AWS Simple Queue Service – SQS

# Simple Queue Service – SQS

* Amazon SQS is a highly available distributed queue system
* A queue is a temporary repository for messages awaiting for processing and acts as a buffer between the component producer and the consumer
* SQS is a message queue service used by distributed applications to exchange messages through a **polling** model, and can be used to decouple sending and receiving components.
* Amazon SQS
  + fully managed and requires no administrative overhead and little configuration
  + offers a reliable, highly-scalable, hosted queue for storing messages in transit between applications
  + provides fault tolerant, loosely coupled, flexibility of distributed components of applications to send & receive without requiring each component to be concurrently available
  + helps build distributed application with decoupled components
  + supports the HTTP over SSL (HTTPS) and Transport Layer Security (TLS) protocols for security
* SQS provides two types of Queues – Standard & FIFO

## SQS FIFO Queue Features

Refer to Blog Post – [SQS FIFO Queue](https://jayendrapatil.com/aws-sqs-fifo-queue/)

## SQS Standard Queue Features

* **Redundant infrastructure**
  + offers reliable and scalable hosted queues for storing messages
  + is engineered to always be available and deliver messages
  + provides ability to store messages in a fail safe queue
  + highly concurrent access to messages
* **At-Least-Once delivery**
  + **ensures delivery of each message at least once**
  + stores copies of the messages on multiple servers for redundancy and high availability
  + might deliver duplicate copy of messages, if the servers storing a copy of a message is unavailable when you receive or delete the message and the copy of the message is not deleted on that unavailable server
  + Applications should be designed to be idempotent with the ability to handle duplicate messages and not be adversely affected if it processes the same message more than once
* **Message Attributes**
  + SQS message can contain up to 10 metadata attributes.
  + take the form of name-type-value triples
  + can be used to separate the body of a message from the metadata that describes it.
  + helps process and store information with greater speed and efficiency because the applications don’t have to inspect an entire message before understanding how to process it
* **Message Sampling**
  + behavior of retrieving messages from the queue depends on whether short (standard) polling, the default behavior, or long polling is used
  + With **short polling**,
    - samples only a subset of the servers (based on a weighted random distribution) and returns messages from just those servers.
    - A receive request might not return all the messages in the queue. But a subsequent receive request would return the message
  + With **Long polling**,
    - request persists for the time specified and returns as soon as the message is available thereby reducing costs and time the message has to dwell in the queue
    - long polling doesn’t return a response until a message arrives in the message queue, or the long poll times out.
    - makes it inexpensive to retrieve messages from the SQS queue as soon as the messages are available.
    - might help reduce the cost of using SQS, as the number of empty receives are reduced
* **Batching**
  + SQS allows send, receive and delete batching, which helps club up to 10 messages in a single batch while charging price for a single message
  + helps lower cost and also increases the throughput
* **Configurable settings per queue**
  + All queues don’t have to be alike
* **Order**
  + **makes a best effort to preserve order in messages does not guarantee first in, first out delivery of messages**
  + can be handled by placing sequencing information within the message and performing the ordering on the client side
* **Loose coupling**
  + removes tight coupling between components
  + provides the ability to move data between distributed components of the applications that perform different tasks without losing messages or requiring each component to be always available
* **Multiple writers and readers**
  + supports multiple readers and writers interacting with the same queue as the same time
  + locks the message during processing, using **Visibility Timeout**, preventing it to be processed by any other consumer
* **Variable message size**
  + supports message in any format up to 256KB of text.
  + messages larger than 256 KB can be managed using the S3 or DynamoDB, with SQS holding a pointer to the S3 object
* **Access Control**
  + Access can be controlled for who can produce and consume messages to each queue
* **Delay Queues**
  + delay queue allows the user to set a default delay on a queue such that delivery of all messages enqueued is postponed for that time duration
* **Dead Letter Queues**
  + Dead letter queue is a queue for messages that were not able to be processed after a maximum number of attempts
  + useful to isolate messages that can’t be processed for later analysis.
* **PCI Compliance**
  + supports the processing, storage, and transmission of credit card data by a merchant or service provider, and has been validated as being PCI-DSS (Payment Card Industry – Data Security Standard) compliant

