Amazon SQS FAQs

Overview

Q: What are the benefits of Amazon SQS over homegrown or packaged message queuing systems?

Amazon SQS provides several advantages over building your own software for managing message queues or using commercial or open-source message queuing systems that require significant up-front time for development and configuration.

These alternatives require ongoing hardware maintenance and system administration resources. The complexity of configuring and managing these systems is compounded by the need for redundant storage of messages that ensures messages are not lost if hardware fails.

In contrast, Amazon SQS requires no administrative overhead and little configuration. Amazon SQS works on a massive scale, processing billions of messages per day. You can scale the amount of traffic you send to Amazon SQS up or down without any configuration. Amazon SQS also provides extremely high message durability, giving you and your stakeholders added confidence.

Q: How is Amazon SQS different from Amazon SNS?

Amazon SNS allows applications to send time-critical messages to multiple subscribers through a “push” mechanism, eliminating the need to periodically check or “poll” for updates. Amazon 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.

Q: How is Amazon SQS different from Amazon MQ?

If you're using messaging with existing applications, and want to move your messaging to the cloud quickly and easily, we recommend you consider Amazon MQ. It supports industry-standard APIs and protocols so you can switch from any standards-based message broker to Amazon MQ without rewriting the messaging code in your applications. If you are building brand new applications in the cloud, we recommend you consider Amazon SQS and Amazon SNS. Amazon SQS and SNS are lightweight, fully managed message queue and topic services that scale almost infinitely and provide simple, easy-to-use APIs.

Q: Does Amazon SQS provide message ordering?

Yes. FIFO (first-in-first-out) queues preserve the exact order in which messages are sent and received. If you use a FIFO queue, you don't have to place sequencing information in your messages. For more information, see FIFO Queue Logic in the *Amazon SQS Developer Guide*.

Standard queues provide a loose-FIFO capability that attempts to preserve the order of messages. However, because standard queues are designed to be massively scalable using a highly distributed architecture, receiving messages in the exact order they are sent is not guaranteed.

Q: Does Amazon SQS guarantee delivery of messages?

Standard queues provide at-least-once delivery, which means that each message is delivered at least once.

FIFO queues provide exactly-once processing, which means that each message is delivered once and remains available until a consumer processes it and deletes it. Duplicates are not introduced into the queue.

Q: How is Amazon SQS different from Amazon Kinesis Streams?

Amazon SQS offers a reliable, highly-scalable hosted queue for storing messages as they travel between applications or microservices. It moves data between distributed application components and helps you decouple these components. Amazon SQS provides common middleware constructs such as dead-letter queues and poison-pill management. It also provides a generic web services API and can be accessed by any programming language that the AWS SDK supports. Amazon SQS supports both standard and FIFO queues.

Amazon Kinesis Streams allows real-time processing of streaming big data and the ability to read and replay records to multiple Amazon Kinesis Applications. The Amazon Kinesis Client Library (KCL) delivers all records for a given partition key to the same record processor, making it easier to build multiple applications that read from the same Amazon Kinesis stream (for example, to perform counting, aggregation, and filtering).

For more information, see the Amazon Kinesis Documentation.

Q: Does Amazon use Amazon SQS for its own applications?

Yes. Developers at Amazon use Amazon SQS for a variety of applications that process large numbers of messages every day. Key business processes in both Amazon.com and Amazon Web Services use Amazon SQS.

Billing

Q: How much does Amazon SQS cost?

You pay only for what you use, and there is no minimum fee.

The cost of Amazon SQS is calculated per request, plus data transfer charges for data transferred out of Amazon SQS (unless data is transferred to Amazon EC2 instances or to AWS Lambda functions within the same region). For detailed pricing breakdowns per queue type and region, see Amazon SQS Pricing.

Q: What can I do with the Amazon SQS Free Tier?

The Amazon SQS Free Tier provides you with 1 million requests per month at no charge.

Many small-scale applications are able to operate entirely within the limits of the Free Tier. However, data transfer charges might still apply. For more information, see Amazon SQS Pricing.

The Free Tier is a monthly offer. Free usage does not accumulate across months.

