Serverless Overviews

Uday Manchanda

August 29, 2021

1 Serverless

- devs don't manage servers, just deploy code
- infrastructure is still there but the dev doesn't worry about it
- lambda, dynamodb, cognito, api gateway, s3, sns/sqs, kinesis, step functions, and fargate

2 Lambda

- virtual functions - no servers to manage
- short executions, run on-demand, scaling is automated
- increasing RAM will improve CPU and network
- lambda container image - the image must implement the lambda runtime API
- ECS/Fargate is preferred for running arbitrary Docker images

2.1 Lambda Integrations

- API gateway = create REST API
- Kinesis = data transformations
- DynamoDB = insert data into DB
- EX: serverless thumbnail creation
  - New image uploaded in S3
  - Trigger s3 event notification a lambda function which creates a thumbnail
  - push the new thumbnail to S3
  - OR insert metadata into dynamoDB

- EX: serverless CRON job
  - CW event rule to trigger every 1 hr
  - integrated with a lambda function

2.2 Lambda Limits

- Execution
  - Memory: 128 MB - 10GB
  - Time: 900 seconds
  - Env vars 4KB
  - Disk capacity in the "function container": 512 MB
  - Concurrency: 1000
- Deployment
  - Deployment size compressed: 50mb
  - Uncompressed: 250mb

2.3 Lambda@Edge

- Deploy lambda functions alongside each region in the world with a CloudFront CDN
- Build more responsive applications, no need to manage servers, customize CDN content
- Can use lambda to change CloudFront requests and responses
  - After CF receives request from a viewer (viewer request)
  - Before CF forwards request to the origin (origin request)
  - After CF receives the response from the origin (origin response)
  - Before CF forwards the response to the viewer (viewer response)
- EX: Global application, static website hosted on S3, user visits website, dynamic API to CF

3 DynamoDB

- Serverless DB, NoSQL - not relational
- Scales to massive workloads, distributed database
- Fast and consistent in performance (low latency)

- Made of tables, each table with a primary key
- Each table can have an infinite number of items (rows)
- Each item has attributes
- Provisioned Throughput
  - Table must have provisioned read and write capacity units
  - Read Capacity Units (RCU): throughput for reads
  - 1 RCU = 1 strongly consistent read of 4kB/sec
  - 1 RCU = 2 eventually consistent read of 4kb/sec
  - Write Capacity Units (WCU): throughput for writes
  - 1 WCU = 1 write of 1kb/sec
  - Can set up auto-scaling of throughput to meet demand

3.1 Advanced Features

- DAX = DynamoDB accelerator, seamless cache for DynamoDB
- Streams, changes in DynamoDB can end up in a DynamoDB stream. Stream can be read by Lambda
- Transactions - all or nothing type of operations
- On demand - no capacity planning needed, automatic scaling
- Global tables - cross region replication

4 API Gateway

- Create REST APIs for a client
- clients talk to API Gateway which will then proxy the request to the lambda function
- API Gateway + Lambda: no infra to manage
- Support for Web Socket protocol
- Transform and validate responses/requests
- Integrations
  - Lambda
  - HTTP - expose HTTP endpoints in the backend

    - AWS Service - expose any AWS API through API Gateway (EX: start AWS Step function)

- Endpoint Types
    - Edge Optimized: for global clients, is the default
    - Regional: for client in the same region
    - Private: only accessed from your VPC

4.1 Security

- IAM Permissions - create IAM policy authorization and attach to user/role
- API Gateway verifies IAM permissions
- Lambda Authorizer - use Lambda to validate the token in header being passed
- Cognito User Pools - cognito fully manages user lifecycle, API Gateway verifies identity automatically from AWS Cognito. No need to write custom code

5 AWS Cognito

- We want to give our users an identity so they can interact with our application
- Cognito User Pools - sign in functionality
- Cognito Identity Pools - Federated identity, allow users to access AWS resources directly
- Cognito Sync - Synchronize data from device to Cognito

6 Serverless Application Model

- Framework for developing and deploying serverless application
- all the configuration is YAML
- Helps you run serverless stuff locally
- Can use code deploy to deploy lambda functions