Amazon Web Services (AWS) Integration

Published: April 2014

Applies To

API Gateway v7.3.0

Feedback

Send suggestions and comments about this document to support@axway.com

Contents

Overview

AWS credentials

The API Gateway uses APIs exposed by Amazon Web Services (AWS). In order to access these APIs on AWS, both an access key id and a private (secret) key are required. This is the process for obtaining one:-

- You need to create an account at http://aws.amazon.com.
- Once you have created your account, you sign in at https://console.aws.amazon.com/ec2/home . Ignore the message to sign up to EC2, you don't need it.
- Click on Account->Security Credentials
- Look down the page to 'Access Credentials'. Your access key id is displayed in the table under 'access key id'. Your private (secret) key can be displayed by clicking the 'show' link next to it.

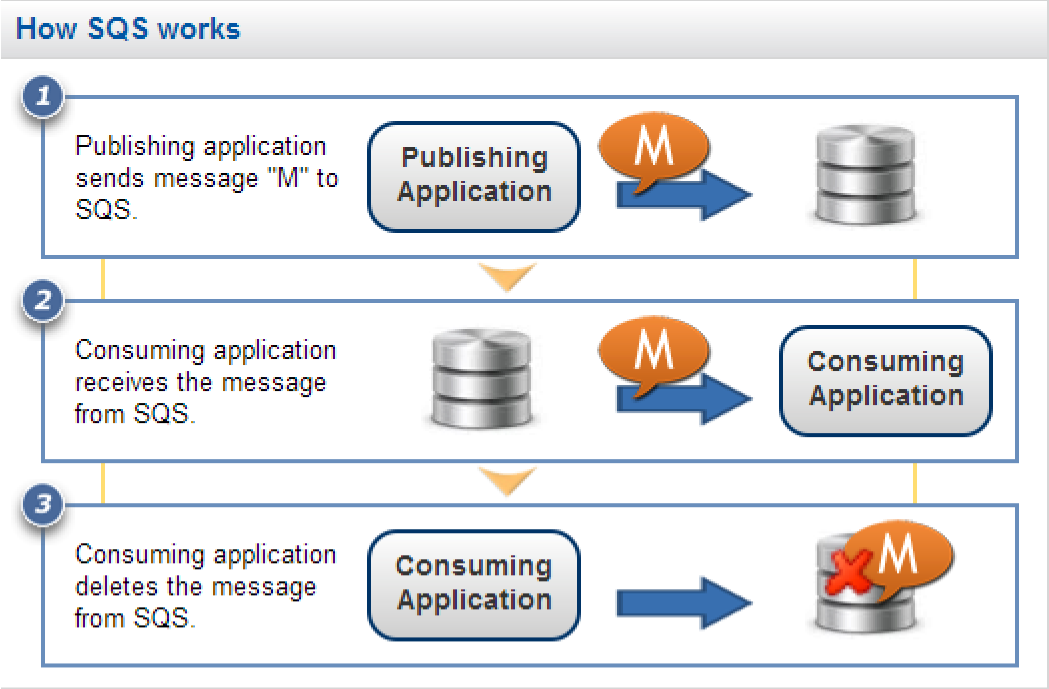
AWS Regions

AWS is located in 8 geographical "regions": US East (northern Virginia), U.S. West (northern California), U.S. West (Oregon), Brazil (São Paulo), Europe (Ireland), South Asia (Singapore), East Asia (Tokyo), and Australia (Sydney). There is also a "GovCloud" in the United States provided for U.S. government customers. Each Region is wholly contained within a single country and all of its data and services stay within the designated Region

Note that not all services (SQS, S3, SNS etc.) are available in all the regions. To find out if a service is available in a particular regions see http://aws.amazon.com/about-aws/globalinfrastructure/regional-product-services/. In most situations it is best to deploy your application as close to the end user as possible. For example, if most of your users are from the UK it would be best to use the EU (Ireland) region. Other things to keep in mind are legal requirements and cost (not all the regions are prices the same).