Q: Will I be charged for all Amazon SQS requests?

Yes, for any requests beyond the free tier. All Amazon SQS requests are chargeable, and they are billed at the same rate.

Q: Do Amazon SQS batch operations cost more than other requests?

No. Batch operations (SendMessageBatch, DeleteMessageBatch, and ChangeMessageVisibilityBatch) all cost the same as other Amazon SQS requests. By grouping messages into batches, you can reduce your Amazon SQS costs.

Q: How will I be charged and billed for my use of Amazon SQS?

There are no initial fees to begin using Amazon SQS. At the end of the month, your credit card will be automatically charged for the month's usage.

You can view your charges for the current billing period at any time on the AWS website:

1. Log into your AWS account.
2. Under **Your Web Services Account**, select **Account Activity**.

Q: How can I track and manage the costs associated with my Amazon SQS queues?

You can tag and track your queues for resource and cost management using cost allocation tags. A tag is a metadata label comprised of a key-value pair. For example, you can tag your queues by cost center and then categorize and track your costs based on these cost centers.

For more information, see Tagging Your Amazon SQS Queues in the Amazon SQS Developer Guide. For more information on cost allocation tagging of AWS resources, see Using Cost Allocation Tags in the AWS Billing and Cost Management User Guide.

Q: Do your prices include taxes?

Except as noted otherwise, our prices don't include any applicable taxes and duties such as VAT or applicable sales tax.

For customers with a Japanese billing address, the use of AWS in any region is subject to Japanese Consumption Tax. For more information, see the Amazon Web Services Consumption Tax FAQ.

Features, functionality, and interfaces

Q: Can I use Amazon SQS with other AWS services?

Yes. You can make your applications more flexible and scalable by using Amazon SQS with compute services such as Amazon EC2, Amazon EC2 Container Service (Amazon ECS), and AWS Lambda, as well as with storage and database services such as Amazon Simple Storage Service (Amazon S3) and Amazon DynamoDB.

Q: How can I interact with Amazon SQS?

You can access Amazon SQS using the AWS Management Console, which helps you create Amazon SQS queues and send messages easily.

Amazon SQS also provides a web services API. It is also integrated with the AWS SDKs, allowing you to work in the programming language of your choice.

Q: What API actions are available for Amazon SQS?

For information on message queue operations, see the Amazon SQS API Reference.

Q: Who can perform operations on a message queue?

Only an AWS account owner (or an AWS account that the account owner has delegated rights to) can perform operations on an Amazon SQS message queue.

Q: Can I use Java Message Service (JMS) with Amazon SQS?

Yes. You can take advantage of the scale, low cost, and high availability of Amazon SQS without the worry and high overhead of running your own JMS cluster.

Amazon provides the Amazon SQS Java Messaging Library that implements the JMS 1.1 specification and uses Amazon SQS as the JMS provider. For more information, see Using JMS with Amazon SQS in the *Amazon SQS Developer Guide*.

Q: How does Amazon SQS identify messages?

All messages have a global unique ID that Amazon SQS returns when the message is delivered to the message queue. The ID isn’t required to perform any further actions on the message, but it is useful for tracking the receipt of a particular message in the message queue.

When you receive a message from the message queue, the response includes a receipt handle that you must provide when deleting the message.

For more information, see Queue and Message Identifiers in the Amazon SQS Developer Guide.

Q: How does Amazon SQS handle messages that can't be processed?

In Amazon SQS, you can use the API or the console to configure dead letter queues, which are queues that receive messages from other source queues.

If you make a queue into a dead letter queue, it receives messages after a maximum number of processing attempts cannot be completed. You can use dead letter queues to isolate messages that can't be processed for later analysis.

For more information, see "Can I use a dead letter queue with FIFO queues?" on this page and Using Amazon SQS

Dead Letter Queues in the Amazon SQS Developer Guide.

Q: What is a visibility timeout?

The visibility timeout is a period of time during which Amazon SQS prevents other consuming components from receiving and processing a message. For more information, see Visibility Timeout in the *Amazon SQS Developer Guide*.

Q: Does Amazon SQS support message metadata?

