cloudonaut

AWS Cost Optimization

- EC2
  - Purchase Savings Plans for baseline capacity
  - Identify and terminate unused instances
  - Verify that instance type still reflects the current workload
  - Verify that the maximum I/O performance of the instance matches with the EBS volumes
  - Use Spot Instances for stateless and non-production workloads
  - Switch to latest instance types
  - Make use of AMD or ARM based instances
  - Switch to Amazon Linux or any other Operating System that is Open Source
- EBS
  - Delete snapshots created to backup data that are no longer needed
  - Check whether your backup solution deletes old snapshots
  - Delete snapshots belonging to unused AMIs
  - Search for unused volumes and delete them
- S3
  - Delete unnecessary objects and buckets
  - Consider using S3 Intelligent Tiering
  - Configure lifecycle policies define a retention period for objects
  - Use Glacier Deep Archive for long-term data archiving
- VPC
  - Create VPC endpoints for S3 and DynamoDB
  - Check costs for NAT gateways and change architecture if necessary
  - Check costs for traffic between AZs and reduce traffic when possible
  - Try to avoid VPC endpoints for other services
- CloudWatch
  - Configure a retention period for all log groups
  - Check costs for metrics API calls caused by 3rd party tools
  - Delete needless alarms and dashboards
  - Identify unnecessary custom metrics
  - Check costs for log ingestion
- Serverless
  - Optimize memory configuration for Lambda functions
  - Use Provisioned Concurrency to reduce costs for high traffic Lambda functions
  - Evaluate HTTP APIs as an alternative to API Gateway
- ECS
  - Use Fargate instead of EC2 instances
  - Purchase Savings Plans for Fargate
  - Use ECS Capacity Provider to scale the fleet of EC2 instances
  - Use Fargate Spot for non-production workloads
- RDS
  - Enable RDS Storage Auto Scaling instead of overprovisioning storage capacity
  - Consider switching to Aurora Serverless for unsteady
  - Verify that instance type still reflects the current workload
  - Verify that the maximum I/O performance of the compute layer matches with the storage layer
  - Switch to an Open Source database system (e.g., PostgreSQL instead of Oracle)
- DynamoDB
  - Switch to On-demand capacity mode for unsteady workloads
  - Use auto-scaling to adjust the provisioned capacity to the workload
- Elasticsearch
  - Make use of Reserved Instances were planning ahead is feasible
  - Evaluate UltraWarm tier (Preview) to retain large amounts of data at lower costs
- Route 53
  - Increase TTL for records to reduce queries
  - Consolidate resolver endpoints
- CloudFront
  - Check hit/miss ratio of cache and adjust configuration and TTL accordingly
  - Restrict access to S3 by using an Origin Access Identity
- CloudTrail
  - Delete unnecessary trails
  - Check costs for data events (S3 and Lambda)

Tools

- Cost Explorer
  - Aggregate costs by service
    - Which services cause the highest costs?
    - Does the cost per service match with your assumptions?
  - Identify unexpected trends
    - Are costs increasing by a similar amount each month for a specific service?
    - Any high cost increases not caused by changes to your cloud infrastructure?
- Monthly Bill
  - Drill down into service costs
    - Which resources of a service cause high costs? Justify the costs.
    - Do the costs per service match with your estimations?
    - Are there any hints for costs caused by unused resources?
- Budgets
  - Monitor costs by creating a budget alarm on forecasted costs
  - Monitor your Savings Plans/Reserved Instances by creating utilization budget alarms