Internship Experience Report: AWS Services, Cloud Operations, and Cost Optimization

Introduction

During my internship, I had the opportunity to work extensively with a wide array of AWS services and DevOps tools, contributing to the management and optimization of client infrastructures. My responsibilities spanned provisioning and managing AWS resources (EC2, S3, VPC, ECS, EKS, ECR, IAM, Aurora, RDS, AWS Organizations, IAM Identity Center, API Gateway, WAF), containerization (Docker, Kubernetes), CI/CD (Jenkins, Git), and cloud networking (VPN tunnels, Route 53, CloudFront, Certificates, CloudWatch, CloudTrail). I also handled client support tickets related to infrastructure automation, security, and cost management, gaining hands-on experience in real-world cloud operations.

Student Name: Aryaman Sharma

Reg No.: 12100937

1. Core AWS Services and Use Cases

Amazon EC2 (Elastic Compute Cloud)

- **Overview:** EC2 provides scalable, on-demand virtual servers in the AWS cloud. It allows launching, configuring, and managing instances with various CPU, memory, storage, and networking configurations.
- **Key Features:**
    - Multiple instance types for different workloads (general purpose, compute-optimized, memory-optimized).
    - Integration with Amazon Machine Images (AMIs) for rapid deployment.
    - Security groups and key pairs for secure access.
- **Use Cases:**
    - Hosting web applications, backend servers, and batch processing.
    - Autoscaling to handle traffic spikes (e.g., e-commerce sales events).
    - Running Jenkins masters/agents for CI/CD pipelines.
- **Example:** For a client's e-commerce platform, I set up EC2 auto-scaling groups to ensure high availability and cost efficiency during traffic surges.

Amazon S3 (Simple Storage Service)

- **Overview:** S3 is a highly scalable object storage service for data backup, archiving, content distribution, and data lakes.
- **Key Features:**
    - 11 nines of durability, lifecycle policies, versioning, and cross-region replication.
    - Fine-grained access control via IAM policies and bucket policies.
- **Use Cases:**
    - Storing website assets, logs, and backups.
    - Serving static content via CloudFront.
    - Data lakes for analytics and ML workloads.
- **Example:** Implemented S3 bucket policies for secure log storage and enabled versioning for critical business data.

Amazon VPC (Virtual Private Cloud)

- **Overview:** VPC allows the creation of logically isolated networks within AWS, supporting secure communication between resources.
- **Key Features:**
    - Subnets (public/private), route tables, NAT gateways, and security groups.
    - Integration with Site-to-Site VPN and Direct Connect for hybrid cloud.
- **Use Cases:**
    - Multi-tier web applications with isolated database subnets.
    - Hybrid cloud architectures connecting on-premises data centers to AWS.
- **Example:** Designed a VPC with public/private subnets for a SaaS application, enabling secure access to RDS and ECS services.

Amazon ECS (Elastic Container Service) and EKS (Elastic Kubernetes Service)

- **ECS:** Managed container orchestration for Docker containers, supporting both EC2 and serverless Fargate launch types.
- **EKS:** Fully managed Kubernetes service for deploying, scaling, and managing containerized applications.
- **Key Features:**
    - Integration with ECR for container images.
    - IAM roles for task-level security.
    - Autoscaling and service discovery.
- **Use Cases:**
    - Microservices deployment, batch processing, and ML workloads.
    - CI/CD pipelines using containers.
- **Example:** Migrated a legacy monolith to ECS microservices, improving scalability and deployment speed.

Amazon ECR (Elastic Container Registry)

- **Overview:** Managed Docker container registry integrated with ECS and EKS6.
- **Key Features:**
  - High availability, encryption at rest, and image replication.
  - Fine-grained access control with IAM.
- **Use Cases:**
  - Storing and managing container images for multi-region deployments.
- **Example:** Automated image builds and pushes to ECR as part of CI/CD workflows.

IAM (Identity and Access Management) and IAM Identity Center

- **Overview:** IAM controls user and service permissions, while IAM Identity Center (formerly AWS SSO) provides centralized access management across AWS accounts.
- **Key Features:**
    - Fine-grained policies, multi-factor authentication, and role-based access.
    - Integration with AWS Organizations for unified identity management.
- **Use Cases:**
    - Enforcing least-privilege access.
    - Centralized login for multi-account environments.
- **Example:** Implemented IAM roles for ECS tasks and set up IAM Identity Center for federated access.

Amazon Aurora and RDS (Relational Database Service)

- **Aurora:** High-performance, MySQL- and PostgreSQL-compatible database engine.
- **RDS:** Managed relational database service supporting multiple engines (MySQL, PostgreSQL, SQL Server, Oracle).
- **Key Features:**
    - Automated backups, multi-AZ deployments, and read replicas.
    - Encryption at rest and in transit.
- **Use Cases:**
    - Transactional applications, analytics, and reporting.
- **Example:** Deployed Aurora clusters for a high-traffic SaaS platform, leveraging read replicas for scaling.