Yes. An Amazon SQS message can contain up to 10 metadata attributes. You can use message attributes to separate the body of a message from the metadata that describes it. This helps process and store information with greater speed and efficiency because your applications don't have to inspect an entire message before understanding how to process it.

Amazon SQS message attributes take the form of name-type-value triples. The supported types include string, binary, and number (including integer, floating-point, and double). For more information, see Using Amazon SQS Message Attributes in the *Amazon SQS Developer Guide*.

Q: How can I determine the time-in-queue value?

To determine the time-in-queue value, you can request the SentTimestamp attribute when receiving a message. Subtracting that value from the current time results in the time-in-queue value.

Q: What is the typical latency for Amazon SQS?

Typical latencies for SendMessage, ReceiveMessage, and DeleteMessage API requests are in the tens or low hundreds of milliseconds.

Q: For anonymous access, what is the value of the SenderId attribute for a message?

When the AWS account ID is not available (for example, when an anonymous user sends a message), Amazon SQS provides the IP address.

Q: What is Amazon SQS long polling?

Amazon SQS long polling is a way to retrieve messages from your Amazon SQS queues. While the regular short polling returns immediately, even if the message queue being polled is empty, long polling doesn't return a response until a message arrives in the message queue, or the long poll times out.

Long polling makes it inexpensive to retrieve messages from your Amazon SQS queue as soon as the messages are available. Using long polling might reduce the cost of using SQS, because you can reduce the number of empty receives. For more information, see Amazon SQS Long Polling in the *Amazon SQS Developer Guide*.

Q: Is there an additional charge for using Amazon SQS long polling?

No. Long-polling ReceiveMessage calls are billed exactly the same as short-polling ReceiveMessage calls.

Q: When should I use Amazon SQS long polling, and when should I use Amazon SQS short polling?

In almost all cases, Amazon SQS long polling is preferable to short polling. Long-polling requests let your queue consumers receive messages as soon as they arrive in your queue while reducing the number of empty ReceiveMessageResponse instances returned.

Amazon SQS long polling results in higher performance at reduced cost in the majority of use cases. However, if your application expects an immediate response from a ReceiveMessage call, you might not be able to take advantage of long polling without some modifications to your application.

For example, if your application uses a single thread to poll multiple queues, switching from short polling to long polling will probably not work, because the single thread will wait for the long-poll timeout on any empty queues, delaying the processing of any queues that might contain messages.

In such an application, it is a good practice to use a single thread to process only one queue, allowing the application to take advantage of the benefits that Amazon SQS long polling provides.

Q: What value should I use for my long-poll timeout?

In general, you should use maximum 20 seconds for a long-poll timeout. Because higher long-poll timeout values reduce the number of empty ReceiveMessageResponse instances returned, try to set your long-poll timeout as high as possible.

If the 20-second maximum doesn't work for your application (see the example in the previous question), set a shorter long-poll timeout, as low as 1 second.

All AWS SDKs work with 20-second long polls by default. If you don't use an AWS SDK to access Amazon SQS, or if you configured your AWS SDK to specifically have a shorter timeout, you might need to modify your Amazon SQS client to allow longer requests or to use a shorter long-poll timeout.

Q: What is the AmazonSQSBufferedAsyncClient for Java?

The AmazonSQSBufferedAsyncClient 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 your application to immediately process messages from Amazon SQS without waiting for the messages to be retrieved

Working together, automatic batching and prefetching increase the throughput and reduce the latency of your application while reducing your costs by making fewer Amazon SQS requests. For more information, see Client-Side Buffering and Request Batching in the *Amazon SQS Developer Guide*.

Q: Where can I download the AmazonSQSBufferedAsyncClient for Java?

You can download the AmazonSQSBufferedAsyncClient as part of the AWS SDK for Java.

Q: Do I have to rewrite my application to use the AmazonSQSBufferedAsyncClient for Java?

No. The AmazonSQSBufferedAsyncClient for Java is implemented as a drop-in replacement for the existing AmazonSQSAsyncClient.

If you update your application to use the latest AWS SDK and change your client to use the AmazonSQSBufferedAsyncClient for Java instead of the AmazonSQSAsyncClient, your application will receive the added benefits of automatic batching and prefetching.

