



AWS SNS Simple Notification Service

Amazon Simple Notification Service (SNS) is a fully managed messaging service provided by AWS that enables you to decouple and scale microservices, distributed systems, and serverless applications. Here are some key reasons why you might need AWS SNS:

Decoupling Microservices: SNS allows different parts of an application to communicate without needing to be directly connected. This decoupling improves the modularity and scalability of the application.

Broadcasting Messages: SNS supports the fan-out pattern where a single message can be sent to multiple subscribers, such as AWS Lambda functions, HTTP/S endpoints, or other AWS services. This is useful for sending notifications or updates to many subscribers at once.

Event-Driven Architectures: SNS can be used to trigger actions in response to events. For example, change in data can trigger an SNS notification, which in turn triggers AWS Lambda functions or other services.

Cross-Account Notifications: SNS can send notifications across different AWS accounts, enabling secure and efficient communication between different teams or departments.

Mobile Push Notifications: SNS supports sending push notifications to mobile devices, making it easy to send updates and messages to mobile app users.

Scalability and Reliability: SNS is designed to handle high-throughput, making it suitable for applications that need to send a large number of messages. It also ensures reliable delivery of messages with built-in retry mechanisms.

Simplified Message Delivery: SNS abstracts the complexities of message delivery, allowing you to focus on the logic of your application rather than the underlying infrastructure.


Integration with Other AWS Services: SNS seamlessly integrates with various AWS services such as AWS Lambda, Amazon SQS, and Amazon S3, providing a comprehensive solution for building event-driven applications.

Security: SNS offers several features to secure message delivery, including encryption, access policies, and Amazon VPC support for private communication.

Cost-Effective: SNS offers a pay-as-you-go pricing model, making it a cost-effective solution for applications with variable or unpredictable messaging needs.



AWS SNS Simple Notification Service

 SNS has many twin features with SQS, starting with FIFO & Standard types, so functionality is replica of SQS.

In Simple Terms

- Used to broadcast messages
- Pub-Sub model (publisher publishes messages to a topic, subscribers listen to the topic)
- Instant message delivery (does not queue messages)

Encryption

- In-flight encryption by default using HTTPS API
- At-rest encryption using KMS keys (optional)
- Client-side encryption

Access Management

- **IAM policies** to regulate access to the SNS API
- **SNS Access Policies** (resource-based policy)
 - Used for cross-account access to SNS topics
 - Used for allowing other AWS services to publish to an SNS topic

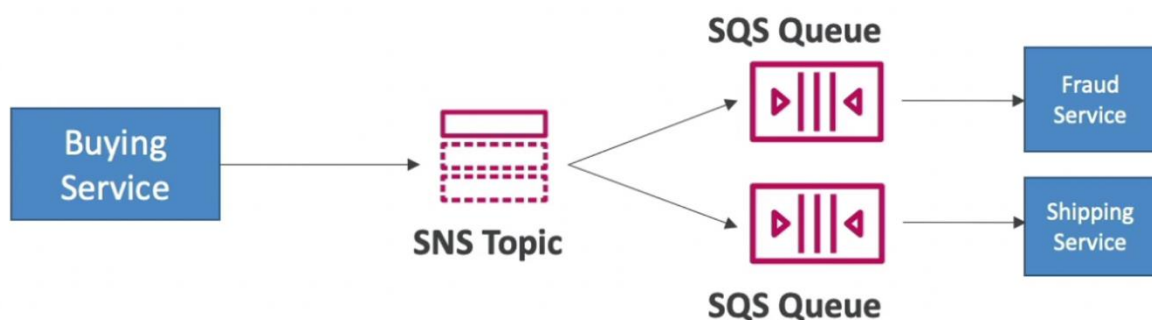


AWS SNS Simple Notification Service

As we mentioned SNS has twin features as SQS – keeping that in mind

- Guaranteed ordering of messages in that topic
- Publishing messages to a FIFO topic requires:
 - Group ID:** messages will be ordered and grouped for each group ID
 - Message deduplication ID:** for deduplication of messages
- **Can only have SQS FIFO queues as subscribers**
- **Limited throughput (same as SQS FIFO)** because only SQS FIFO queues can read from FIFO topics
- **The topic name must end with. fifo**

Sample diagram





AWS SNS Simple Notification Service

Keeping the above diagram as base -- > There are two components for the AWS SNS

- One is the Producers – the services who are sending the messages to AWS SNS
- Second is the Consumers (also called as subscribers) – the services which can receive the messages from the AWS SNS.