Amazon SQS

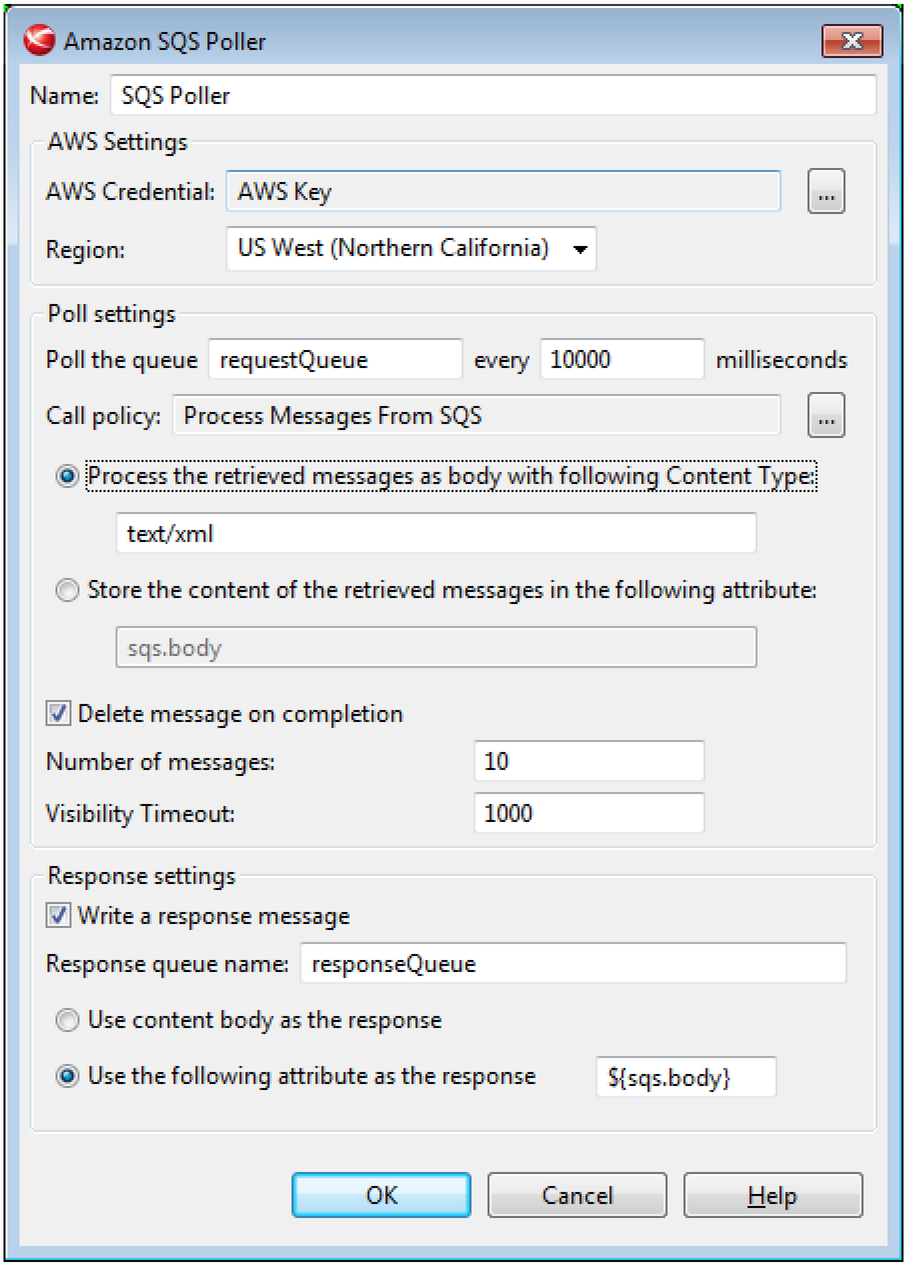
Amazon SQS (Simple Queue Service) is a hosted message queueing service for distributing messages amongst machines. The API Gateway acts as a client to SQS, as such it can poll SQS queues for messages to be processed and it can send messages to SQS.


How SQS works
1
Publishing application sends message "M" to SQS.
Publishing Application
M
2
Consuming application receives the message from SQS.
M
Consuming Application
3
Consuming application deletes the message from SQS.
Consuming Application
M

How SQS works - API Gateway can publish and consume to SQS

SQS Poller

The API Gateway can be configured to poll an SQS queue at a set rate, any message message found on the queue in this interval will can be sent to a policy for processing. To configure an SQS Poller follow the steps below.