Q: How can I subscribe Amazon SQS message queues to receive notifications from Amazon SNS topics?

1. In the Amazon SQS console, select an Amazon SQS standard queue.
2. Under Queue Actions, select Subscribe Queue to SNS Topic from the drop-down list.
3. In the dialog box, select the topic from the Choose a Topic drop-down list, and click Subscribe.

For more information, see Subscribing a Queue to an Amazon SNS Topic in the *Amazon SQS Developer Guide*.

Q: Can I delete all messages in a message queue without deleting the message queue itself?

Yes. You can delete all messages in an Amazon SQS message queue using the PurgeQueue action.

When you purge a message queue, all the messages previously sent to the message queue are deleted. Because your message queue and its attributes remain, there is no need to reconfigure the message queue; you can continue using it.

To delete only specific messages, use the DeleteMessage or DeleteMessageBatch actions.

For more information, see this Tutorial: Purging Messages from an Amazon SQS Queue.

FIFO queues

Q: What regions are FIFO queues available in?

FIFO queues are currently available in the following regions:

- US West (Oregon, N. California)
- US East (Ohio, N. Virginia)
- GovCloud (US-East)
- GovCloud (US-West)
- EU (Ireland, Frankfurt, London, Paris, Stockholm)
- Asia Pacific (Sydney, Tokyo, Mumbai, Seoul, Singapore)
- Canada (Central)
- South America (Sao Paulo)
- China (Ningxia), operated by NWCD
- China (Beijing), operated by SINNET
- Asia Pacific (Hong Kong)

Q: How many copies of a message will I receive?

FIFO queues are designed to never introduce duplicate messages. However, your message producer might introduce duplicates in certain scenarios: for example, if the producer sends a message, does not receive a response, and then resends the same message. Amazon SQS APIs provide deduplication functionality that prevents your message producer from sending duplicates. Any duplicates introduced by the message producer are removed within a 5-minute deduplication interval.

For standard queues, you might occasionally receive a duplicate copy of a message (at-least-once delivery). If you use a standard queue, you must design your applications to be idempotent (that is, they must not be affected adversely when processing the same message more than once).

For more information, see Exactly-Once Processing in the Amazon SQS Developer Guide.

Q: Are the Amazon SQS queues I used previously changing to FIFO queues?

No. Amazon SQS *standard* queues (the new name for existing queues) remain unchanged, and you can still create standard queues. These queues continue to provide the highest scalability and throughput; however, you will not get ordering guarantees and duplicates might occur.

Standard queues are appropriate for many scenarios, such as work distribution with multiple idempotent consumers.

Q: Can I convert my existing standard queue to a FIFO queue?

No. You must choose the queue type when you create it. However, it is possible to move to a FIFO queue. For more information, see Moving From a Standard Queue to a FIFO Queue in the *Amazon SQS Developer Guide*.

Q: Are Amazon SQS FIFO queues backwards-compatible?

To take advantage of FIFO queue functionality, you must use the latest AWS SDK.

FIFO queues use the same API actions as standard queues, and the mechanics for receiving and deleting messages and changing the visibility timeout are the same. However, when sending messages, you must specify a message group ID. For more information, see FIFO Queue Logic in the *Amazon SQS Developer Guide*.

Q: With which AWS or external services are Amazon SQS FIFO queues compatible?

Some AWS or external services that send notifications to Amazon SQS might not be compatible with FIFO queues, despite allowing you to set a FIFO queue as a target.

The following features of AWS services aren't currently compatible with FIFO queues:

- Auto Scaling Lifecycle Hooks

- AWS IoT Rule Actions
- AWS Lambda Dead Letter Queues

For information about compatibility of other services with FIFO queues, see your service documentation.

Q: Are Amazon SQS FIFO queues compatible with the Amazon SQS Buffered Asynchronous Client, the Amazon SQS Extended Client Library for Java, or the Amazon SQS Java Message Service (JMS) Client?

FIFO queues aren't currently compatible with the Amazon SQS Buffered Asynchronous Client.

