we carefully rebuilt the main.tf, variables.tf, and outputs.tf with correct syntax. For the ACL issue, we updated the logging bucket configuration: - Changed object ownership from `BucketOwnerEnforced` to `BucketOwnerPreferred`. - Added an `aws_s3_bucket_acl` resource with `acl = "log-delivery-write"` to allow the CloudFront log-delivery group. - Retained public access block for security.

Final Outcome

After these changes, we ran `terraform plan` successfully, then `terraform apply`. The apply completed after ~3 minutes, during which CloudFront distribution was created, ACM certificate was validated, and Route 53 alias records were provisioned. Outputs confirmed successful deployment: cdn_url, distribution_id, site bucket, and logs bucket. We can now upload static site files to the private S3 bucket, invalidate the CloudFront cache, and securely serve the site via HTTPS and OAC.

Lessons Learned

- Always double-check provider aliases when working with regional requirements (like ACM in us-east-1). - Typos in Terraform HCL can cascade into multiple confusing errors; using `terraform validate` early helps. - CloudFront logging still depends on S3 ACLs. Avoid disabling ACLs on the log bucket. - Iterative debugging (fix one error, re-run, observe output) is the fastest way to converge on a working state. - Good module structure with variables, outputs, and validated configs saves hours of frustration.