## SQS Use Cases

* **Work Queues**
  + Decouple components of a distributed application that may not all process the same amount of work simultaneously.
* **Buffer and Batch Operations**
  + Add scalability and reliability to the architecture and smooth out temporary volume spikes without losing messages or increasing latency
* **Request Offloading**
  + Move slow operations off of interactive request paths by enqueueing the request.
* **Fan-out**
  + Combine SQS with SNS to send identical copies of a message to multiple queues in parallel for simultaneous processing.
* **Auto Scaling**
  + SQS queues can be used to determine the load on an application, and combined with Auto Scaling, the EC2 instances can be scaled in or out, depending on the volume of traffic

## How SQS Queues Works

* SQS allows queues to be created, deleted and messages can be sent and received from it
* SQS queue retains messages for four days, by default.
* Queues can be configured to retain messages for 1 minute to 14 days after the message has been sent.
* SQS can delete a queue without notification if any action hasn’t been performed on it for 30 consecutive days.
* SQS allows the deletion of the queue with messages in it

## Queue and Message Identifiers

### Queue URLs

* Queue is identified by a unique queue name within the same AWS account
* SQS assigns each queue with a Queue URL identifier for e.g. http://sqs.us-east-1.amazonaws.com/123456789012/queue2
* Queue URL is needed to perform any operation on the Queue

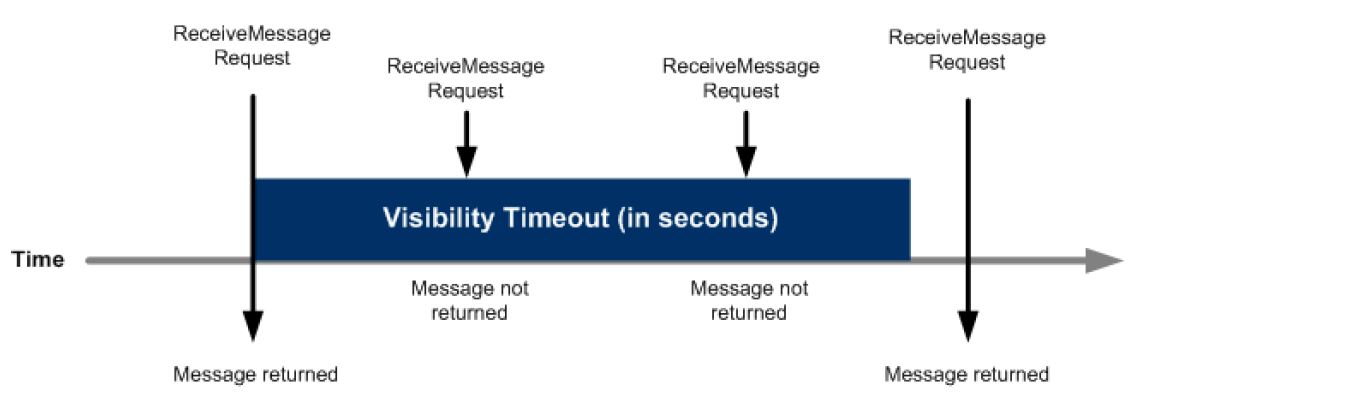
### Message ID

* Message IDs are useful for identifying messages,
* Each message receives a system-assigned message ID that SQS returns to with the SendMessage response.
* To delete a message, the message’s receipt handle instead of the message ID is needed
* Message ID can be of is 100 characters max

