JIDNYASA

CORPORATE TRAINNING CENTRE

STUDY NOTES

AWS/DevOps

AWS-SIMPLE NOTIFICATION SERVICE (SNS)

What is Amazon Simple Notification Service (SNS) ?

- Amazon Simple Notification Service (SNS) is a **fully managed messaging service** provided by Amazon Web Services (AWS) that allows you to send notifications or messages to a large number of subscribers or systems.
- Let's understand this in simple words, SNS is a tool that helps **send notifications to people or systems** at once. Imagine you have something important to share, like an update or alert or alarm, and you want to make sure it reaches everyone quickly. With SNS, you can send that message, and it will automatically be delivered to all the people or systems that need to see it.
- You can also integrate **SNS with Amazon CloudWatch to send alarms and alerts generated for aws resources**.
- When you monitor your AWS resources using CloudWatch, you often want to be notified if something goes wrong or if a specific event occurs. This is where SNS can help us to notify people about.
- With SNS you can send messages through different methods, like text messages, emails, or even directly to other apps.
- You can send message/notification to millions of people. It's a reliable and efficient way to keep everyone informed.
- SNS provides this service to both application-to-person and application-to-application.

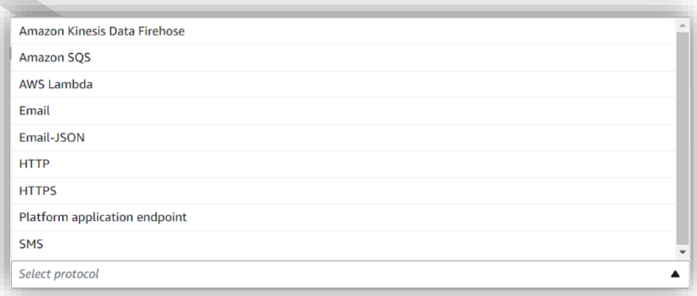
Features of SNS:

1. Pub/Sub Messaging Model:

- Publisher/Subscriber: SNS follows a publish/subscribe (pub/sub) messaging model, where publishers send messages to topics, and subscribers receive those messages. Topics act as a communication channel.
- Fan-out Messaging: With SNS, a single message can be sent to multiple subscribers, including AWS Lambda functions, Amazon SQS queues, HTTP/HTTPS endpoints, email, and mobile devices.

2. Multiple Protocol Support:

- SNS supports several protocols for message delivery, including:
    - HTTP/HTTPS
    - Email/Email-JSON
    - SMS (text messages)
    - Amazon SQS
    - AWS Lambda



3. Scalability and High Throughput:

- SNS is designed to handle millions of messages per second, making it highly scalable for applications that require high-throughput messaging.

4. Serverless and Fully Managed:

- As a fully managed service, SNS handles the infrastructure, scaling, and maintenance, allowing developers to focus on building their applications rather than managing messaging infrastructure.

5. Message Filtering:

- SNS allows message filtering, enabling subscribers to receive only the messages that match specific criteria, reducing the need for custom filtering logic on the subscriber side.

6. Durability and Reliability:

- SNS ensures message delivery with built-in retries and dead-letter queues (DLQs) for messages that fail to be delivered.

7. Security:

- SNS integrates with AWS Identity and Access Management (IAM) to control access and permissions for sending and receiving messages.
- Messages can be encrypted at rest and in transit using AWS Key Management Service (KMS).

8. Integration with Other AWS Services:

- SNS is deeply **integrated with other AWS services, such as AWS Lambda, S3, Cloud Watch, SQS**, and more, enabling powerful automation and event-driven architectures.

9. Use Cases:

- **Application Alerts**: Send alerts and notifications to administrators or users.
- **Fan-out Messaging**: Distribute messages to multiple endpoints, like pushing updates to mobile apps.
- **Decoupling Microservices**: Allow different microservices to communicate asynchronously.

10. Cost-Effectiveness:

- SNS is pay-as-you-go, where you are charged based on the number of messages published and delivered, making it cost-effective for varying workloads.

SNS with S3:

Amazon Simple Notification Service (SNS) can also be used to send alarms and alerts related to Amazon S3 (Simple Storage Service) events. This is useful for monitoring and automating actions in response to specific events that occur in your S3 buckets, such as file uploads, deletions, or access attempts. Here's how SNS can be used with S3 for alarms and alerts.