FIFO queues are compatible with the Amazon SQS Extended Client Library for Java and the Amazon SQS Java Message Service (JMS) client.

Q: Which AWS CloudWatch metrics do Amazon SQS FIFO queues support?

FIFO queues support all metrics that standard queues support. For FIFO queues, all approximate metrics return accurate counts. For example, the following AWS CloudWatch metrics are supported:

- ApproximateNumberOfMessagesDelayed - The number of messages in the queue that are delayed and not available for reading immediately.
- ApproximateNumberOfMessagesVisible - The number of messages available for retrieval from the queue.
- ApproximateNumberOfMessagesNotVisible - The number of messages that are in flight (sent to a client but have not yet been deleted or have not yet reached the end of their visibility window).

Q: What are message groups?

Messages are grouped into distinct, ordered "bundles" within a FIFO queue. For each message group ID, all messages are sent and received in strict order. However, messages with different message group ID values might be sent and received out of order. You must associate a message group ID with a message. If you don't provide a message group ID, the action fails.

If multiple hosts (or different threads on the same host) send messages with the same message group ID are sent to a FIFO queue, Amazon SQS delivers the messages in the order in which they arrive for processing. To ensure that Amazon SQS preserves the order in which messages are sent and received, ensure that multiple senders send each message with a unique message group ID.

For more information, see FIFO Queue Logic in the *Amazon SQS Developer Guide*.

Q: Do Amazon SQS FIFO queues support multiple producers?

Yes. One or more producers can send messages to a FIFO queue. Messages are stored in the order that they were successfully received by Amazon SQS.

If multiple producers send messages in parallel, without waiting for the success response from SendMessage or SendMessageBatch actions, the order between producers might not be preserved. The response of SendMessage or SendMessageBatch actions contains the final ordering sequence that FIFO queues use to place messages in the queue, so your multiple-parallel-producer code can determine the final order of messages in the queue.

Q: Do Amazon SQS FIFO queues support multiple consumers?

By design, Amazon SQS FIFO queues don't serve messages from the same message group to more than one consumer at a time. However, if your FIFO queue has multiple message groups, you can take advantage of parallel consumers, allowing Amazon SQS to serve messages from different message groups to different consumers.

Q: Can I use a dead letter queue with FIFO queues?

Yes. However, you must use a FIFO dead letter queue with a FIFO queue. (Similarly, you can use only a standard dead letter queue with a standard queue.)

Q: What is the throughput limit for an Amazon SQS FIFO queue?

By default, FIFO queues support up to 3,000 messages per second with batching, or up to 300 messages per second

(300 send, receive, or delete operations per second) without batching. If you require a higher throughput, submit a support ticket to request a review of your FIFO queue requirements.

Q: Are there any limits specific to FIFO queue attributes?

The name of a FIFO queue must end with the .fifo suffix. The suffix counts towards the 80-character queue name limit. To determine whether a queue is FIFO, you can check whether the queue name ends with the suffix.

Security and reliability

Q: How reliable is the storage of my data in Amazon SQS?

Amazon SQS stores all message queues and messages within a single, highly-available AWS region with multiple redundant Availability Zones (AZs), so that no single computer, network, or AZ failure can make messages inaccessible. For more information, see Regions and Availability Zones in the *Amazon Relational Database Service User Guide*.

Q: How can I secure the messages in my message queues?

Authentication mechanisms ensure that messages stored in Amazon SQS message queues are secured against unauthorized access. You can control who can send messages to a message queue and who can receive messages from a message queue. For additional security, you can build your application to encrypt messages before they are placed in a message queue.

Amazon SQS has its own resource-based permissions system that uses policies written in the same language as AWS Identity and Access Management (IAM) policies: for example, you can use variables, just like in IAM policies. For more information, see Amazon SQS Policy Examples in the *Amazon SQS Developer Guid*e.

Amazon SQS supports the HTTP over SSL (HTTPS) and Transport Layer Security (TLS) protocols. Most clients can automatically negotiate to use newer versions of TLS without any code or configuration change. Amazon SQS supports versions 1.0, 1.1, and 1.2 of the Transport Layer Security (TLS) protocol in all regions.