Who can be the Producers – Any service via SDKs / Customizations

To keep you updated – these are the few ones, we use probably – in our day-to-day life

- **AWS Console:** You can manually publish messages to an SNS topic through the AWS Management Console.
- **AWS SDKs:** Applications written in various programming languages (e.g., Python, Java, JavaScript, etc.) can use the AWS SDKs to publish messages to SNS.
- **AWS CLI:** You can use the AWS Command Line Interface (CLI) to publish messages to an SNS topic.
- **AWS Lambda:** Lambda functions can be configured to publish messages to SNS topics. This is commonly used in event-driven architectures.
- **Amazon CloudWatch Alarms:** CloudWatch Alarms can send notifications to SNS topics based on specific monitoring metrics or thresholds.
- **Amazon S3:** S3 can be configured to send event notifications to SNS topics when certain actions occur, such as object creation or deletion.
- **Amazon EC2 Auto Scaling:** Auto Scaling groups can send notifications to SNS topics when scaling events occur (e.g., instance launch or termination).
- **Amazon CloudFormation:** CloudFormation stacks can send notifications to SNS topics about stack events, such as creation, updates, and deletions.
- **AWS CloudTrail:** CloudTrail can be configured to send log events to SNS topics, providing notifications about API activity in your AWS account.
- **Amazon RDS:** Amazon RDS can send notifications to SNS topics about database events such as backups, snapshots, and instance status changes.
- **Third-Party Services:** Any third-party service or application that can make HTTP/HTTPS requests can publish messages to SNS by calling the SNS API endpoints.
- **Custom Applications:** Any custom application or service that has network access to the AWS SNS API can publish messages to an SNS topic.



AWS SNS Simple Notification Service

Who can be the Subscribers – AWS provided (limited certain) services

09 Subscriber Types that – AWS SNS supports

- **AWS Lambda:** You can invoke AWS Lambda functions with SNS messages. This is useful for processing the messages or triggering other AWS services in a serverless architecture.
- **HTTP & HTTPS:** SNS can send messages to endpoints via HTTP or HTTPS. This allows integration with web services and applications that can handle HTTP requests.
- **Amazon SQS:** Messages can be delivered to Amazon Simple Queue Service (SQS) queues. This is useful for decoupling and buffering messages between different parts of a system.
- **Email & Email-JSON:** SNS can send messages as plain text emails or JSON-formatted emails. This is useful for sending notifications and alerts to email addresses.
- **SMS (Short Message Service):** SNS can send text messages to mobile phones, making it useful for sending alerts and notifications directly to users.
- **Application End point:** SNS supports sending push notifications to mobile devices through various push notification services, including:
 - Apple Push Notification Service (APNs) for iOS devices
 - Firebase Cloud Messaging (FCM) for Android devices
 - Amazon Device Messaging (ADM) for Amazon devices
 - Baidu Cloud Push for Android devices in China
- **Kinesis Data Firehose:** KDF - Kinesis Data Firehose can ingest and process large volumes of streaming data in real time. SNS can push notifications and events that trigger real-time processing pipelines.



AWS SNS Simple Notification Service

The SNS Fan Out Pattern

The Fan-Out Pattern in AWS SNS refers to a design where a single message published to an SNS topic is replicated and delivered to multiple subscribers. This pattern is particularly useful for distributing the same data or event notification to multiple services or systems simultaneously. Here's a more detailed explanation:

How the Fan-Out Pattern Works

SNS Topic Creation:

You create an SNS topic that acts as a central hub for message distribution.

Subscriptions:

Various endpoints subscribe to the SNS topic. These endpoints can be AWS Lambda functions, Amazon SQS queues, HTTP/HTTPS endpoints, email addresses, or mobile push notifications.

Message Publishing:

When a message is published to the SNS topic, SNS automatically replicates and delivers the message to all subscribed endpoints.

Message Filtering

- JSON policy used to filter messages sent to SNS topic's subscriptions
- Each subscriber will have its own filter policy (if a subscriber doesn't have a filter policy, it receives every message)
- Ex: filter messages sent to each queue by the order status



AWS SNS Simple Notification Service

AWS SNS – Numbers Play

- Maximum Message Size: **256 KB**
- Maximum Number of Topics per AWS Account: **100,000**
- Maximum Number of Subscriptions per Topic: **No explicit limit**
- Message Retention Period: **4 days (96 hours):**
- Maximum Number of Message Attributes: **10**
- Delivery Retry Attempts for HTTP/HTTPS Endpoints: **5**
- Minimum Retry Delay for HTTP/HTTPS Endpoints: **20 seconds**
- Maximum Delivery Delay: **900 seconds (15 minutes)**
- Maximum Length of a Subject Line for Email Notifications: **100 characters**
- Default SMS Rate Limit for US Destinations: **100 messages per second**
- Maximum Number of Tags per SNS Topic: **50**
- Maximum Length of an SNS Topic Name: **256 characters**
- Maximum Number of Characters in SNS Message Attribute Names: **256 characters**
- Maximum Number of FIFO Topics per AWS Account: **100,000**
- Maximum Number of SMS Messages in a Single Message: **160 characters**
- Number of Supported Regions: **24: As of July 2024**
- Number of Retries for Messages to SQS: **100,000**
- Maximum Number of Characters in SNS Message Body for Mobile Push: **4,096 characters**
- Number of Subscription Protocols Supported: **09**
- Maximum Number of Delivery Attempts to HTTP/HTTPS Endpoints: **15**



AWS SNS Simple Notification Service

Numeric based Questions

<https://lisireddy.medium.com/aws-sns-numerical-questions-b2de02fe3d44>

Scenario based Questions

<https://lisireddy.medium.com/aws-sns-scenario-based-questions-370a8e4f9021>

Wish you the best ...! Happy Learning ...!

Yours' Love (@lisireddy across all the platforms)