1. Setting Up S3 Event Notifications:

- **Event Triggers**: In S3, you can configure event notifications for specific actions, such as:
  - **Object Created**: When a new object (file) is uploaded to the bucket.
  - **Object Removed**: When an object is deleted from the bucket.
  - **Object Accessed**: When an object is accessed (using access logs).
- **SNS as a Notification Destination**: You can specify SNS as the destination for these event notifications. When an event occurs, S3 will automatically send a message to the associated SNS topic.

2. Subscribing to the SNS Topic:

- **Subscribers Receive Alerts**: Once an S3 event triggers a notification, the message is sent to the SNS topic. Subscribers to this topic (e.g., administrators, security teams, or automated systems) will receive the alert through their preferred method, such as:
  - **Email**: Notify team members of important S3 events via email.
  - **SMS**: Send critical alerts via text messages.
  - **HTTP/HTTPS Endpoints**: Forward the notification to a web service for automated processing.
  - **AWS Lambda**: Invoke a Lambda function for further processing, like moving the file, triggering a workflow, or logging the event.

3. Example Use Cases:

- **Data Ingestion Pipeline**: When a new file is uploaded to an S3 bucket, an SNS notification can trigger a Lambda function that processes the file or moves it to another storage location.
- **Security Alerts**: Set up alerts for specific events, such as unauthorized access attempts, or when a file is deleted from a sensitive bucket.
- **Operational Monitoring**: Receive alerts when certain actions occur in S3, like when log files are rotated or new backups are created.

4. Benefits:

- **Immediate Notification**: SNS provides real-time alerts when S3 events occur, allowing for quick response and action.
- **Scalable and Reliable**: SNS ensures that notifications are delivered to all subscribers, even when handling a large volume of S3 events.
- **Integration with Other AWS Services**: SNS can trigger actions in other AWS services, enabling complex event-driven architectures and automated workflows.

5. Setting It Up:

- **Step 1**: In the S3 bucket, navigate to **Properties** > **Event notifications** and create a new event notification.
- **Step 2**: Specify the event type (e.g., object created, deleted) and choose SNS as the destination.
- **Step 3**: Select or create an SNS topic where the notifications will be sent.
- **Step 4**: Add subscribers to the SNS topic who need to receive the alerts.

Using SNS with S3 event notifications is an effective way to stay informed about important changes and activities in your S3 buckets, allowing you to automate responses and maintain better oversight of your storage environment.

SNS with Cloud Watch :

1. Setting Up Alarms in CloudWatch:

Create an Alarm: In CloudWatch, you can create an alarm that watches a specific metric (like CPU usage, disk space, or error rates). You define the conditions under which the alarm should trigger, such as when CPU usage exceeds 80% for more than 5 minutes.

2. Linking the Alarm to SNS:

Choose SNS as the Action: When setting up the alarm, you can specify SNS as the action to take when the alarm state changes (e.g., from "OK" to "ALARM").
SNS Topic: You select or create an SNS topic, which acts as the channel for sending the notifications. Subscribers to this topic will receive the alert.

3. Notifying Subscribers:

Automatic Alerts: When the alarm is triggered, CloudWatch sends a message to the SNS topic. This message is then automatically distributed to all subscribers of that topic.
Multiple Notification Channels: SNS can send these alerts via different methods, such as:

- **Email:** Send an email to notify administrators or support teams.
- **SMS**: Send a text message to key personnel.
- **HTTP/HTTPS**: Trigger an HTTP endpoint for automated responses or integrations.
- **AWS Lambda**: Invoke a Lambda function for custom automated responses.

4. Example Use Cases:

System Monitoring: Receive alerts when a server's CPU usage is too high, indicating a potential issue that needs immediate attention.
Cost Monitoring: Get notified if your AWS spending exceeds a certain threshold.
Application Errors: Automatically alert the development team if an application is experiencing high error rates.

5. Benefits:

Real-Time Notifications: Get instant alerts when something goes wrong, allowing for quick response times.
Multi-Channel Alerts: Ensure that the right people are notified via the method that works best for them.
Scalability: Handle a large number of alarms and alerts, sending notifications to many subscribers without any manual intervention.

SNS Pricing:

Amazon Simple Notification Service (SNS) pricing is based on a pay-as-you-go model, where you are charged based on the number of messages published, delivered, and any additional features you use. Here's a breakdown of the pricing components:

1. Free Tier

- **First 1 million requests**: Free every month.
- **First 1,000 SMS messages**: Free every month for SMS in the United States.