Amazon SQS Poller

Name: SQS Poller

AWS Settings

AWS Credential: AWS Key

Region: US West (Northern California)

Poll settings

Poll the queue requestQueue every 10000 milliseconds

Call policy: Process Messages From SQS

● Process the retrieved messages as body with following Content Type:

text/xml

○ Store the content of the retrieved messages in the following attribute:

sqs.body

☑ Delete message on completion

Number of messages: 10

Visibility Timeout: 1000

Response settings

☑ Write a response message

Response queue name: responseQueue

○ Use content body as the response

● Use the following attribute as the response ${sqs.body}

Configuring an SQS poller

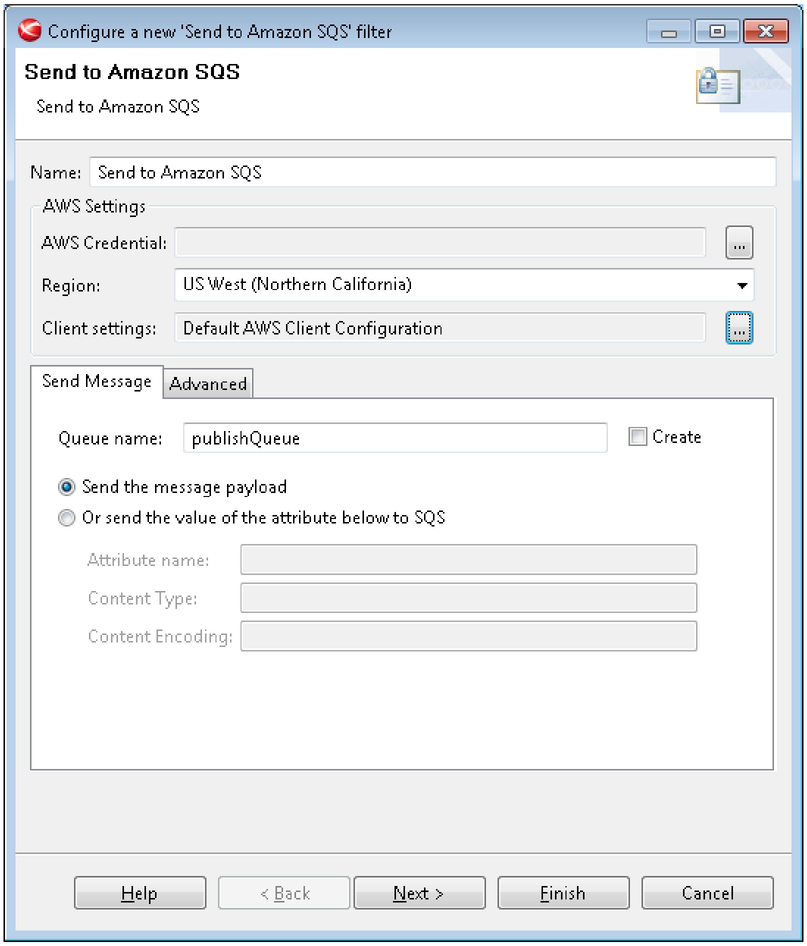
1. Click and expand the **Listeners** node in Policy Studio.
2. Right-click on **API Gateway**, and select **Amazon Web Services** and click **Add SQS Queue Listener**.
3. Select the **AWS Credentials** (see above) to be used by the API Gateway when connecting to SQS.
4. Select the **region** (see above) appropriate for your deployment.
5. Enter the name of the **queue** in SQS which will be polled
6. Enter the **poll rate** that the named queue
7. Once a messages has been consumed from the queue it is passed to the selected "**call policy"** for processing.
8. How the content of the SQS message is handled by the policy is determined by the either creating a

new **body** for processing the message or by storing the content the **named attribute**.

9. If you wish to delete the message from the queue when processing is coming the check the **Delete message on completion** button
10. Enter the **visibility timeout**, this is the duration (in seconds) that the received messages are hidden from subsequent retrieve requests after being retrieved by the API Gateway
11. Enter the maximum **number of messages** to return. Amazon SQS never returns more messages than this value but may return fewer.
12. If you wish to write data to a response queue then select the option to **write a response message.** Enter the **response queue name** and select where the contents of the response message are to be read from.

SQS Filter

The SQS Filter can send message to an SQS queue.