### Receipt Handle

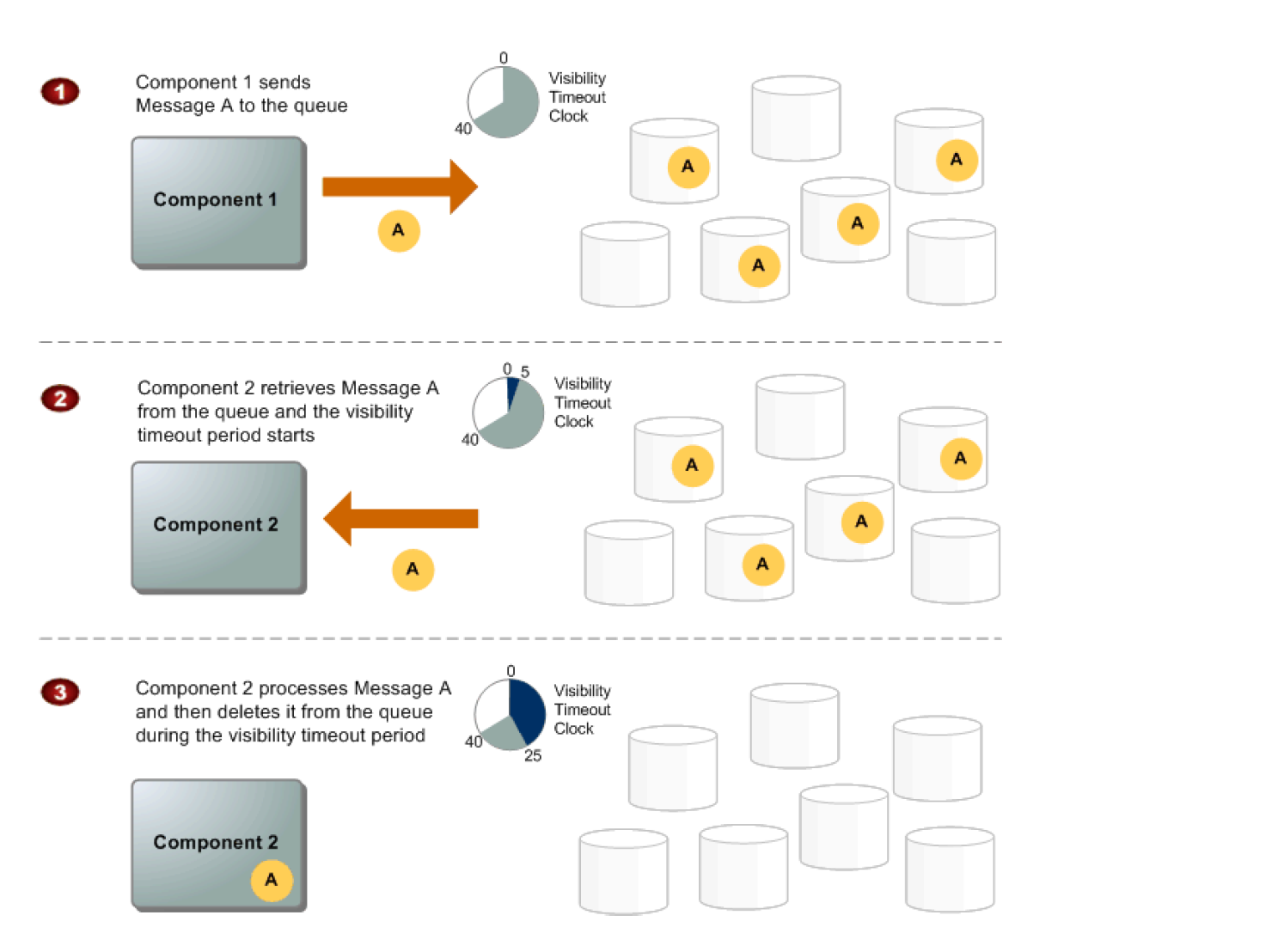
* When a message is received from a queue, a receipt handle is returned with the message which is associated with the act of receiving the message rather then the message itself
* Receipt handle is required, not the message id, to delete a message or to change the message visibility
* If a message is received more than once, each time its received, a different receipt handle is assigned and the latest should be used always

## Visibility timeout



* SQS does not delete the message once it is received by a consumer, because the system is distributed, there’s no guarantee that the consumer will actually receive the message (it’s possible the connection could break or the component could fail before receiving the message)
* Consumer should **explicitly delete** the message from the Queue once it is received and successfully processed
* As the message is still available on the Queue, other consumers would be able to receive and process and this needs to be prevented
* SQS handles the above behavior using **Visibility timeout**.
* **SQS blocks the visibility of the message for the Visibility timeout period, which is the time during which SQS prevents other consuming components from receiving and processing that message**
* **Consumer should delete the message within the Visibility timeout. If the consumer fails to delete the message before the visibility timeout expires, the message is visible again for other consumers.**
* Once Visible the message is available for other consumers to consume and can lead to **duplicate** messages.
* Visibility timeout considerations
  + clock starts ticking once SQS returns the message
  + should be large enough to take into account the processing time for each of the message
  + **default Visibility timeout for each Queue is 30 seconds and can be changed at the Queue level**
  + **when receiving messages, a special visibility timeout for the returned messages can be set without changing the overall queue timeout using the receipt handle**
  + **can be extended by the consumer, using ChangeMessageVisibility , if the consumer thinks it won’t be able to process the message within the current visibility timeout period. SQS restarts the timeout period using the new value**
  + a message’s Visibility timeout extension applies only to that particular receipt of the message and does not affect the timeout for the queue or later receipts of the message
* SQS has an 120,000 limit for the number of inflight messages per queue i.e. message received but not yet deleted and any further messages would receive an error after reaching the limit