2. Pricing for Requests

- **Publish Requests**: $0.50 per 1 million requests.
- **Delivery to HTTP/HTTPS Endpoints**: $0.60 per 1 million notifications.
- **Delivery to Email/Email-JSON**: $2.00 per 1 million notifications.

3. Pricing for SMS Messages

- **SMS Delivery Pricing**: The cost varies based on the destination country. Here are some examples:
    - **United States**: $0.0075 per SMS.
    - **India**: $0.0027 per SMS.
    - **United Kingdom**: $0.0338 per SMS.
- **Additional Fees**: Depending on the carrier and the country, additional fees may apply.

4. Pricing for Mobile Push Notifications

- **Delivery to Mobile Devices (Push Notifications)**: $0.50 per 1 million mobile push notifications delivered.

5. Delivery to Other AWS Services

- **Amazon SQS**: No charge for delivering messages to Amazon SQS.
- **AWS Lambda**: No charge for delivering messages to AWS Lambda.

6. Data Transfer Costs

- **Intra-Region Data Transfer**: Free for data transferred between SNS and other AWS services in the same region.
- **Inter-Region Data Transfer**: Standard AWS data transfer rates apply when messages are delivered across regions.

7. Additional Charges

- **Message Filtering**: No additional charges for using message filtering with SNS topics.
- **Subscriptions**: You're not charged for setting up subscriptions, but delivery charges apply as mentioned above.

8. Example Costs

- If you publish 5 million messages in a month, your cost would be:
    - **Publishing Requests**: $0.50 per 1 million requests = $2.50.
    - **HTTP Delivery**: If 5 million messages are delivered to an HTTP endpoint, the cost would be $3.00.
    - **SMS Delivery**: If 10,000 SMS are sent to the U.S., the cost would be $75.00 (10,000 x $0.0075).

9. Cost Optimization

- **Use the Free Tier**: For small applications or initial testing, you can leverage the free tier.
- **Filter Messages**: Only deliver messages to relevant subscribers to reduce costs.
- **Monitor and Budget**: Use AWS Budgets to monitor and control SNS spending.

10. Conclusion

SNS pricing is straightforward and can be very cost-effective, especially if you stay within the free tier limits. However, costs can increase with high-volume messaging, particularly with SMS and email notifications. It's essential to monitor your usage and optimize where possible.

PRACTICAL

Let's publish a sample message using Amazon Simple Notification Service (SNS):

Part 1: Create Topic:
Go to the AWS Management Console → In the AWS Console, search for "SNS" in the search bar and click on "Simple Notification Service" to open the SNS Dashboard → In the SNS dashboard, click on "Topics" from the left-hand menu → Click the "Create topic" button → Choose a topic type: **Standard** (most common) or **FIFO** (for first-in, first-out delivery order) → Enter a name for the topic (e.g., "MyTopic") and configure additional settings like Display Name (for SMS notifications) → Click "Create topic"

Part2: Create Subscription:
After creating your topic go on "Create subscription" option from left hand sides menu → Click on "Create subscription" → For Topic ARN select ARN of Topic we have created in Part1 → Choose the protocol as "Email" → For the endpoint enter your email id example@gmail.com on which you want to send message → Click on "Create Subscription" *(You can add multiple emails in subscribers)

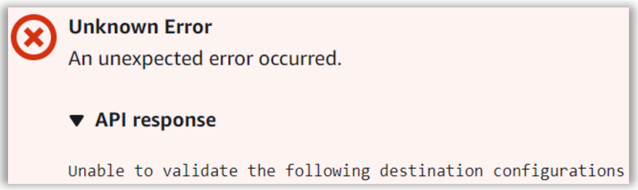
Part3: **Confirm the Subscription:**
Now open your email id and confirm the subscription by clicking the link sent to the email.

Part4: **Test/Publish the message:**
In the SNS console, click on "Topics" in the left-hand menu and select the topic you have just created to publish → Click on the "Publish message" button in the topic details page → Enter a subject for the message → Enter the text of the message you want to send in Message Body → You can choose to send a simple message or a JSON message. JSON allows you to customize messages for each protocol → Click "Publish message" to send the message to all the subscribers of the topic.

