Decoupling Workflows

>Decoupling Overview

- Tightly coupled architecture
  - This is where a single component fails or changes. It causes issues to other components or the system. This reminds me of UDP protocol where the tightly coupled isn't reliable since you can't get everything or nothing at all.
- Loosely coupled architecture
  - This is where a single failure won't cause cascading failures.
  - All levels of your architecture need to be loosely coupled, internally and externally

If application A sends a message to app B. And eventually app B fails, app A does not drop the messages that were sent. They wait in the queue and wait for app B to come back online.

2 AWS services that assist in this…

- Amazon SQS
  - Send, store, receive messages between software components at any volume
- Amazon SNS
  - Is a publish/subscribe service. Using Amazon SNS topics, a publisher publishes messages to subscribers.
  -
- **Simple Queue Service (SQS)**
  - Amazon Simple Queue Service (SQS) is a fully managed message queuing service that enables you to decouple and scale microservices, distributed systems, and serverless applications.
  - Also allows asynchronous (Not direct communication) processing of work.
  - QS eliminates the complexity and overhead associated with managing and operating message-oriented middleware and empowers developers to focus on differentiating work.
  - AWS manages all ongoing operations and underlying infrastructure needed to provide a highly available and scalable message queuing service. With SQS, there is no upfront cost, no need to acquire, install, and configure messaging software, and no time-consuming build-out and maintenance of supporting infrastructure.
  - Simple Queue Service (SQS) enables developers to create decoupled, distributed applications in the cloud. SQS is a message broker that different components of your application can use to send messages to each other. SQS scales automatically to accommodate any volume. For more information, visit https://aws.amazon.com/sqs/.

With SQS there is a delivery delay which the default is 0; can be set up to 15 mins. There is also a message size limit. Which messages can be up to 256 KB in any format.

- Be default… messages will last up to 4 days by default. Can be set between 1 minute and 14 days.
- Long vs. Short

- Long polling isn’t the default, but it should be.

- SQS - Don't forget messages in queues are processed in FIFO order. - SQS - Remember that message queues support loose coupling.

FIFO guarantees that your messages will arrive in the correct order.

- **Simple notification service (SNS)**
  - Amazon Simple Notification Service (Amazon SNS) is a fully managed messaging service for both application-to-application (A2A) and application-to-person (A2P) communication.
  - Notification using Simple Notification Service. The Simple Notification Service (SNS) allows applications, users, and devices to send and receive notifications from AWS. SNS uses a publisher-subscriber model, wherein a publisher such as an AWS service generates a notification and a subscriber such as an end user receives it. The communication channel that SNS uses to map publishers and subscribers is called a topic. SNS can send notifications to subscribers via a variety of protocols that includes

Proactive notifications = SNS any push-based notifications, think SNS

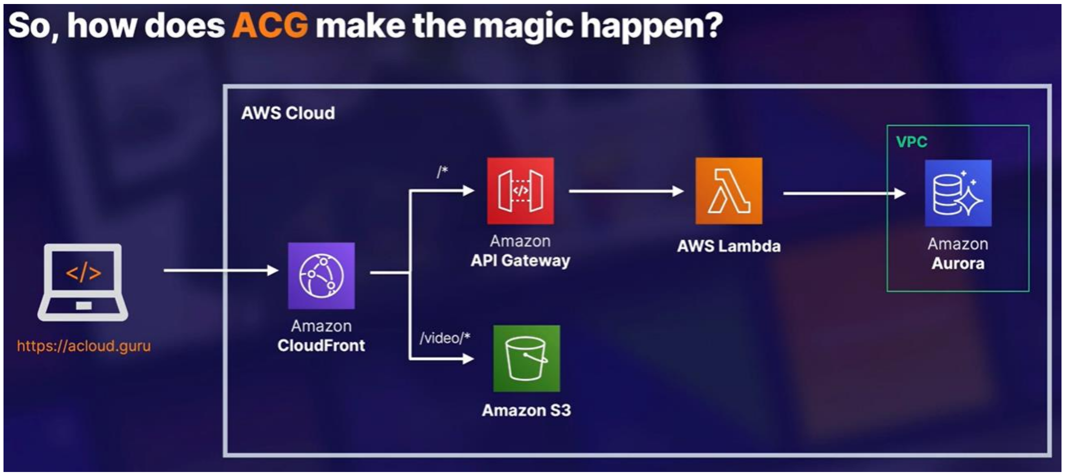
- HTTP(S)
- Simple Queue Service (SQS)
- Lambda
- Mobile push notification
- Email
- Email-JSON
- Short Message Service (SMS) text messages

- **Simple Email Service (SES)**

Is an email service that allows you to send richly formatted HTML emails from your applications
Amazon Simple Email Service (SES) is a cost-effective, flexible, and scalable email service that enables developers to send mail from within any application
SES allows you to send richly formatted HTML emails in bulk and gain valuable insights about the effectiveness of your campaign.