Q: Why are there separate ReceiveMessage and DeleteMessage operations?

When Amazon SQS returns a message to you, the message stays in the message queue whether or not you actually receive the message. You're responsible for deleting the message and the deletion request acknowledges that you’re done processing the message.

If you don’t delete the message, Amazon SQS will deliver it again on when it receives another receive request. For more information, see Visibility Timeout in the *Amazon SQS Developer Guide*.

Q: Can a deleted message be received again?

No. FIFO queues never introduce duplicate messages.

For standard queues, under rare circumstances, you might receive a previously-deleted message a second time.

Q: What happens if I issue a DeleteMessage request on a previously-deleted message?

When you issue a DeleteMessage request on a previously-deleted message, Amazon SQS returns a *success* response.

Server-Side encryption (SSE)

Q: What are the benefits of server-side encryption (SSE) for Amazon SQS?

Server-side encryption (SSE) lets you transmit sensitive data in encrypted queues. SSE protects the contents of messages in Amazon SQS queues using keys managed in the AWS Key Management Service (AWS KMS). SSE encrypts messages as soon as Amazon SQS receives them. The messages are stored in encrypted form and Amazon SQS decrypts messages only when they are sent to an authorized consumer.

For more information, see Protecting Data Using Server-Side Encryption (SSE) and AWS KMS in the Amazon SQS Developer Guide

Q: Can I use SNS, Cloud Watch Events and S3 Events with encrypted queues?

Yes. To do this you need to enable compatibility between AWS services (eg. Amazon CloudWatch Events, Amazon S3, and Amazon SNS), and Queues with SSE. For detailed instructions see the Compatibility section of the SQS Developer Guide.

Q: What regions are queues with SSE available in?

Server-side encryption (SSE) for Amazon SQS is available in the following regions: Asia Pacific (Mumbai, Osaka, Seoul, Singapore, Sydney, Tokyo, Ningxia, and Beijing), Canada (Central), EU (Frankfurt, Ireland, London, and Paris), South America (Sao Paulo), US West (N. California, Oregon), US East (N. Virginia, Ohio) and GovCloud (US).

Q: How do I enable SSE for a new or existing Amazon SQS queue?

To enable SSE for a new or existing queue using the Amazon SQS API, specify the customer master key (CMK) ID: the alias, alias ARN, key ID, or key ARN of the an AWS-managed CMK or a custom CMK by setting the KmsMasterKeyId attribute of the CreateQueue or SetQueueAttributes action.

For detailed instructions, see Creating an Amazon SQS Queue with Server-Side Encryption and Configuring Server-Side Encryption (SSE) for an Existing Amazon SQS Queue in the *Amazon SQS Developer Guide*.

Q: What Amazon SQS queue types can use SSE?

Both standard and FIFO queues support SSE.

Q: What permissions do I need to use SSE with Amazon SQS?

Before you can use SSE, you must configure AWS KMS key policies to allow encryption of queues and encryption and decryption of messages.

To enable SSE for a queue, you can use the AWS-managed customer master key (CMK) for Amazon SQS or a custom CMK. For more information, see Customer Master Keys in the *AWS KMS Developer Guide*.

To send messages to an encrypted queue, the producer must have the kms:GenerateDataKey and kms:Decrypt permissions for the CMK.

To receive messages from an encrypted queue, the consumer must have the kms:Decrypt permission for any CMK that is used to encrypt the messages in the specified queue. If the queue acts as a dead letter queue, the consumer must also have the kms:Decrypt permission for any CMK that is used to encrypt the messages in the source queue.

For more information, see What Permissions Do I Need to Use SSE? in the *Amazon SQS Developer Guide*.

Q: Are there any charges for using SSE with Amazon SQS?

There are no additional Amazon SQS charges. However, there are charges for calls from Amazon SQS to AWS KMS. For more information, see AWS Key Management Service Pricing.

The charges for using AWS KMS depend on the data key reuse period configured for your queues. For more information, see How Do I Estimate My AWS KMS Usage Costs? in the *Amazon SQS Developer Guide*.

Q: What does SSE for Amazon SQS encrypt and how is it encrypted?