Configure a new 'Send to Amazon SQS' filter

Send to Amazon SQS

Send to Amazon SQS

Name: Send to Amazon SQS

AWS Settings

AWS Credential:

Region: US West (Northern California)

Client settings: Default AWS Client Configuration

Queue name: publishQueue

☐ Create

● Send the message payload

○ Or send the value of the attribute below to SQS

Attribute name:

Content Type:

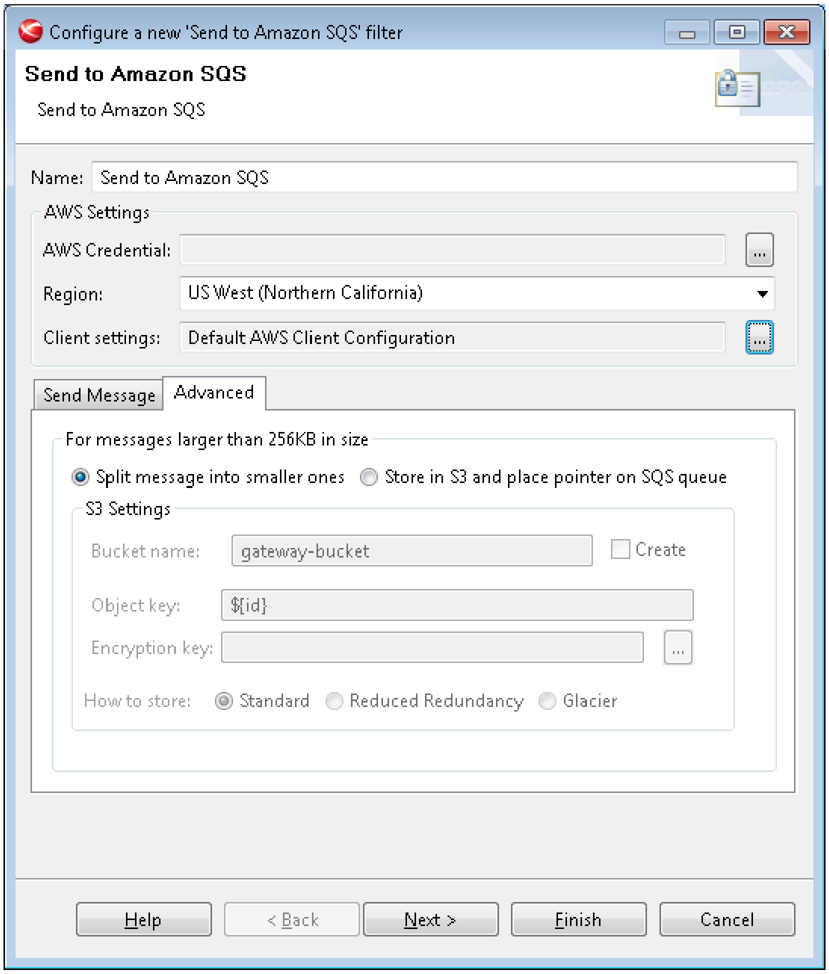
Content Encoding:

Configuring an SQS filter

When configuring the filter:

1. Select the **AWS Credentials** (see above) to be used by the API Gateway when connecting to SNS.
2. Select the **region** (see above) appropriate for your deployment.
3. Select the **client settings** (See appendix below) for the API Gateway.

4. Enter the **queue name** which the API Gateway will send content to
5. By default the API gateway will send the **message payload** to the named SQS queue, alternatively you can configure the filter to write the content of a message attribute from the whiteboard to the queue. If writing the content of a message attribute then you can supply the **content type** and the **content encoding** for the payload.
6. If the payload to be send to the queue is large than 256KB then it is possible to either **split the message into smaller one** or **store the payload in S3 and place point to S3 in the queue.**
7. If you wish to send large payloads to S3 configure the S3 settings (see S3 filter below)



Configure a new 'Send to Amazon SQS' filter

Send to Amazon SQS

Send to Amazon SQS

Name: Send to Amazon SQS

AWS Settings

AWS Credential:

Region: US West (Northern California)

