Rajolu Abheenash

Houston, TX | 832-891-4093 | abheenash007@gmail.com
abheenash.com | github.com/Abheenash | linkedin.com/in/abheenash

Summary

AWS Certified Solutions Architect – Associate and Cloud Practitioner with an M.S. in Computer & Systems Engineering and professional experience as a DevOps Engineer in AWS cloud operations (HCLTech), targeting roles in cloud engineering, DevOps, and cloud security. Supported a production containerized platform on AWS — weekly on-call incident response, Terraform migrations, and CI/CD automation — and designed, built, and deployed production-grade AWS projects including an AI-powered serverless application (Amazon Bedrock), a zero-knowledge encrypted file-sharing service, a Kubernetes platform on Amazon EKS, and a DevSecOps-gated containerized service — each provisioned end-to-end with Terraform and shipped through security-gated CI/CD. Backed by a systems foundation in C++, multithreading, and performance engineering.

Education

University of Houston — **Houston, TX**
Master of Science in Computer & Systems Engineering; GPA: 3.60 — *Dec. 2025*

- **Relevant Coursework:** Advanced Hardware Design, Advanced Computer Architecture, VLSI Design, Principles of Internetworking, Introduction to Cybersecurity, Open Systems, Engineering Management

VRS & YRN College of Engineering and Technology — **Chirala, India**
Bachelor of Technology in Computer Science and Engineering — *Apr. 2023*

Experience

HCLTech — **Hyderabad, India**
DevOps Engineer — AWS Cloud Operations — *Apr. 2022 – Dec. 2023*

- Supported a US client's containerized B2B platform across development, staging, and production AWS environments — multiple ECS Fargate services behind an Application Load Balancer, with RDS, Route 53, and Linux hosts — troubleshooting incidents across compute, load balancing, databases, IAM, VPC networking, and the OS layer.
- Held a weekly on-call rotation: triaged CloudWatch and PagerDuty alerts, assessed customer impact, executed documented rollbacks and recoveries, and authored root-cause analyses for production incidents.
- Rebuilt a half-manual release process into an automated GitHub Actions pipeline — tests, security scanning, image versioning, staged deploys, and a gated production rollout with automatic rollback — shortening deployments and removing console-based production changes.
- Migrated manually-created AWS resources into reusable Terraform modules with remote state, state locking, pull-request plans, and drift detection — ending configuration drift and making environments rebuildable from code.
- Improved observability by replacing static-threshold alarms with golden-signal and composite service-health alarms, each tied to a runbook — cutting non-actionable alert noise and speeding triage.
- Automated recurring operations with Python, Boto3, Lambda, EventBridge, and Systems Manager (patch-compliance reporting, non-production scheduling, resource-health checks), and remediated IAM, encryption, and configuration findings from AWS Config, Inspector, GuardDuty, and Security Hub.

Technical Skills

Cloud (AWS): Lambda, API Gateway, S3, DynamoDB, Cognito, ECS Fargate, EKS, ECR, EC2 & Auto Scaling, RDS, VPC, ALB, CloudFront, Route 53, KMS, Secrets Manager, IAM, WAF, CloudWatch, X-Ray, Synthetics, SNS, SES, EventBridge, Systems Manager (SSM), CloudTrail, GuardDuty, Config, Inspector, Security Hub, Amazon Bedrock
GenAI / LLM: Amazon Bedrock (Claude), prompt engineering, context-stuffing & prompt caching, structured extraction, RAG-style Q&A, LLM-in-the-loop automation
Infrastructure as Code & CI/CD: Terraform (modules, remote state & locking, drift detection), GitHub Actions, OIDC (keyless auth), branch protection, automated deployments & rollbacks
DevSecOps & Security: IAM least privilege, KMS/SSE encryption, client-side cryptography (WebCrypto / AES-256-GCM), zero-knowledge design, JWT / Cognito auth, Secrets Manager & secret rotation, AWS WAF, VPC isolation (private subnets, VPC endpoints), Checkov, tfsec, Trivy, gitleaks, TLS
Containers, Kubernetes & Observability: Docker, ECS Fargate, ECR, Kubernetes (Amazon EKS, HPA autoscaling, Helm, IRSA, ALB Ingress), CloudWatch (dashboards, alarms, Logs Insights, RUM), X-Ray tracing, Synthetics canaries, SLOs & error budgets, on-call incident response, runbooks & RCAs, fault-injection drills
Programming / Scripting: Python (Boto3), Bash/Shell, SQL, JavaScript, C, C++
Systems Programming: C++ multithreading (std::thread, mutexes, condition variables), OpenMP, POSIX sockets (TCP), producer–consumer & thread-pool design, CMake, performance analysis (compute- vs. memory-bound)
Networking & Systems: VPC, Subnetting, Routing, DNS, TCP/IP, Load Balancing, Linux
Web & Databases: React, Node.js, Flask, MySQL, MongoDB, REST APIs

Projects

Job Hunt Command Center | *Amazon Bedrock (Claude), Step Functions, SQS, Cognito, Lambda, DynamoDB, Terraform* · Code

