Notification Services: SNS

Dr Peadar Grant

November 14, 2022

Required reading

1. https://docs.aws.amazon.com/sns/latest/dg/welcome.html
2. https://docs.aws.amazon.com/general/latest/gr/aws-arns-and-namespaces.html

1 Use cases

- Sending end-user e-mails, push notifications and SMS messages to individual or groups of users.
- Notifying or triggering actions when **events** occur in parts of a system.

1.1 Asynchronous processes

Synchronous process is where the requestor makes a request, waits for it to complete, receives the results (if any) before continuing.

Asynchronous process is where the requestor makes a request and continues on working whilst its request is being processed. Results are returned either by polling or callback.

2 Simple notification service (SNS)

SNS is an **asynchronous** notification service. It is useful for both system-to-system and system-to-user communications. shows the key components of SNS.

The key unit of SNS is the **topic** to which two types of client connect:

Publisher wishes to send message to a number of subscribers via the topic. Any number of publishers per topic (within limits).

Subscribers wish to be notified of messages sent to the topic. SNS **pushes** messages to subscribers. Subscriptions to a single topic can be in a number of different forms. Any number of subscribers per topic (within limits).

2.1 Properties

There are some important properties that distinguish SNS from queueing systems (like SQS):

1. Messages always attempted to be delivered to all topic subscribers (fan-out,).
2. Topic subscribers are inhomogeneous (not all the same) and may process received message in different ways.
3. No persistence. Message received by whatever subscribers are present when message arrives. No “catch-up”.

2.2 Topic creation

```
# create topic
aws sns create-topic --name my-topic

# or, better:
$TopicArn=(aws sns create-topic --name my-topic | ConvertFrom-Json).TopicArn
```

2.3 Naming

Amazon Resource Names (ARNs) are the canonical naming format for SNS topics. Format:

```
arn:aws:sns:eu-west-1:637146340431:my-topic
|   |   |   |         |            |
1   2   3   4         5            6
```

Components:

1. All ARNs begin with `arn`.
2. **Partition** always `aws` unless `aws-cn` (China) or `aws-us-gov` GovCloud.
3. **Service** here `sns`
4. **Region** as per your default
5. **Account ID** number
6. **Topic name** as entered

If you have the required information you can construct the ARN for any SNS topic.

2.4 Listing topics

```
# list of topic ARNs
aws sns list-topics

# list of topic ARNs as PowerShell list
$Topics=(aws sns list-topics | ConvertFrom-Json).Topics
```

2.5 Deleting topics

```
# assume $TopicArn holds the topic ARN

aws sns delete-topic --topic-arn $TopicArn
```

2.6 SNS clients

SNS clients come in the form of publishers and subscribers, .

We will first meet subscribers and then publishers.

3 Subscribers

Subscribers to a topic can be managed by the `subscribe` and `unsubscribe` commands. Different types of subscribers possible to a topic, :

Machine-to-human: email, sms

Machine-to-machine (generic): email-json, http, https

Other AWS services: application, sqs, lambda

3.1 Subscribing

Subscribing requires a topic ARN, protocol and usually an endpoint.

```
# full info on command options
aws sns subscribe help

# assume $TopicArn holds ARN for topic

# subscribe to e-mail
aws sns subscribe `
--topic-arn $TopicArn `
--protocol email ` # also try email-json to see difference
--notification-endpoint "mail@mydomain.ie"
--return-subscription-arn
# may be a confirmation step required

# see help for other formats
```

3.2 Subscription ARN

Each subscription itself has an ARN, based on the topic ARN. Example:

```
arn:aws:sns:eu-west-1:637116340434:my-topic:74b543bc-0eab-46ea-81c1-0a654a6fb236
1   2   3   4         5            6        7
```

The first 6 components are identical to the topic ARN. The last component (7) identifies the subscription.

3.3 Checking if confirmed

```
aws sns get-subscription-attributes  --subscription-arn $SubscriptionArn
```

Check output of `get-subscription-attributes` for `PendingConfirmation`.

3.4 Unsubscribing

Unsubscribing requires the subscription ARN only:

```
aws sns unsubscribe --subscription-arn $SubscriptionArn
```

4 Publishing

Messages are *published* to a specific topic and are delivered to subscribers.

```
# send message to topic
aws sns publish --message "hello there" --topic-arn $TopicArn
```

SNSsns For the following exercise, attempt to do as much as possible using the command-line interface. Use the inbuilt help to discover the required commands. Document the required commands in your notes for each step.

1. Create an SNS topic named `lab-sns-test`.
2. Add two e-mail subscribers (e.g. your student and personal email).
    - Notice how SNS deals with subscription confirmation! Confirm *one* of the subscriptions only for now.
3. Send a test message to the topic.
    - Is it received by all endpoints?
    - Confirm the other subscription.
    - Does the message now send to the second subscription? Does your finding agree with the documentation?
    - Now try a new test message (with different content) and not what happens.