Client settings: Default AWS Client Configuration

Send Message | Advanced

For messages larger than 256KB in size

● Split message into smaller ones ○ Store in S3 and place pointer on SQS queue

S3 Settings

Bucket name: gateway-bucket □ Create

Object key: ${id}

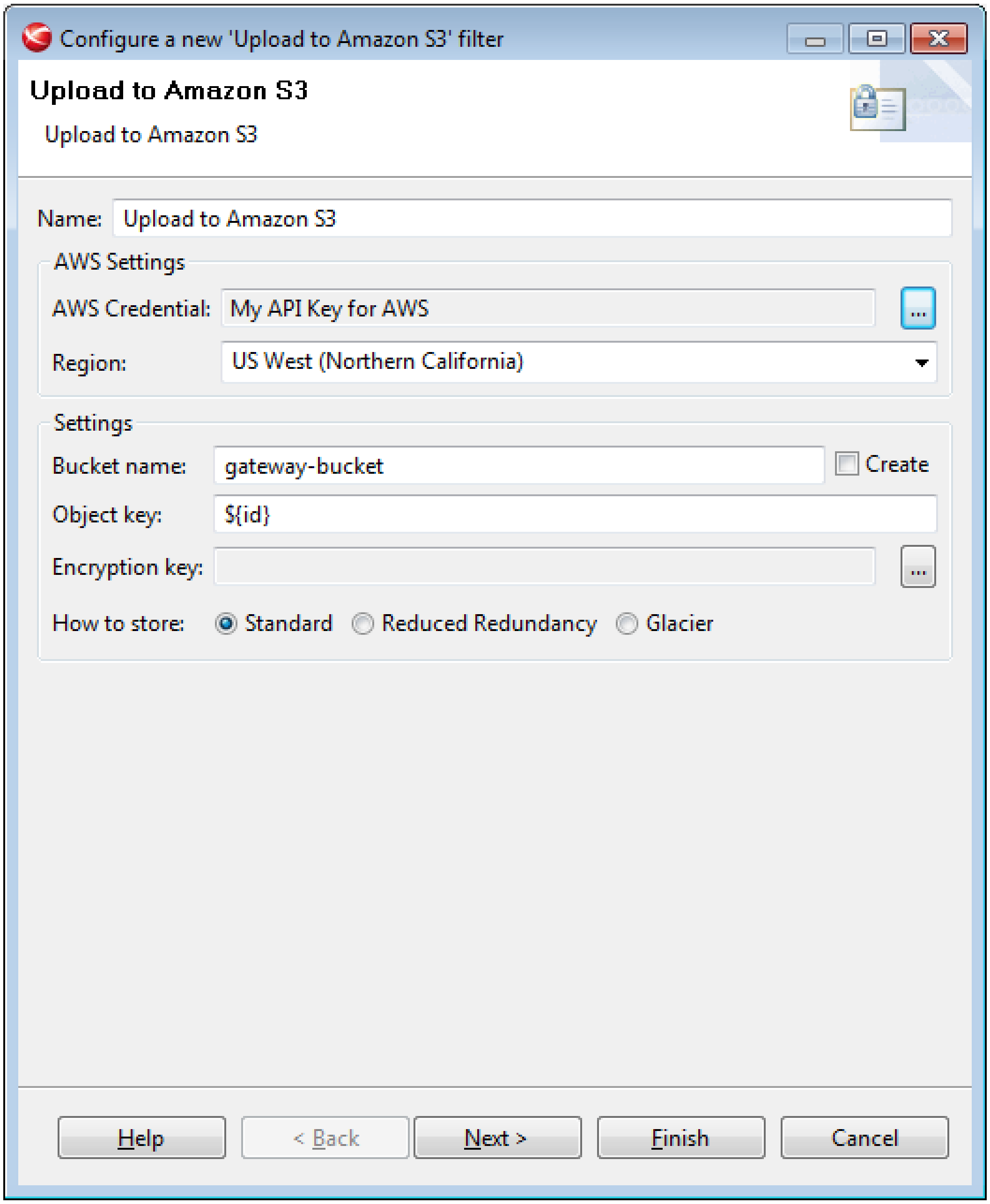
Encryption key:

How to store: ● Standard ○ Reduced Redundancy ○ Glacier

Configuring an SQS filter to process large messages

Amazon S3

Amazon S3 (Simple Storage Service) is an online storage web service offered by Amazon Web Services.



Configure a new 'Upload to Amazon S3' filter

Upload to Amazon S3

Upload to Amazon S3

Name: Upload to Amazon S3

AWS Settings

AWS Credential: My API Key for AWS

Region: US West (Northern California)

Settings

Bucket name: gateway-bucket □ Create

Object key: ${id}

Encryption key:

How to store: ● Standard ○ Reduced Redundancy ○ Glacier

Configuring the Upload to S3 filter

When configuring the filter:

1. Select the **AWS Credentials** (see above) to be used by the API Gateway when connecting to SNS.
2. Select the **region** (see above) appropriate for your deployment.
3. Select the **client settings** (See appendix below) for the API Gateway.
4. Enter the name of the **bucket** where the message payload will be written. Amazon buckets are like a container for your files. You can name your buckets the way you like but it should be unique across the Amazon system.
5. Enter the **Object key,** this is equivalent to the file name to be used.
6. If the file is to be encrypted supply the **encryption key**
7. Decide how the file is to be stored:
    a. **Standard**: Amazon S3 guarantes 99.999999999% durability and to sustain the concurrent

