Message Queues

Cloud Computing

Brenden west

Contents

Learning Outcomes

- Overview of Async Processing
- Using AWS Simple Queue Service (SQS)
- Using AWS Simple Notification Service (SNS)

Reading

- AWS Cloud Developing - Module 10 - Messaging Services

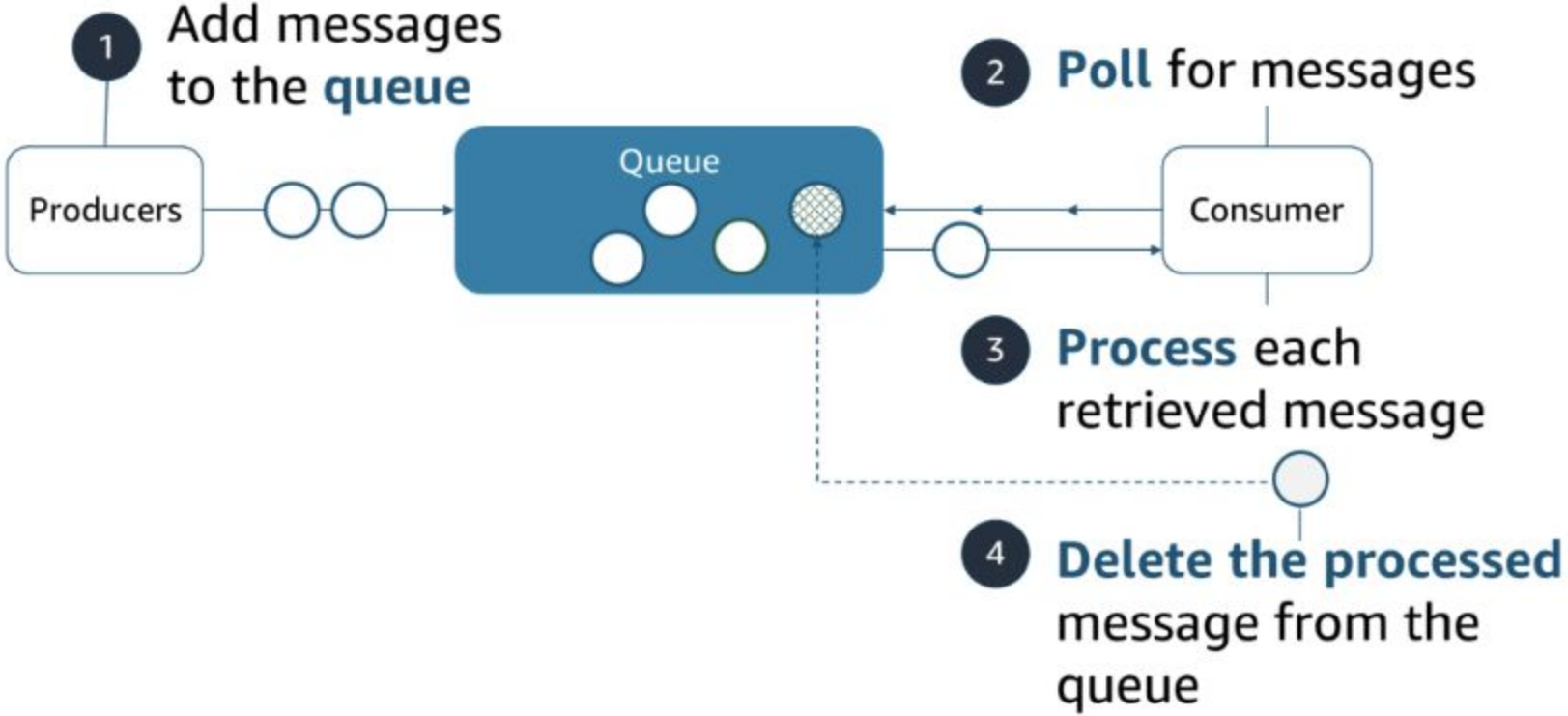
Asynchronous Processing

- Client can make a request & doesn't need to wait for results
- Client can continue making requests independent of consumer load
- System components are loosely coupled and more resilient to failure
- Service can't respond directly to client, so errors need to be handled differently from synchronous system
- AWS has 3 types of fully managed async services:
  - Message queues (AWS SQS)
  - Pub/sub messaging (AWS SNS)
  - Data streams (AWS Kinesis)

Message Queues

- A temp repo for messages waiting to be processed by a consumer
- Producers **poll** the queue for new messages & delete a message if it was successfully processed
-

Message queues overview



AWS SQS

- Fully managed message queue service
- Limited admin overhead with dynamic scaling
- Reliable message delivery without data loss
- Messages can be encrypted
- Automatically deletes messages in a queue longer than message retention period (default = 4 days)
- Offers standard and FIFO options
    - **Standard** - provides best-effort ordering & at-least-once delivery. May deliver >1 copy of a message. Nearly unlimited transactions per second
    - **FIFO** - message order preserved, exactly-once delivery. More limited throughput

AWS SQS, cont.