## Message Lifecycle



1. Component 1 sends Message A to a queue, and the message is redundantly distributed across the SQS servers.
2. When Component 2 is ready to process a message, it retrieves messages from the queue, and Message A is returned. While Message A is being processed, it remains in the queue but is not returned to subsequent receive requests for the duration of the visibility timeout.
3. Component 2 deletes Message A from the queue to avoid the message being received and processed again once the visibility timeout expires.

## SQS Buffered Asynchronous Client

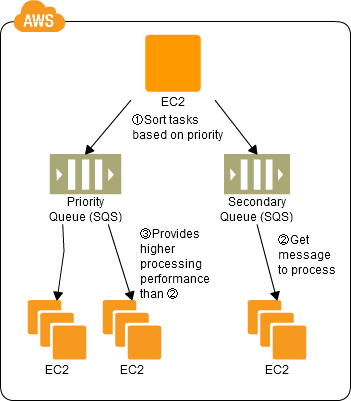
* Amazon SQS Buffered Async Client for Java provides an implementation of the AmazonSQSAsyncClient interface and adds several important features:
  + **Automatic batching** of multiple SendMessage, DeleteMessage, or ChangeMessageVisibility requests without any required changes to the application
  + **Prefetching** of messages into a local buffer that allows the application to immediately process messages from SQS without waiting for the messages to be retrieved
* Working together, automatic batching and prefetching increase the throughput and reduce the latency of the application while reducing the costs by making fewer SQS requests.

## SQS Security and reliability

* SQS stores all message queues and messages within a single, highly-available AWS region with multiple redundant Availability Zones (AZs)
* SQS supports the HTTP over SSL (HTTPS) and Transport Layer Security (TLS) protocols.
* SQS supports Encryption at Rest. SSE encrypts messages as soon as SQS receives them and decrypts messages only when they are sent to an authorized consumer.
* SQS also supports resource-based permissions