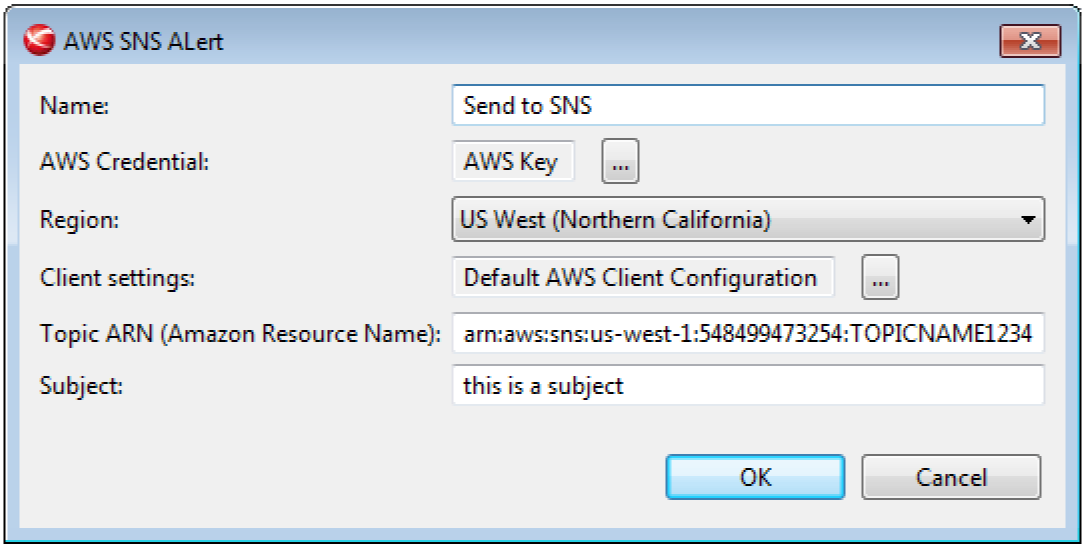
loss of data in two facilities

b. **Reduced Redundancy**: Reduced Redundancy Storage (RRS) is a storage option that enables customers to reduce their costs by storing non-critical, reproducible data at lower levels of redundancy than Amazon S3's standard storage. RRS provides a lower cost, less durable, highly available storage option that is designed to sustain the loss of data in a single facility.

c. **Glacier**: Glacier is an extremely low-cost storage service as storage for data archival. Amazon Glacier stores data for as little as $0.01 per gigabyte per month, and is optimized for data that is infrequently accessed and for which retrieval times of several hours are suitable.

Amazon SNS

Amazon SNS (Simple Notification Service) is a fully managed push messaging service. Amazon SNS makes it simple and cost-effective to push to mobile devices such as iPhone, iPad, Android, Kindle Fire, and internet connected smart devices, as well as pushing to other distributed services. Besides pushing cloud notifications directly to mobile devices, Amazon SNS can also deliver notifications by SMS text message or email, to Amazon Simple Queue Service (SQS) queues, or to any HTTP endpoint. To prevent messages from being lost, all messages published to Amazon SNS are stored redundantly across multiple availability zones. The API Gateway can send alert messages to Amazon SNS.



AWS SNS ALert

Name: Send to SNS

AWS Credential: AWS Key

Region: US West (Northern California)

Client settings: Default AWS Client Configuration

Topic ARN (Amazon Resource Name): arn:aws:sns:us-west-1:548499473254:TOPICNAME1234