SSE encrypts the body of a message in an Amazon SQS queue.

SSE doesn't encrypt the following components:

- Queue metadata (queue name and attributes)
- Message metadata (message ID, timestamp, and attributes)
- Per-queue metrics

Amazon SQS generates data keys based on the AWS-managed customer master key (CMK) for Amazon SQS or a custom CMK to provide envelope encryption and decryption of messages for a configurable time period (from 1 minute to 24 hours).

For more information, see What Does SSE for Amazon SQS Encrypt? in the *Amazon SQS Developer Guide.*

Q: What algorithm does SSE for Amazon SQS use to encrypt messages?

SSE uses the AES-GCM 256 algorithm.

Q: Does SSE limit the transactions per second (TPS) or number of queues that can be created with Amazon SQS?

SSE doesn't limit the throughput (TPS) of Amazon SQS. The number of SSE queues that you can create is limited by the following:

- The data key reuse period (1 minute to 24 hours).
- The AWS KMS per-account limit (100 TPS by default).
- The number of IAM users or accounts that access queues.
- The existence of a large backlog (a larger backlog requires more AWS KMS calls).

For example, let's assume the following limits:

- You set your data key reuse period to 5 minutes (300 seconds).
- Your KMS account has a default AWS KMS TPS limit of 100 TPS.
- You use an Amazon SQS queue without a backlog and with 1 IAM user for
- SendMessage or ReceiveMessage actions to all queues.

In this case, you can calculate the theoretical maximum of Amazon SQS queues with SSE as follows:

> **300 seconds × 100 TPS / 1 IAM user = 30,000 queues**

Q: How can I estimate my AWS KMS usage costs?

To predict costs and better understand your AWS bill, you might want to know how often Amazon SQS uses your CMK.

> **Note**: Although the following formula can give you a very good idea of expected costs, actual costs might be higher because of the distributed nature of Amazon SQS.

To calculate the number of API requests per queue (R), use the following formula:

> **R = B / D * (2 * P + C)**

B is the billing period (in seconds)

D is the data key reuse period (in seconds)

P is the number of producing principals that send to the Amazon SQS queue.

C is the number of consuming principals that receive from the Amazon SQS queue.

> **Important**: In general, producing principals incur double the cost of consuming principals. For more information, see How Does the Data Key Reuse Period Work? in the *Amazon SQS Developer Guide*.
>
> If the producer and consumer have different IAM users, the cost increases.

For more information, see How Do I Estimate My AWS KMS Usage Costs? in the *Amazon SQS Developer Guide*

Compliance

Q: Is Amazon SQS PCI DSS certified?

Yes. Amazon SQS is PCI DSS Level 1 certified. For more information, see PCI Compliance.

Q: Is Amazon SQS HIPAA-eligible?

Yes, AWS has expanded its HIPAA compliance program to include Amazon SQS as a HIPAA Eligible Service. If you have an executed Business Associate Agreement (BAA) with AWS, you can use Amazon SQS to build your HIPAA-compliant applications, store messages in transit, and transmit messages—including messages containing protected health information (PHI).

If you already have an executed BAA with AWS, you can start using Amazon SQS right away. If you don't have a BAA or have other questions about using AWS for your HIPAA-compliant applications, contact us for more information.

Note: If you prefer not to transfer PHI through Amazon SQS (or if you have messages larger than 256 KB), you can alternatively send Amazon SQS message payloads through Amazon S3 using the Amazon SQS Extended Client Library for Java (Amazon S3 is a HIPAA Eligible Service, excluding the use of Amazon S3 Transfer Acceleration). For more information, see Using the Amazon SQS Extended Client Library for Java in the *Amazon SQS Developer Guide*.

Limits and restrictions

Q: How long can I keep my messages in Amazon SQS message queues?

Longer message retention provides greater flexibility to allow for longer intervals between message production and consumption.

You can configure the Amazon SQS message retention period to a value from 1 minute to 14 days. The default is 4 days. Once the message retention limit is reached, your messages are automatically deleted.

Q: How do I configure Amazon SQS to support longer message retention?