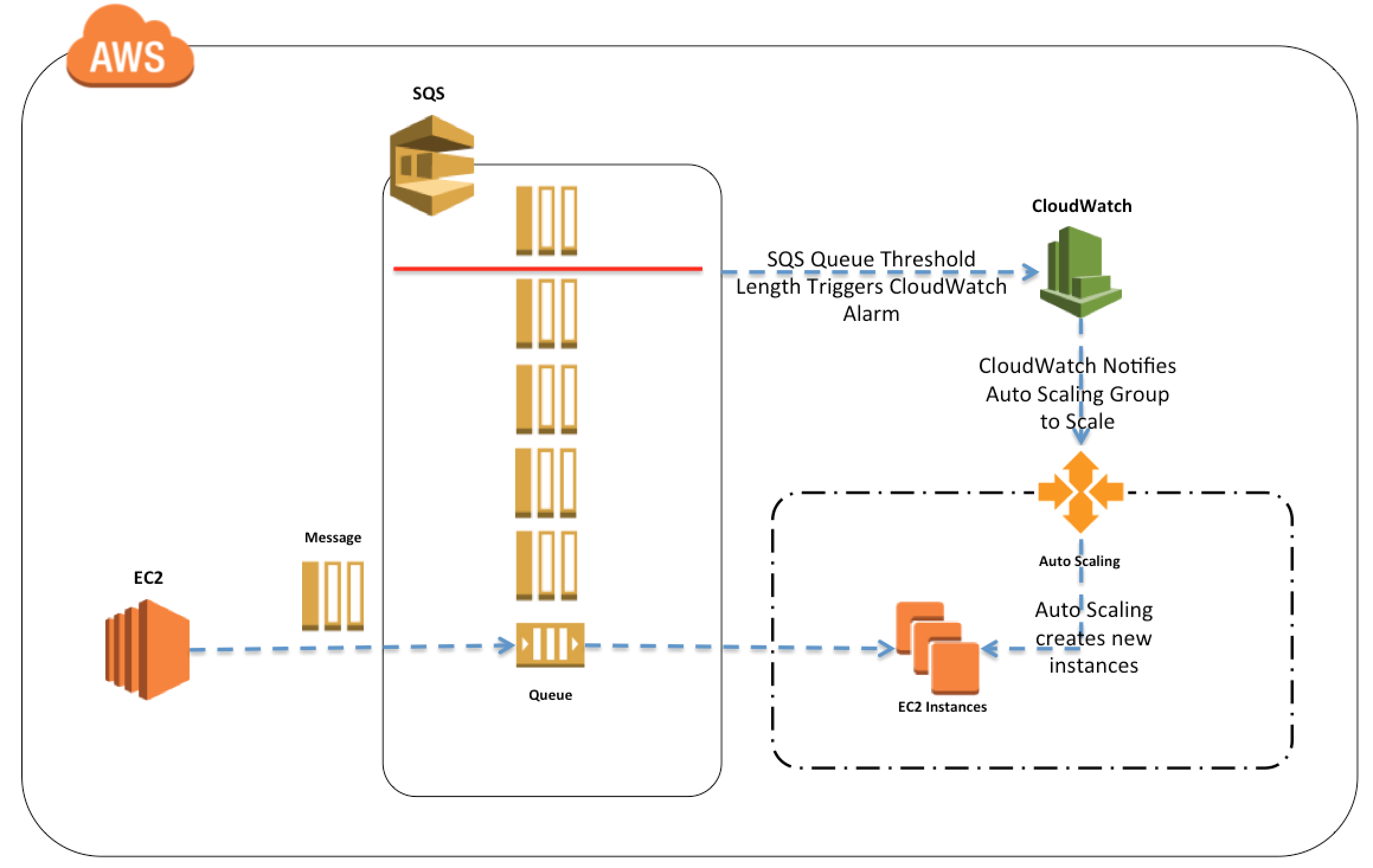
## SQS Design Patterns

### Priority Queue Pattern



1. Use SQS to prepare multiple queues for the individual priority levels.
2. Place those processes to be executed immediately (job requests) in the high priority queue.
3. Prepare numbers of batch servers, for processing the job requests of the queues, depending on the priority levels.
4. Queues have a message “Delayed Send” function, which can be used to delay the time for starting a process.

### SQS Job Observer Pattern



1. Enqueue job requests as SQS messages.
2. Have the batch server dequeue and process messages from SQS.
3. Set up Auto Scaling to automatically increase or decrease the number of batch servers, using the number of SQS messages, with CloudWatch, as the trigger to do so.

## SQS Standard Queues vs SQS FIFO Queues

Refer Blog Post – [SQS Standard Queues vs FIFO Queues](https://jayendrapatil.com/aws-sqs-standard-vs-fifo-queue/)

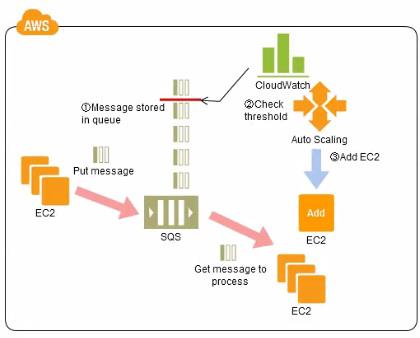
## SQS vs Kinesis Data Streams

Refer post @ [SQS vs Kinesis Data Streams](https://jayendrapatil.com/aws-kinesis-data-streams-vs-sqs/)

[do\_widget id=text-15]

## AWS Certification Exam Practice Questions

* Questions are collected from Internet and the answers are marked as per my knowledge and understanding (which might differ with yours).
* AWS services are updated everyday and both the answers and questions might be outdated soon, so research accordingly.
* AWS exam questions are not updated to keep up the pace with AWS updates, so even if the underlying feature has changed the question might not be updated
* Open to further feedback, discussion and correction.