- Requires credentials to authenticate requests, with permissions to access queues & messages
- Can use IAM policies or SQS policies
- Can use **server-side encryption** (SSE) to encrypt incoming messages before storing on a queue and decrypt on delivery to a consumer

AWS SQS message lifecycle

- SQS receives messages from a producer & stores redundantly across multiple servers
- SQS responds with **MD5** value producer can use to verify message was received correctly, and a system-assigned MessageId
- Consumer retrieves message from SQS and sets a **visibility timeout** period
- During visibility timeout period, no other consumer can retrieve the message
- First consumer processes message and deletes from queue during timeout period
- If not processed before timeout ends, another consumer might retrieve & process the message

Short & long polling

- By default, AWS SQS uses **short polling** to retrieve requested message from a **subset** of SQS servers
- With **long polling**, SQS queries all services and waits until a message is available in the queue before responding to the consumer
  - Helps reduce SQS cost by avoiding empty & false empty responses
  - Configured by **WaitTimeSeconds** parameter in ReceiveMessage request

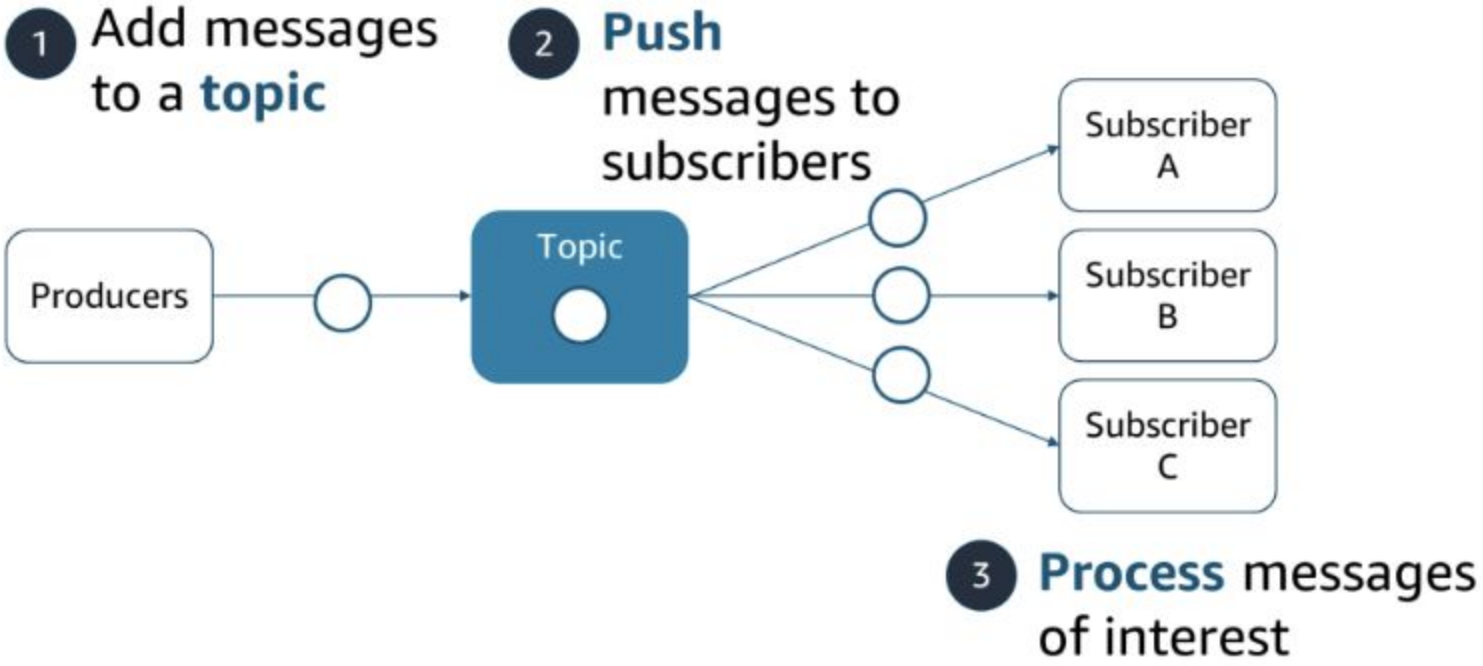
Dead-letter queue

- Common pattern for handling messages that can’t be processed / deleted by consumer
- Useful for troubleshooting delivery issues
- Can receive messages from source queue after max number of processing retries reached