- **API Gateway**
  - One million API calls/month on Amazon API Gateway. API Gateway is a tool for managing your organization’s application programming interfaces (APIs). APIs allow users and applications to connect with mobile and web-based resources.
  - API Gateway sits in front of your applications
  - Allows you to build and fully manage APIs service that allows you to easily publish, create, maintain, monitor and secure your API.
  - Security
    1. This service allows you to easily protect your endpoints by attaching a web application firewall (WAF)
  - Stop abuse
    1. Users can easily implement DDoS protection and rate limiting to curb abuse.
  - Ease of Use
    1. API Gateway is simple to get started with.

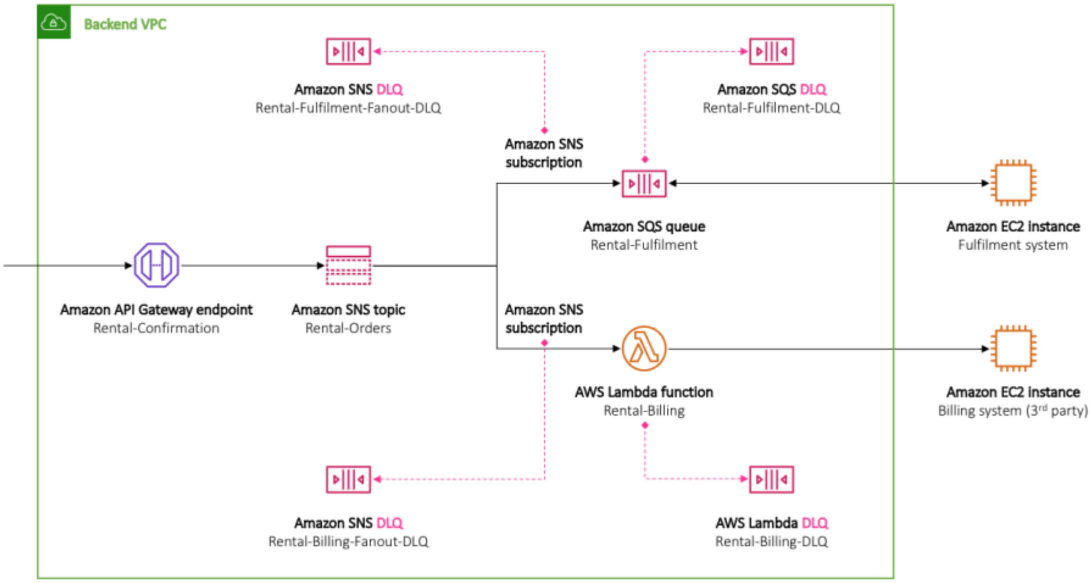
Down below is an example how acloudguru makes it happen when doing labs


So, how does ACG make the magic happen?

>Sidelining Messages with Dead-Letter Queues (DLQ)

In a postal system, a dead-letter office is a facility for processing undeliverable mail. In pub/sub messaging, a dead-letter queue (DLQ) is a queue to which messages published to a topic can be sent, in case those messages cannot be delivered to a subscribed endpoint.

- Client Errors
    - This happens when SNS has stale subscription metadata. A common error occurs when you (the endpoint owner) delete the endpoint. These errors are considered client errors because the client has attempted the delivery of a message to a destination that, from the client's perspective, is no longer accessible.
- Server Errors
    - Server errors happen when the system that powers the subscribed endpoint is unavailable, or when it returns an exception response indicating that it failed to process a valid request from SNS.



When a customer places an order to rent a car, the application sends that request to an API, which is powered by Amazon API Gateway. The REST API is backed by an SNS topic named Rental-Orders, and deployed onto an Amazon VPC subnet. The topic then fans out that order to the following two subscribed endpoints, for parallel processing:

- An SQS queue, named Rental-Fulfilment, which feeds the integration with an internal fulfilment system hosted on Amazon EC2.
- A Lambda function, named Rental-Billing, which processes and loads the customer order into a third-party billing system, also hosted on Amazon EC2.

To increase the durability of this serverless backend API, the following DLQs have been set up:

- Two SNS DLQs, namely Rental-Fulfilment-Fanout-DLQ and Rental-Billing-Fanout-DLQ, which store the order in case either the subscribed SQS queue or Lambda function ever becomes unreachable.
- An SQS DLQ, named Rental-Fulfilment-DLQ, which stores the order when the fulfilment system fails to process the order.
- A Lambda DLQ, named Rental-Billing-DLQ, which stores the order when the function fails to process and load the order into the billing system.