- Built an AI **résumé generator**: paste a job description and **Amazon Bedrock (Claude — Sonnet/Haiku/Opus)** writes a tailored 2-page résumé (LaTeX + a server-compiled PDF via a tectonic Lambda layer), scored with a weighted match rubric + ATS keyword rate — the model returns structured JSON the Lambda renders into LaTeX deterministically, so output always compiles and never fabricates beyond my real corpus.
- Engineered an **event-driven** email-intelligence pipeline (EventBridge → SQS+DLQ → **Step Functions**) where Bedrock reads the inbox (read-only IMAP), classifies recruiter replies / interviews / rejections, and **auto-advances and enriches** the matching application — per-message retries + poison-message DLQ isolation. Full-stack serverless behind Cognito (API Gateway JWT → Lambda → DynamoDB + versioned S3), all Terraform + DevSecOps CI (gitleaks, Checkov/tfsec, Trivy).
- Gave the app its own **observability** (CloudWatch golden-signals dashboard, SLO + composite health alarms → SNS, X-Ray) and an **AWS Budgets** guard on Bedrock spend; added JD extraction, a JD↔résumé match, an "Ask-AI" Q&A over applications, and a weekly SES digest.
- Layered **job-search intelligence** on top: a **visa-sponsorship checker** that fuses JD kill-phrase scanning, curated employer lists, and a live H-1B/LCA history lookup into one apply-or-skip verdict; an **Openings Radar** that pulls entry-level cloud/DevOps roles from companies' ATS JSON boards and ranks them by fit; and an interview-prep generator — every model call least-privilege (per-model IAM) and cost-bounded by a keyword pre-filter.

Serverless File Share (Zero-Knowledge) | *WebCrypto / AES-256-GCM, API Gateway, KMS, Terraform* · Live / Code

- Designed a **zero-knowledge**, self-destructing file / secret sharing app: files are encrypted *in the browser* with AES-256-GCM (WebCrypto) before upload and the key lives only in the share link's URL fragment — so S3, the Lambdas, and the operator only ever hold ciphertext (even the filename is sealed). Live at share.abheenash.com.
- Enforced least privilege end-to-end (one scoped IAM role per Lambda, account-wide Block Public Access, HTTPS-only, short-lived presigned URLs) with SSE-KMS as defense-in-depth, and automatic expiry via DynamoDB TTL → Streams → "reaper" Lambda.
- Codified ∼38 resources as Terraform with a keyless-OIDC GitHub Actions pipeline; diagnosed and fixed SSE-KMS uploads failing until forced to AWS SigV4 and a DynamoDB Streams "LATEST" race.
- Extended the product with **secret-note** sharing, **burn-after-reading** one-time links, drag-and-drop, and QR codes, and wrote a full **threat model** documenting honest limitations — all while keeping it click-and-try with *no login*, so a recruiter can try it in seconds.

AWS EKS Platform | *Amazon EKS / Kubernetes, Terraform, ALB Ingress, IRSA, HPA, GitHub Actions (OIDC)* · Code

- Provisioned a production-shaped Kubernetes platform on **Amazon EKS** entirely in Terraform (official VPC + EKS modules): a managed node group, IRSA/OIDC, the AWS Load Balancer Controller (ALB Ingress via IRSA), Metrics Server, and a CPU Horizontal Pod Autoscaler.
- Deployed through a keyless GitHub Actions pipeline (OIDC → ECR → rollout) and ran it as a cost-controlled build → prove → destroy loop.
- Proved resilience with live drills: a deleted pod self-healed back to full capacity in ∼7 seconds, and the HPA scaled 2 → 6 pods under CPU load in ∼60 seconds — documented with honest findings (e.g., ALB deregistration behavior).
- Wired workload identity with **IRSA** so pods assume narrowly-scoped IAM roles through the cluster's OIDC provider instead of sharing node-wide credentials — the same least-privilege model backing the AWS Load Balancer Controller.

Secure Container Pipeline | *ECS Fargate, VPC, WAF, Terraform, GitHub Actions* · Code

- Built a GitHub Actions pipeline with three fail-the-build security gates — gitleaks (secrets), Checkov/tfsec (IaC misconfiguration), and Trivy (image + dependency CVEs) — enforced by branch protection; proved it by automatically blocking a PR that introduced a hardcoded AWS credential.
- Provisioned the full stack in Terraform: ECS Fargate in private subnets using VPC endpoints instead of a NAT gateway (lower cost, no internet egress), an ALB fronted by AWS WAF, DynamoDB, and Secrets Manager, with a least-privilege IAM role per task.
- Remediated real supply-chain CVEs surfaced by the pipeline and hardened the container (non-root, read-only root filesystem, HTTPS behind WAF managed + rate-based rules).
- Treated the demo stack like a real service — CloudWatch alarms wired to **SNS**, container insights, DynamoDB encryption with point-in-time recovery, and a documented architecture diagram — then *destroyed* it between runs to keep cost near zero.

Certifications

AWS Certified Solutions Architect – Associate (SAA-C03) — Amazon Web Services · Verify
AWS Certified Cloud Practitioner (CLF-C02) — Amazon Web Services · Verify
NPTEL — Social Network Analysis