1. Which AWS service can help design architecture to persist in-flight transactions?
   1. Elastic IP Address
   2. **SQS**
   3. Amazon CloudWatch
   4. Amazon ElastiCache
2. A company has a workflow that sends video files from their on-premise system to AWS for transcoding. They use EC2 worker instances that pull transcoding jobs from SQS. Why is SQS an appropriate service for this scenario?
   1. SQS guarantees the order of the messages.
   2. SQS synchronously provides transcoding output.
   3. SQS checks the health of the worker instances.
   4. **SQS helps to facilitate horizontal scaling of encoding tasks**
3. Which statement best describes an Amazon SQS use case?
   1. Automate the process of sending an email notification to administrators when the CPU utilization reaches 70% on production servers (Amazon EC2 instances) (CloudWatch + SNS + SES)
   2. **Create a video transcoding website where multiple components need to communicate with each other, but can’t all process the same amount of work simultaneously**(SQS provides loose coupling)
   3. Coordinate work across distributed web services to process employee’s expense reports (SWF – Steps in order and might need manual steps)
   4. Distribute static web content to end users with low latency across multiple countries (CloudFront + S3)
4. Your application provides data transformation services. Files containing data to be transformed are first uploaded to Amazon S3 and then transformed by a fleet of spot EC2 instances. Files submitted by your premium customers must be transformed with the highest priority. How should you implement such a system?
   1. Use a DynamoDB table with an attribute defining the priority level. Transformation instances will scan the table for tasks, sorting the results by priority level.
   2. Use Route 53 latency based-routing to send high priority tasks to the closest transformation instances.
   3. **Use two SQS queues, one for high priority messages, and the other for default priority. Transformation instances first poll the high priority queue; if there is no message, they poll the default priority queue**
   4. Use a single SQS queue. Each message contains the priority level. Transformation instances poll high-priority messages first.
5. Your company plans to host a large donation website on Amazon Web Services (AWS). You anticipate a large and undetermined amount of traffic that will create many database writes. To be certain that you do not drop any writes to a database hosted on AWS. Which service should you use?
   1. Amazon RDS with provisioned IOPS up to the anticipated peak write throughput.
   2. **Amazon Simple Queue Service (SQS) for capturing the writes and draining the queue to write to the database**
   3. Amazon ElastiCache to store the writes until the writes are committed to the database.
   4. Amazon DynamoDB with provisioned write throughput up to the anticipated peak write throughput.
6. A customer has a 10 GB AWS Direct Connect connection to an AWS region where they have a web application hosted on Amazon Elastic Computer Cloud (EC2). The application has dependencies on an on-premises mainframe database that uses a BASE (Basic Available, Soft state, Eventual consistency) rather than an ACID (Atomicity, Consistency, Isolation, Durability) consistency model. The application is exhibiting undesirable behavior because the database is not able to handle the volume of writes. How can you reduce the load on your on-premises database resources in the most cost-effective way?
   1. Use an Amazon Elastic Map Reduce (EMR) S3DistCp as a synchronization mechanism between the onpremises database and a Hadoop cluster on AWS.
   2. **Modify the application to write to an Amazon SQS queue and develop a worker process to flush the queue to the on-premises database**
   3. Modify the application to use DynamoDB to feed an EMR cluster which uses a map function to write to the on-premises database.
   4. Provision an RDS read-replica database on AWS to handle the writes and synchronize the two databases using Data Pipeline.
7. An organization has created a Queue named “modularqueue” with SQS. The organization is not performing any operations such as SendMessage, ReceiveMessage, DeleteMessage, GetQueueAttributes, SetQueueAttributes, AddPermission, and RemovePermission on the queue. What can happen in this scenario?
   1. AWS SQS sends notification after 15 days for inactivity on queue
   2. **AWS SQS can delete queue after 30 days without notification**
   3. AWS SQS marks queue inactive after 30 days
   4. AWS SQS notifies the user after 2 weeks and deletes the queue after 3 weeks.
8. A user is using the AWS SQS to decouple the services. Which of the below mentioned operations is not supported by SQS?
   1. SendMessageBatch
   2. DeleteMessageBatch
   3. CreateQueue
   4. **DeleteMessageQueue**
9. A user has created a queue named “awsmodule” with SQS. One of the consumers of queue is down for 3 days and then becomes available. Will that component receive message from queue?