Pub/sub Messaging

- Producers **publish** messages to a **topic**
- Pub/sub service broadcasts (**pushes**) messages to topic **subscribers**
- Subscribers don’t need to check for messages
- Consumers can also connect to topic as an endpoint

Pub/sub messaging overview



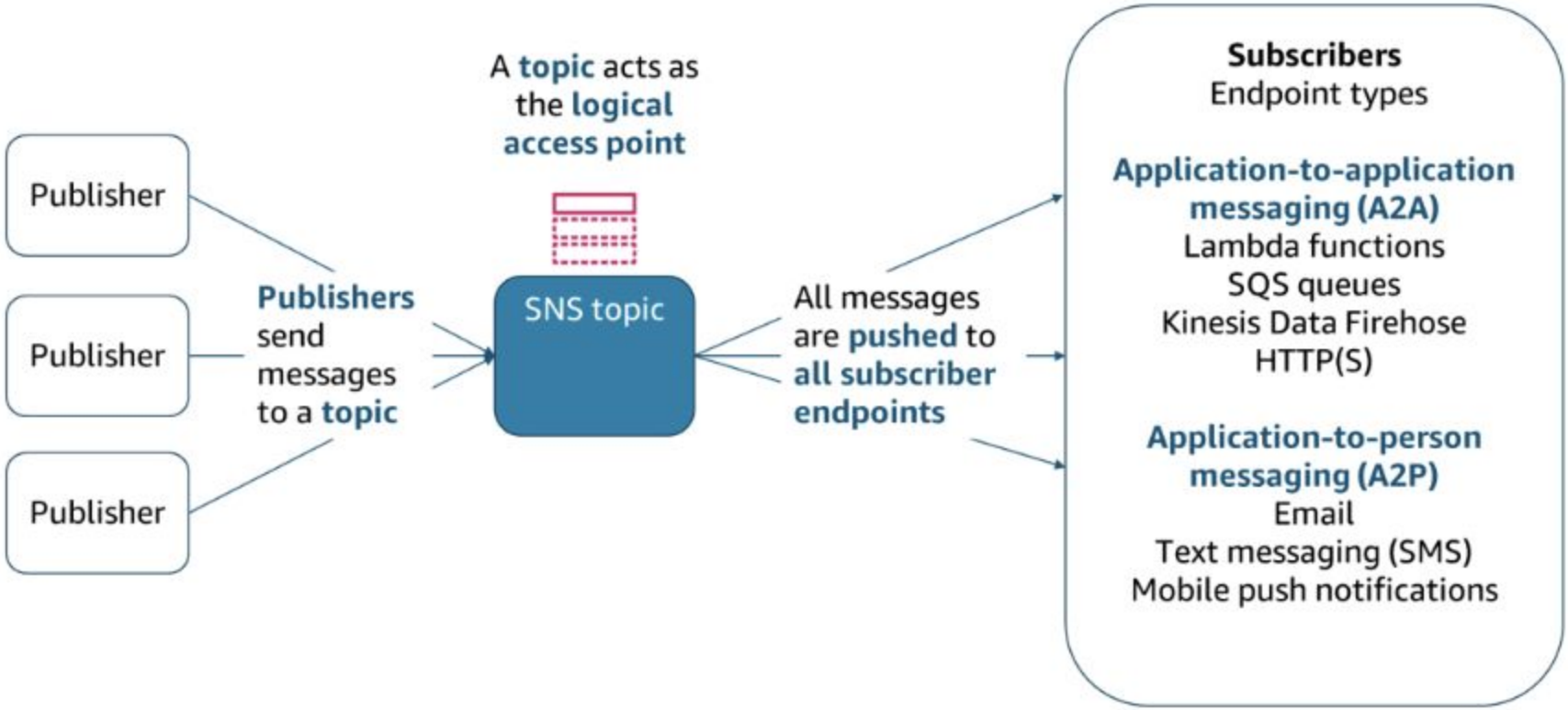
AWS SNS

- Fully managed pub/sub messaging service
- Offloads message filtering logic from subscribers and message routing logic from publishers
- Provides failure management (retries, dead-letter queues)
- Offers FIFO topic option to preserve message order
- Topic creators can define which publishers & subscribers can communicate with it
- All topic subscribers receive the same messages
- **Application-to-application (A2A)** messaging - Lambda functions, SQS queues, Kinesis data streams, etc.
- **Application-to-person (A2P)** - email, SMS, mobile push notifications

AWS SNS, cont.

- **Fanout** processing - message is pushed to multiple subscribers for parallel processing
- Subscriber can define a topic **filter policy** as a JSON object

Amazon SNS overview



Data Streams

- Similar to queues that temporarily hold messages
- Different from queues with a much higher rate of messages
- Stream may have multiple consumers
- Consumers don’t delete messages from a stream

Data streams overview