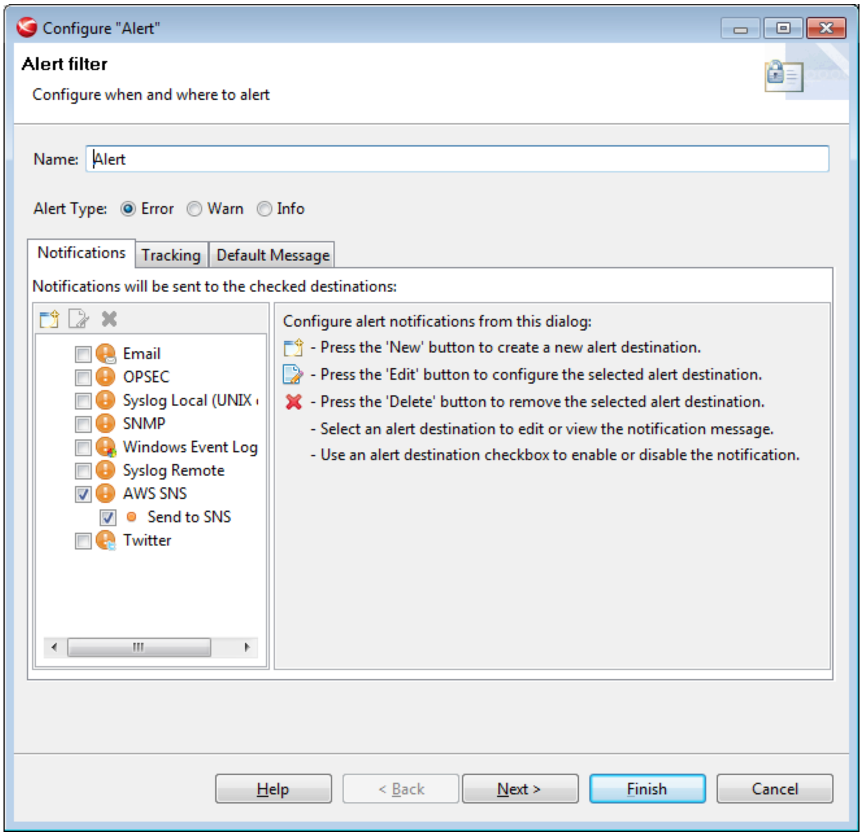
Subject: this is a subject

Configuring the AWS SNS Alert

To add an Amazon SNS alert to the API Gateway:

8. Click and expand the **Libraries** node in Policy Studio.
9. Press the **Add** button and select **Amazon SNS** option.
10. Provide a friendly name for the alert.
11. Select the **AWS Credentials** (see above) to be used by the API Gateway when connecting to SNS.
12. Select the **region** (see above) appropriate for your deployment.
13. Select the **client settings** (See appendix below) for the API Gateway.

14. Enter the **Topic ARN (Amazon Resource Name)** where the alert will be sent. When a topic is created, Amazon SNS will assign a unique ARN (Amazon Resource Name) to the topic, which will include the service name (SNS), region, AWS ID of the user and the topic name. The ARN will be returned as part of the API call to create the topic. Whenever a publisher or subscriber needs to perform any action on the topic, they should reference the unique topic ARN. For example, the following is the ARN for a topic named "mytopic" created by a user with the AWS account ID "123456789012" and hosted in the US East region: arn:aws:sns:us-east-1:1234567890123456:mytopic
15. Enter the **subject** of the alert to be sent to SNS



Alert filter

Configure when and where to alert

Name: Alert

Alert Type: ● Error ○ Warn ○ Info

Notifications will be sent to the checked destinations:

- ☐ Email
- ☐ OPSEC
- ☐ Syslog Local (UNIX
- ☐ SNMP
- ☐ Windows Event Log
- ☐ Syslog Remote
- ☑ AWS SNS
  - ☑ Send to SNS
- ☐ Twitter