   1. **Yes, since SQS by default stores message for 4 days**
   2. No, since SQS by default stores message for 1 day only
   3. No, since SQS sends message to consumers who are available that time
   4. Yes, since SQS will not delete message until it is delivered to all consumers
10. A user has created a queue named “queue2” in US-East region with AWS SQS. The user’s AWS account ID is 123456789012. If the user wants to perform some action on this queue, which of the below Queue URL should he use?
    1. **http://sqs.us-east-1.amazonaws.com/123456789012/queue2**
    2. http://sqs.amazonaws.com/123456789012/queue2
    3. http://sqs. 123456789012.us-east-1.amazonaws.com/queue2
    4. http://123456789012.sqs.us-east-1.amazonaws.com/queue2
11. A user has created a queue named “myqueue” with SQS. There are four messages published to queue, which are not received by the consumer yet. If the user tries to delete the queue, what will happen?
    1. A user can never delete a queue manually. AWS deletes it after 30 days of inactivity on queue
    2. **It will delete the queue**
    3. It will initiate the delete but wait for four days before deleting until all messages are deleted automatically.
    4. I t will ask user to delete the messages first
12. A user has developed an application, which is required to send the data to a NoSQL database. The user wants to decouple the data sending such that the application keeps processing and sending data but does not wait for an acknowledgement of DB. Which of the below mentioned applications helps in this scenario?
    1. AWS Simple Notification Service
    2. AWS Simple Workflow
    3. **AWS Simple Queue Service**
    4. AWS Simple Query Service
13. You are building an online store on AWS that uses SQS to process your customer orders. Your backend system needs those messages in the same sequence the customer orders have been put in. How can you achieve that?
    1. It is not possible to do this with SQS
    2. **You can use sequencing information on each message**
    3. You can do this with SQS but you also need to use SWF
    4. Messages will arrive in the same order by default
14. A user has created a photo editing software and hosted it on EC2. The software accepts requests from the user about the photo format and resolution and sends a message to S3 to enhance the picture accordingly. Which of the below mentioned AWS services will help make a scalable software with the AWS infrastructure in this scenario?
    1. AWS Glacier
    2. AWS Elastic Transcoder
    3. AWS Simple Notification Service
    4. **AWS Simple Queue Service**
15. Refer to the architecture diagram of a batch processing solution using Simple Queue Service (SQS) to set up a message queue between EC2 instances, which are used as batch processors. Cloud Watch monitors the number of Job requests (queued messages) and an Auto Scaling group adds or deletes batch servers automatically based on parameters set in Cloud Watch alarms. You can use this architecture to implement which of the following features in a cost effective and efficient manner? 
    1. Reduce the overall time for executing jobs through parallel processing by allowing a busy EC2 instance that receives a message to pass it to the next instance in a daisy-chain setup.
    2. Implement fault tolerance against EC2 instance failure since messages would remain in SQS and worn can continue with recovery of EC2 instances implement fault tolerance against SQS failure by backing up messages to S3.
    3. Implement message passing between EC2 instances within a batch by exchanging messages through SOS.
    4. **Coordinate number of EC2 instances with number of job requests automatically thus Improving cost effectiveness**
    5. Handle high priority jobs before lower priority jobs by assigning a priority metadata field to SQS messages.
16. How does Amazon SQS allow multiple readers to access the same message queue without losing messages or processing them many times?
    1. By identifying a user by his unique id
    2. By using unique cryptography
    3. **Amazon SQS queue has a configurable visibility timeout**
    4. Multiple readers can’t access the same message queue
17. A user has created photo editing software and hosted it on EC2. The software accepts requests from the user about the photo format and resolution and sends a message to S3 to enhance the picture accordingly. Which of the below mentioned AWS services will help make a scalable software with the AWS infrastructure in this scenario?
    1. AWS Elastic Transcoder
    2. AWS Simple Notification Service
    3. **AWS Simple Queue Service**
    4. AWS Glacier
18. How do you configure SQS to support longer message retention?
    1. **Set the MessageRetentionPeriod attribute using the SetQueueAttributes method**
    2. Using a Lambda function
    3. You can’t. It is set to 14 days and cannot be changed