To configure the message retention period, set the MessageRetentionPeriod attribute using the console or using the Distributiveness method. Use this attribute to specify the number of seconds a message will be retained in Amazon SQS.

You can use the MessageRetentionPeriod attribute to set the message retention period from 60 seconds (1 minute) to 1,209,600 seconds (14 days). For more information on working with this message attribute, see the *Amazon SQS API Reference*.

Q: How do I configure the maximum message size for Amazon SQS?

To configure the maximum message size, use the console or the SetQueueAttributes method to set the MaximumMessageSize attribute. This attribute specifies the limit on bytes that an Amazon SQS message can contain. Set this limit to a value between 1,024 bytes (1 KB), and 262,144 bytes (256 KB). For more information, see Using Amazon SQS Message Attributes in the *Amazon SQS Developer Guide*.

To send messages larger than 256 KB, use the Amazon SQS Extended Client Library for Java. This library lets you send an Amazon SQS message that contains a reference to a message payload in Amazon S3 that can be as large as 2 GB.

Q: What kind of data can I include in a message?

Amazon SQS messages can contain up to 256 KB of text data, including XML, JSON and unformatted text. The following Unicode characters are accepted:

#x9 | #xA | #xD | [#x20 to #xD7FF] | [#xE000 to #xFFFD] | [#x10000 to #x10FFFF]

For more information, see the XML 1.0 Specification.

Q: How large can Amazon SQS message queues be?

A single Amazon SQS message queue can contain an unlimited number of messages. However, there is a 120,000 limit for the number of inflight messages for a standard queue and 20,000 for a FIFO queue. Messages are inflight after they have been received from the queue by a consuming component, but have not yet been deleted from the queue.

Q: How many message queues can I create?

You can create any number of message queues.

Q: Is there a size limit on the name of Amazon SQS message queues?

Queue names are limited to 80 characters.

Q: Are there restrictions on the names of Amazon SQS message queues?

You can use alphanumeric characters, hyphens (-), and underscores (_).

Q: Can I reuse a message queue name?

A message queue's name must be unique within an AWS account and region. You can reuse a message queue's name after you delete the message queue.

Queue sharing

Q: How do I share a message queue?

You can associate an access policy statement (and specify the permissions granted) with the message queue to be shared. Amazon SQS provides APIs for creating and managing access policy statements:

- AddPermission
- RemovePermission
- SetQueueAttributes
- GetQueueAttributes

For more information, see the *Amazon SQS API Reference*.

Q: Who pays for shared queue access?

The message queue owner pays for shared message queue access.

Q: How do I identify another AWS user I want to share a message queue with?

The Amazon SQS API uses the AWS account number to identify AWS users.

Q: What do I need to provide to an AWS user I want to share a message queue with?

To share a message queue with an AWS user, provide the full URL from the message queue you want to share. The CreateQueue and ListQueues operations return this URL in their responses.

Q: Does Amazon SQS support anonymous access?

Yes. You can configure an access policy that allows anonymous users to access a message queue.

Q: When should I use the permissions API?

The permissions API provides an interface for sharing access to a message queue to developers. However, this API cannot allow conditional access or more advanced use cases.

Q: When should I use the SetQueueAttributes operation with JSON objects?

The SetQueueAttributes operation supports the full access policy language. For example, you can use the policy language to restrict access to a message queue by IP address and time of day. For more information, see Amazon SQS Policy Examples in the *Amazon SQS Developer Guide*.

Service access and regions

Q: What regions is Amazon SQS available in?

For service region availability, see the AWS Global Infrastructure Region Table.

Q: Can I share messages between queues in different regions?

No. Each Amazon SQS message queue is independent within each region.

Q: Is there a pricing difference between regions?

Amazon SQS pricing is the same for all regions, except China (Beijing). For more information, see Amazon SQS Pricing.

Q: What is the pricing structure between various regions?

You can transfer data between Amazon SQS and Amazon EC2 or AWS Lambda free of charge within a single region.

When you transfer data between Amazon SQS and Amazon EC2 or AWS Lambda in different regions, you will be charged the normal data transfer rate. For more information, see Amazon SQS Pricing.