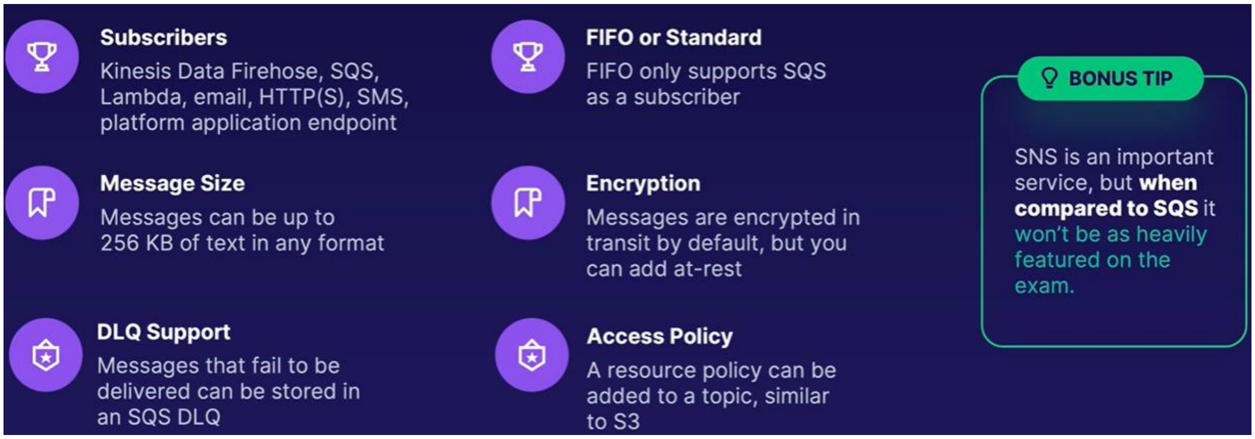
https://aws.amazon.com/blogs/compute/designing-durable-serverless-apps-with-dlqs-for-amazon-sns-amazon-sqs-aws-lambda/

More information listed above

>Delivering messages with SNS

DLQs for SNS, SQS, and Lambda increase the resiliency and durability of your applications. These DLQs address different failure modes and can be used together.

- SNS DLQs store messages that failed to be delivered to subscribed endpoints.
  - SNS is a pushed-based messaging service. Proactively deliver messages to the endpoints subscribed to it.



Subscribers
Kinesis Data Firehose, SQS, Lambda, email, HTTP(S), SMS, platform application endpoint

Message Size
Messages can be up to 256 KB of text in any format

DLQ Support
Messages that fail to be delivered can be stored in an SQS DLQ

FIFO or Standard
FIFO only supports SQS as a subscriber

Encryption
Messages are encrypted in transit by default, but you can add at-rest

Access Policy
A resource policy can be added to a topic, similar to S3

BONUS TIP
SNS is an important service, but **when compared to SQS** it won't be as heavily featured on the exam.

- SQS DLQs store messages that the consumer system failed to process.
  - Standard
    - Best-effort ordering
    - Duplicate messages
    - Nearly unlimited transactions per second
  - FIFO (First-In-First-Out)
    - Guaranteed ordering
    - No message duplication
    - 300 messages per second
- Lambda DLQs store the messages that resulted in failed asynchronous executions of your functions.

Setting up DLQs for subscriptions, queues, and functions can be done using the AWS Management Console, SDK, CLI, API, or CloudFormation. DLQs are available in all AWS Regions.

>Fronting Applications with API gateway

Exam Tips

2. Decoupling Workflows
    - Always Loosely Couple
        - Even when not specifically asked, loose coupling is the answer
    - Internal and External
        - Every level of an application should be loosely coupled
    - Never Tightly couple
        - Don't select answers that include instance-toinstance communication.
    - One size
        - Doesn't fit all. There's no one single way to decouple
3. Ordered Messages with SQS FIFO
    - Performance
        - FIFO queues do not have the same level of performance
    - Not the only way
        - You can order messages with SQS standard, but it is on you to do it
    - Message group ID
        - This ensures messages are processed one by one
    - Cost
        - It cost more since AWS must send compute power to deduplicate messages
4. Delivering messages with SNS
    - Push
        - Anytime you see this word, thing SNS
    - CloudWatch
        - SNS and CloudWatch are best friends and the easiest way to alert you that something happened
    - Where does it go?
        - You'll need to know all the subscriber options
    - No Do-Overs
        - SNS will only retry HTTP(s) endpoints, nothing else
5. API Gateway
    - API
        - Anytime the exam talks about creating or managing an API think API gateway
    - DDoS
        - You can front API gateway with a WAF
    - Versioning
        - API gateway supports versioning of your API
    - No Baking
        - Using API gateway stops you from baking credentials into your code