Set Up S3 Event Notification with SNS:

➔ Go to S3 console → Click on "Buckets" in the left-hand menu → Select the bucket for which you want to set up notifications → Go to the "Properties" tab of the selected bucket → Scroll down to the "Event notifications" section → Click "Create event notification." → Enter a name for the event (eg: TestEvent) → Under "Event types" choose the types of events you want to trigger the notification (For eg: All object create events) → Now scroll down and in Destination section click on "SNS topic" → Click on "SNS topic" drop down and select your SNS topic name that we have created earlier → Click on "Save changes" to apply the changes.

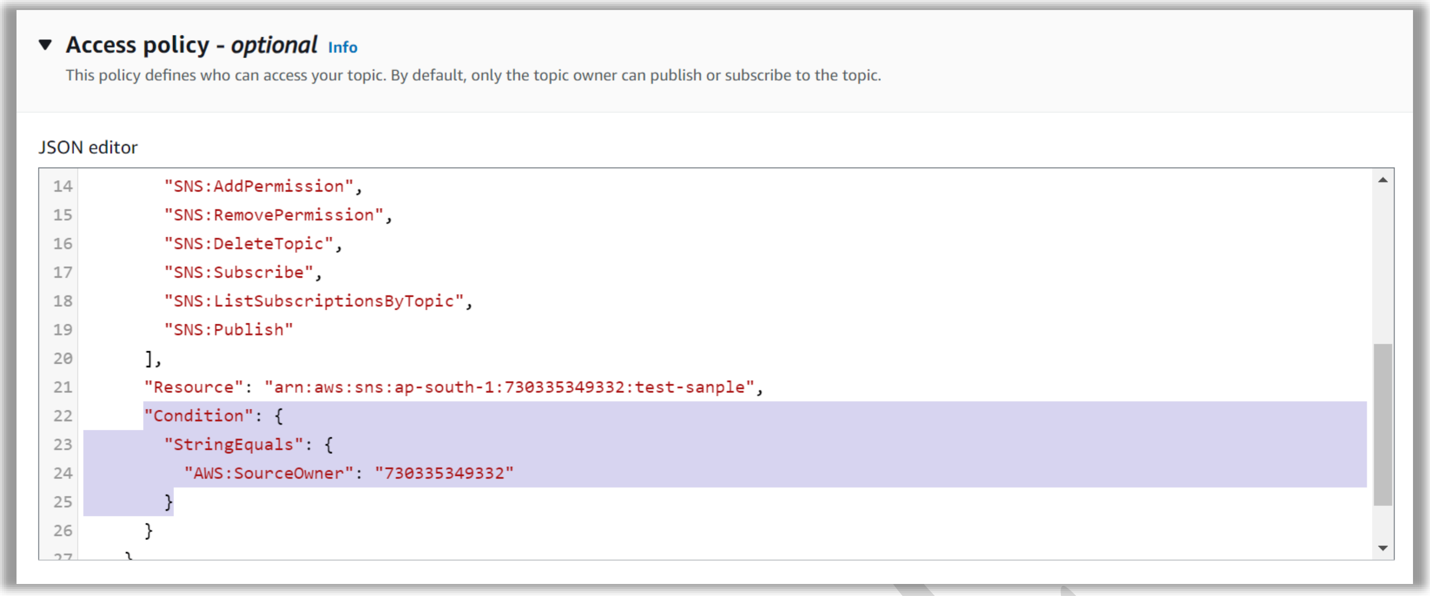
`Note: If you get below error while trying to create s3 event notification`


Unknown Error
An unexpected error occurred.

▼ **API response**

```
Unable to validate the following destination configurations
```

Then go to SNS Topic, and click on “Edit” → Go to ‘Access Policy’ and remove the condition from Policy as show below, remove highlighted part from policy:


▼ Access policy - *optional* Info
This policy defines who can access your topic. By default, only the topic owner can publish or subscribe to the topic.
JSON editor

```
        "SNS:AddPermission",
        "SNS:RemovePermission",
        "SNS:DeleteTopic",
        "SNS:Subscribe",
        "SNS:ListSubscriptionsByTopic",
        "SNS:Publish"
      ],
      "Resource": "arn:aws:sns:ap-south-1:730335349332:test-sanple",
      "Condition": {
        "StringEquals": {
          "AWS:SourceOwner": "730335349332"
        }
      }
```

Now try to create S3 event notification configuration:

Test:

Upload and object in s3 bucket and you can see you are receiving notification on email.