Cloud & Automation Troubleshooting Case Studies for CloudOps Engineers

10 Real-world Cloud Automation Scenarios with Commands, Logs, and Remediation Steps

Case 1: Terraform State File Corruption

Problem: terraform apply fails with state lock errors and plan mismatches.
Investigation:

```
$ terraform plan
Error: Failed to read state file: unexpected EOF
$ ls -l terraform.tfstate
-rw-r--r-- 1 ubuntu ubuntu 12 bytes terraform.tfstate.corrupt
```

Root Cause: State file partially overwritten during concurrent runs; lock not honored.
Resolution: Restore state from latest backup (S3 versioning) and run `terraform state replace-provider` if needed; enable remote state locking.
Prevention: Use remote state backend with versioning and strong lock (S3 + DynamoDB) and enforce CI pipeline gating.

Case 2: EC2 Provisioning Fails due to IAM Role Misconfiguration

Problem: EC2 instances fail to fetch user data and SSM agent cannot register.
Investigation:

```
$ aws ec2 describe-instances --instance-ids i-0123456789abcdef0
InstanceProfileArn: arn:aws:iam::123456789012:instance-profile/EC2Role
$ aws iam get-role --role-name EC2Role
AssumeRolePolicyDocument missing ec2.amazonaws.com principal
```

Root Cause: IAM role's trust policy didn't allow EC2 to assume the role.
Resolution: Updated role trust policy to include `ec2.amazonaws.com` and reattached instance profile; re-run SSM agent registration.
Prevention: Validate IAM trust policies in IaC and run role-assumption checks in pre-deploy pipeline.

Case 3: Kubernetes Pod in CrashLoopBackOff

Problem: Critical service pod repeatedly crashes on startup on EKS cluster.
Investigation:

```
$ kubectl get pods -n prod
api-1 CrashLoopBackOff 5 (3m ago)
$ kubectl logs api-1 -n prod --previous
panic: failed to connect to redis: dial tcp 10.0.2.5:6379: connect: connection
refused
```

Root Cause: Application started before Redis service was ready; no startup probe configured.
Resolution: Added readiness/startup probes and an initContainer to wait for Redis; redeployed.
Prevention: Implement proper probes and dependency handling; use liveness/startup probes in manifests.

Case 4: Ansible Playbook Timeout (Unreachable Host)

Problem: Ansible playbook fails with 'UNREACHABLE!' for many hosts during deployment.
Investigation:

```
$ ansible-playbook site.yml -vvv
fatal: [web-03]: UNREACHABLE! => SSH Error: Connection timed out
$ ping -c 3 web-03
Request timed out
```

Root Cause: Temporary network partition and SSH rate-limiting on bastion host caused connection timeouts.
Resolution: Rerouted traffic through alternate bastion and re-ran playbook; increased SSH retries in ansible.cfg.
Prevention: Add orchestration retries, use connection pools, and monitor bastion resource utilization.

Case 5: Auto Scaling Group Not Triggering on Load Spike

Problem: Traffic spike caused instances to saturate but ASG didn't scale out.
Investigation:

```
$ aws cloudwatch get-metric-statistics --metric-name CPUUtilization --namespace
AWS/EC2 ...
Average CPU: 85%
$ aws autoscaling describe-policies --auto-scaling-group-name app-asg
Scaling policy exists but last-invocation shows no Enqueue
```

Root Cause: CloudWatch alarm used wrong metric namespace and threshold due to template bug.
Resolution: Corrected CloudWatch alarm to use ASG aggregated metric and tested scaling with `put-metric-alarm`.
Prevention: Test autoscaling with synthetic load in staging and alert on missing actions.

Case 6: CloudWatch Alarms Not Firing Due to Misconfigured Thresholds

Problem: Monitoring alerts silent during partial outage.
Investigation:

```
$ aws cloudwatch describe-alarms --alarm-names 'HighLatency'
Threshold: 5000ms, EvaluationPeriods: 1
$ grep 'Latency' /var/log/app | tail -n 20
Requests averaging 1200ms
```

Root Cause: Alarm threshold set too high (ms vs seconds mismatch) causing no alerts.
Resolution: Adjusted alarm threshold to correct units and reconfigured SNS targets.
Prevention: Standardize metric units in dashboards and include runbooks for alarm tuning.

Case 7: CI/CD Deployment Rollback Fails

Problem: Automated rollback didn't restore previous stable release, leaving partial deploy.
Investigation:

```
$ kubectl rollout undo deployment/api --to-revision=3
error: revision 3 not found
$ git log --oneline -n 5
Previous container image tag missing in registry
```

Root Cause: Image registry garbage collection removed previous image tag referenced by rollback.
Resolution: Rebuilt and republished the stable image tag and performed rollback; update deployment to use immutable tags.
Prevention: Keep immutable image tags for releases and retain previous images for rollback window.

Case 8: S3 Bucket Access Denied for Automation User

Problem: Automation job fails with AccessDenied when accessing S3 artifacts.
Investigation:

```
$ aws s3 cp s3://ci-artifacts/build.tar.gz .
An error occurred (AccessDenied) when calling the HeadObject operation
```

Root Cause: IAM policy missing `s3:GetObject` for the specific prefix due to overly restrictive condition.
Resolution: Updated IAM policy to allow required actions on the prefix and verified with `aws sts get-caller-identity`.
Prevention: Use least-privilege policies but test role permissions in staging; document required S3 prefixes.

Case 9: EKS Node Not Joining Cluster

Problem: New EC2 node launched by ASG fails to register with EKS control plane.
Investigation:

```
$ kubectl get nodes
No new node appears
On node: `kubelet` logs show: 'certificate signing request pending'
```

Root Cause: Kubelet CSR not auto-approved; cluster RBAC or approver controller misconfigured.
Resolution: Manually approved CSR to bring node up and fixed controller configuration to auto-approve bootstrap CSRs.
Prevention: Ensure bootstrap tokens and approvers are correctly set in IaC; automate CSR handling securely.

Case 10: Cloud-init Failure on Instance Launch

Problem: Instances show 'cloud-init failed' and never complete initialization.
Investigation:

```
$ cloud-init status --long
status: failed
$ tail -n 50 /var/log/cloud-init.log
Error: No valid network interface found for datasource
```

Root Cause: Incorrect cloud-init datasource configuration and network interface naming mismatch.
Resolution: Corrected cloud-init datasource and userdata, re-provisioned instance with proper network config.
Prevention: Validate cloud-init userdata and network configuration in golden images and CI tests.