    4. You need to request it from AWS
19. A user has developed an application, which is required to send the data to a NoSQL database. The user wants to decouple the data sending such that the application keeps processing and sending data but does not wait for an acknowledgement of DB. Which of the below mentioned applications helps in this scenario?
    1. AWS Simple Notification Service
    2. AWS Simple Workflow
    3. AWS Simple Query Service
    4. **AWS Simple Queue Service**
20. If a message is retrieved from a queue in Amazon SQS, how long is the message inaccessible to other users by default?
    1. 0 seconds
    2. 1 hour
    3. 1 day
    4. forever
    5. **30 seconds**
21. Which of the following statements about SQS is true?
    1. Messages will be delivered exactly once and messages will be delivered in First in, First out order
    2. Messages will be delivered exactly once and message delivery order is indeterminate
    3. Messages will be delivered one or more times and messages will be delivered in First in, First out order
    4. **Messages will be delivered one or more times and message delivery order is indeterminate** (Before the introduction of FIFO queues)
22. How long can you keep your Amazon SQS messages in Amazon SQS queues?
    1. From 120 secs up to 4 weeks
    2. From 10 secs up to 7 days
    3. **From 60 secs up to 2 weeks**
    4. From 30 secs up to 1 week
23. When a Simple Queue Service message triggers a task that takes 5 minutes to complete, which process below will result in successful processing of the message and remove it from the queue while minimizing the chances of duplicate processing?
    1. **Retrieve the message with an increased visibility timeout, process the message, delete the message from the queue**
    2. Retrieve the message with an increased visibility timeout, delete the message from the queue, process the message
    3. Retrieve the message with increased DelaySeconds, process the message, delete the message from the queue
    4. Retrieve the message with increased DelaySeconds, delete the message from the queue, process the message
24. You need to process long-running jobs once and only once. How might you do this?
    1. Use an SNS queue and set the visibility timeout to long enough for jobs to process.
    2. Use an SQS queue and set the reprocessing timeout to long enough for jobs to process.
    3. **Use an SQS queue and set the visibility timeout to long enough for jobs to process.**
    4. Use an SNS queue and set the reprocessing timeout to long enough for jobs to process.
25. You are getting a lot of empty receive requests when using Amazon SQS. This is making a lot of unnecessary network load on your instances. What can you do to reduce this load?
    1. Subscribe your queue to an SNS topic instead.
    2. **Use as long of a poll as possible, instead of short polls.**(Refer [link](http://docs.aws.amazon.com/AWSSimpleQueueService/latest/SQSDeveloperGuide/sqs-long-polling.html))
    3. Alter your visibility timeout to be shorter.
    4. Use <code>sqsd</code> on your EC2 instances.
26. You have an asynchronous processing application using an Auto Scaling Group and an SQS Queue. The Auto Scaling Group scales according to the depth of the job queue. The completion velocity of the jobs has gone down, the Auto Scaling Group size has maxed out, but the inbound job velocity did not increase. What is a possible issue?
    1. **Some of the new jobs coming in are malformed and unprocessable.**(As other options would cause the job to stop processing completely, the only reasonable option seems that some of the recent messages must be malformed and unprocessable)
    2. The routing tables changed and none of the workers can process events anymore. (If changed, none of the jobs would be processed)
    3. Someone changed the IAM Role Policy on the instances in the worker group and broke permissions to access the queue. (If IAM role changed no jobs would be processed)
    4. The scaling metric is not functioning correctly. (scaling metric did work fine as the autoscaling caused the instances to increase)
27. Company B provides an online image recognition service and utilizes SQS to decouple system components for scalability. The SQS consumers poll the imaging queue as often as possible to keep end-to-end throughput as high as possible. However, Company B is realizing that polling in tight loops is burning CPU cycles and increasing costs with empty responses. How can Company B reduce the number of empty responses?
    1. Set the imaging queue visibility Timeout attribute to 20 seconds
    2. **Set the Imaging queue ReceiveMessageWaitTimeSeconds attribute to 20 seconds**(Long polling. Refer [link](http://docs.aws.amazon.com/AWSSimpleQueueService/latest/SQSDeveloperGuide/sqs-long-polling.html))
    3. Set the imaging queue MessageRetentionPeriod attribute to 20 seconds
    4. Set the DelaySeconds parameter of a message to 20 seconds