AWS Organizations and Service Control Policies (SCPs)

- **Overview:** AWS Organizations enables centralized management of multiple AWS accounts. SCPs enforce permission guardrails across accounts9.
- **Use Cases:**
    - Enforcing security and compliance policies.
    - Restricting root user actions and sensitive operations (e.g., S3 deletions).
- **Example:** Used SCPs to deny S3 bucket deletions and restrict EC2 actions for non-production accounts.

API Gateway and WAF (Web Application Firewall)

- **API Gateway:** Managed service for creating, deploying, and managing APIs, with pay-per-use pricing16.
- **WAF:** Protects web applications from common threats (SQL injection, XSS).
- **Use Cases:**
    - Building RESTful APIs for mobile/web apps.
    - Protecting APIs and web apps from attacks.
- **Example:** Deployed API Gateway with WAF for a fintech client, enabling secure, scalable API access.

---

2. Containers, DevOps, and CI/CD

Docker and Kubernetes

- **Docker:** Used for containerizing applications, ensuring consistency across development and production.
- **Kubernetes (K8s):** Orchestrates deployment, scaling, and management of containerized workloads.
- **Use Cases:**
    - Microservices, batch processing, and ML pipelines.
    - Hybrid and multi-cloud deployments with EKS Anywhere5.
- **Example:** Built Docker images for Node.js and Python apps, deployed to EKS clusters with rolling updates.

Jenkins, Git, and CI/CD Pipelines

- **Jenkins:** Automated build, test, and deployment pipelines, integrated with AWS services[7].
- **Git:** Source code management and version control.
- **AWS CI/CD Tools:** CodePipeline, CodeBuild, CodeDeploy for end-to-end automation[8].
- **Use Cases:**
    - Continuous integration and delivery for web/mobile applications.
    - Automated infrastructure deployments (IaC).
- **Example:** Set up Jenkins pipeline to build Docker images, push to ECR, and deploy to ECS with blue/green deployments.

---

3. Cloud Networking and Security

VPN Tunnels and Site-to-Site VPN

- **Overview:** Secure, encrypted tunnels connecting on-premises networks to AWS VPCs19.
- **Key Features:**
  - Redundant tunnels for high availability.
  - IPsec encryption for data confidentiality and integrity.
- **Use Cases:**
  - Hybrid cloud connectivity for data center integration.
  - Disaster recovery and secure remote access.
- **Example:** Configured Site-to-Site VPN for a healthcare client to securely connect their on-premises EMR system to AWS-hosted analytics.

CloudFront (CDN) and Route 53 (DNS)

- **CloudFront:** Delivers static/dynamic content globally with low latency, integrates with S3, EC2, and Lambda@Edge10.
- **Route 53:** Managed DNS service with domain registration, health checks, and 100% SLA11.
- **Use Cases:**
  - Global content delivery for media and web assets.
  - DNS-based failover for high availability.
- **Example:** Used CloudFront for video streaming and Route 53 for DNS failover between AWS regions.

Certificates and AWS Certificate Manager (ACM)

- **Overview:** ACM automates SSL/TLS certificate provisioning, deployment, and renewal12.
- **Use Cases:**
  - Securing web applications and APIs.
  - Enabling HTTPS for CloudFront, ELB, and API Gateway.
- **Example:** Automated certificate management for a multi-domain SaaS platform, reducing manual renewal errors.

CloudWatch and CloudTrail

- **CloudWatch:** Monitors AWS resources and applications, providing metrics, logs, and alarms.
- **CloudTrail:** Tracks API calls and user activity for auditing and compliance.
- **Use Cases:**
  - Real-time monitoring of EC2, RDS, Lambda, etc.
  - Security auditing and incident response.
- **Example:** Set up CloudWatch alarms for CPU/memory thresholds and used CloudTrail logs for investigating security incidents.

---

4. Handling Client Tickets and Infrastructure Operations

- **Ticket Resolution:** Addressed client tickets related to infrastructure provisioning, troubleshooting, security incidents, and performance optimization.
- **Common Issues:**
    - EC2 instance failures, S3 access errors, VPC connectivity issues.
    - Jenkins build failures, CI/CD pipeline errors.
    - Certificate expirations, DNS misconfigurations, and CloudFront cache invalidations.
- **Examples:**
    - Resolved a Route 53 DNS propagation issue affecting client website uptime.
    - Automated S3 bucket policy updates for compliance.
    - Investigated and remediated CloudTrail alerts for unauthorized access attempts.