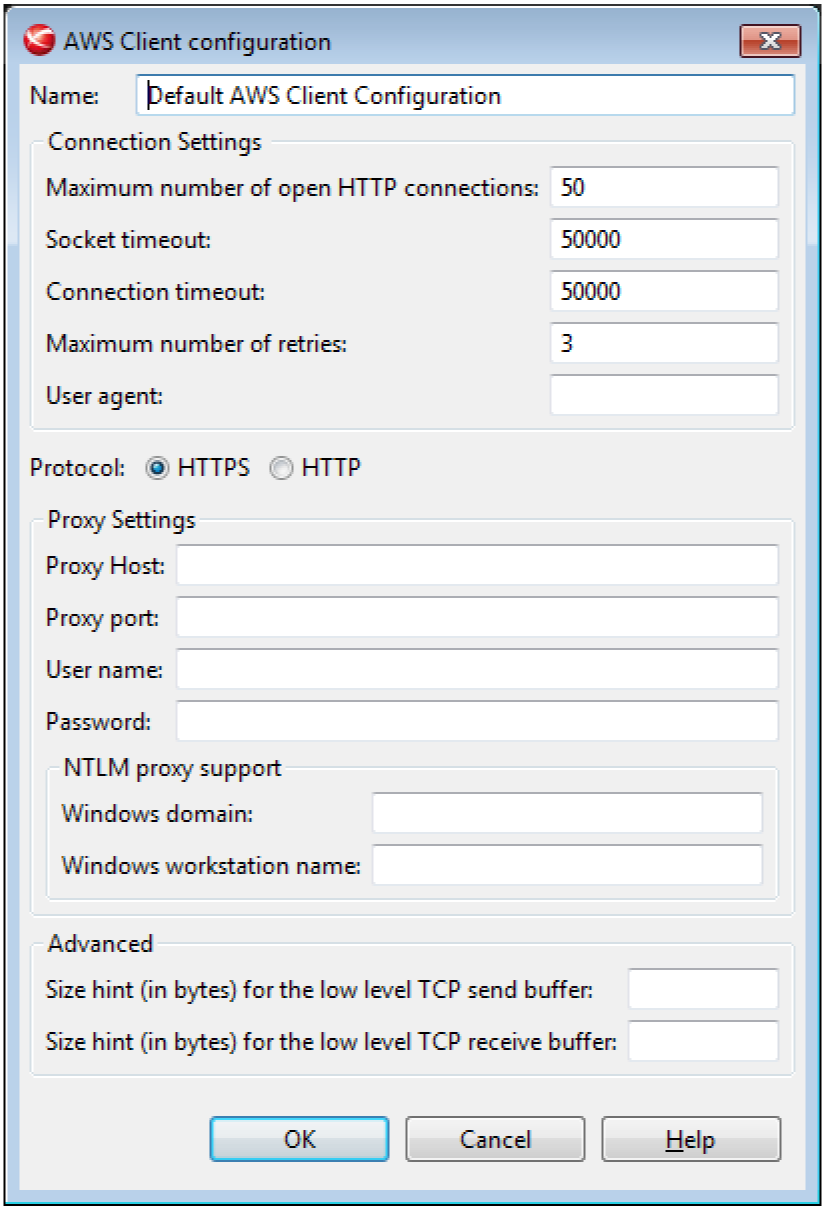
Configure alert notifications from this dialog:

- Press the 'New' button to create a new alert destination.
- Press the 'Edit' button to configure the selected alert destination.
- Press the 'Delete' button to remove the selected alert destination.
- Select an alert destination to edit or view the notification message.
- Use an alert destination checkbox to enable or disable the notification.

Configuring the AWS SNS aler

The created SNS alert destination can be used the same as any alert destination in the **Alert filter** (see above).

Appendix A - Client configuration



AWS Client configuration

Name: Default AWS Client Configuration

Connection Settings

Maximum number of open HTTP connections: 50

Socket timeout: 50000

Connection timeout: 50000

Maximum number of retries: 3

User agent:

Protocol: ● HTTPS ○ HTTP

Proxy Settings

Proxy Host:

Proxy port:

User name:

Password:

NTLM proxy support

Windows domain:

Windows workstation name:

Advanced

Size hint (in bytes) for the low level TCP send buffer:

Size hint (in bytes) for the low level TCP receive buffer:

Configuring the AWS client configuration settings

The API Gateway is a client of AWS as such it is possible to define a client profile by which is will connect to AWS. Possible items that can be set:

- The **maximum number of open HTTP connections**,the default is 50.
- **Socket timeout**: The amount of time to wait (in milliseconds) for data to be transferred over an established, open connection before the connection is timed out. A value of 0

means infinity, and is not recommended.

- **Connection timeout**: The amount of time to wait (in milliseconds) when initially establishing a connection before giving up and timing out. A value of 0 means infinity, and is not recommended.
- Set the maximum number of **retry** attempts for failed retryable requests (ex: 5xx error responses from services).
- Set the HTTP **user agent** header to send with all requests.
- Set the **protocol** (HTTP or HTTPS) to use when connecting to Amazon Web Services.
- Proxy Setttings
    - Optionally specifies the proxy host to connect through
    - Optionally specifies the port on the proxy host to connect through.
    - Optionally specifies the user name to use when connecting through a proxy.
    - Optionally specifies the password to use when connecting through a proxy.
    - Optional Windows domain name for configuring NTLM proxy support.
    - Optional Windows workstation name for configuring NTLM proxy support.
- Advanced
- Send buffer size hint: Optional size hint (in bytes) for the low level TCP send buffer. This is an advanced option for advanced users who want to tune low level TCP parameters to try and squeeze out more performance.
- Receive buffer size hint: Optional size hint (in bytes) for the low level TCP receive buffer. This is an advanced option for advanced users who want to tune low level TCP parameters to try and squeeze out more performance