---

5. AWS Pricing Models and Cost Optimization

AWS Pricing Fundamentals

- **Pay-as-You-Go:** Only pay for what you use, with no upfront commitments.
- **Reserved Instances:** Commit to 1- or 3-year terms for EC2, RDS, and other services, offering significant discounts for predictable workloads.
- **Savings Plans:** Flexible pricing model providing discounts for committed usage across EC2, Fargate, and Lambda, with up to 72% savings.
- **Volume Discounts:** Lower per-unit costs as usage increases.

Service-Specific Pricing Examples

Service	Pricing Model	Example/Use Case
EC2	On-demand, Reserved, Savings Plans	Reserved for always-on servers, On-demand for burst workloads
S3	Pay per GB/month, data transfer, requests	Lifecycle policies to move infrequent data to Glacier
RDS/Aurora	On-demand, Reserved	Reserved for production DBs, On-demand for dev/test
ECS/EKS/ECR	Per vCPU/hour, per pod/task, storage	Savings Plans for predictable container workloads

API Gateway	Per API call, data transfer	REST APIs for mobile apps, HTTP APIs for microservices
CloudFront	Per GB delivered, requests	Media streaming, global asset delivery
Route 53	Per hosted zone, per DNS query	Multi-region DNS failover
VPN	Per hour per connection, data transfer	Hybrid cloud, disaster recovery

Cost and Usage Reports (CUR)

- **Overview:** AWS CUR provides detailed, customizable reports on usage and costs, broken down by service, region, account, or tag[15].
- **Benefits:**
  - Hourly, daily, or monthly granularity.
  - Integration with AWS Budgets and Cost Explorer for proactive management.
  - Essential for chargeback, showback, and cost optimization.
- **Example:** Used CUR to analyze S3 storage costs across projects, identifying unused buckets for lifecycle policy enforcement.

Reservations, Savings Plans, and CUR in Practice

- **EC2:** Purchased Reserved Instances for baseline workloads, used Savings Plans for flexible compute needs, and tracked utilization with CUR.
- **ECS/EKS:** Leveraged Savings Plans for container compute, monitored with CUR to optimize pod/task usage.
- **RDS/Aurora:** Reserved Instances for production databases, On-demand for test environments, tracked with CUR for right-sizing.
- **API Gateway/CloudFront:** Used CUR to monitor API request and CDN delivery costs, optimizing cache policies and endpoint usage.

---

6. Real-World Scenarios and Examples

Scenario 1: Hybrid Cloud with VPN Tunnels

A retail client required secure connectivity between their on-premises ERP system and AWS-hosted analytics. I set up a Site-to-Site VPN with redundant tunnels using AWS VPC, ensuring high availability and encrypted data transfer19. Integrated CloudWatch for tunnel health monitoring and used CUR to track VPN data transfer costs20.

Scenario 2: Multi-Account Governance with AWS Organizations

For a fintech company, I implemented AWS Organizations with SCPs to enforce security policies across production and development accounts. IAM Identity Center provided centralized access, and CUR helped allocate costs by account and project.

Scenario 3: CI/CD Automation with Jenkins and ECS

Built a CI/CD pipeline using Jenkins (hosted on EC2), Git, and ECS. Automated Docker image builds, ECR pushes, and ECS deployments. Used CloudWatch for monitoring and CUR to track pipeline resource consumption.

Scenario 4: Cost Optimization for S3 and RDS

Analyzed CUR data to identify infrequently accessed S3 objects and implemented lifecycle policies to transition them to Glacier, reducing storage costs. For RDS, right-sized instances and reserved capacity for production databases.

---

7. AWS Pricing Model Deep Dive

On-Demand vs. Reserved vs. Savings Plans

- **On-Demand:** Pay for compute/storage/networking by the hour/second/GB. Ideal for unpredictable workloads.
- **Reserved Instances:** Up to 75% savings for 1- or 3-year commitments. Best for steady-state usage (e.g., always-on databases).
- **Savings Plans:** Flexible, covering EC2, Fargate, and Lambda. Commitment-based, but can shift instance families, regions, or OS types14.

Container Pricing Models

- **Per Task/Pod:** ECS/EKS tasks or pods billed by the second, with 1-minute minimums17.
- **Monthly/Contract:** Fixed monthly or annual pricing for container products in AWS Marketplace.
- **Custom Metrics:** Pricing based on custom usage dimensions (e.g., users, data processed).

CUR in Cost Optimization

- **Granular Insights:** Break down costs by service, resource, or tag.
- **Proactive Management:** Identify underutilized resources, optimize reservations, and monitor trends.
- **Integration:** CUR data feeds into Cost Explorer and Budgets for alerts and forecasting15.

Conclusion

My internship provided a comprehensive, hands-on understanding of AWS cloud services, DevOps practices, and cost optimization strategies. I contributed to client success by designing secure, scalable, and cost-effective cloud architectures, automating deployments, and resolving operational issues. Leveraging tools like CUR, Savings Plans, and AWS Organizations, I helped clients achieve both technical excellence and financial efficiency in the cloud.