Simple Message Notification

User Guide

Issue	01
Date	2023-01-31

HUAWEI

HUAWEI TECHNOLOGIES CO., LTD.

Trademarks and Permissions


and other Huawei trademarks are trademarks of Huawei Technologies Co., Ltd.
All other trademarks and trade names mentioned in this document are the property of their respective holders.

Notice

The purchased products, services and features are stipulated by the contract made between Huawei and the customer. All or part of the products, services and features described in this document may not be within the purchase scope or the usage scope. Unless otherwise specified in the contract, all statements, information, and recommendations in this document are provided "AS IS" without warranties, guarantees or representations of any kind, either express or implied.

The information in this document is subject to change without notice. Every effort has been made in the preparation of this document to ensure accuracy of the contents, but all statements, information, and recommendations in this document do not constitute a warranty of any kind, express or implied.

Contents

1 Topic Management

1.1 Creating a Topic

Scenarios

A topic is a specified event to publish messages and subscribe to notifications. It serves as a message sending channel, where publishers and subscribers can interact with each other.

Creating a Topic

1. Log in to the management console.
2. Click  on the upper left to select the desired region and project.
3. and select **Simple Message Notification** under **Application Service**.
   The SMN console is displayed.
4. In the navigation pane, choose **Topic Management** > **Topics**.
   The **Topics** page is displayed.
5. In the upper right corner, click **Create Topic**.
6. Enter a topic name and display name.

Table 1-1 Parameter descriptions

Parameter	Description
Topic Name	Topic name, which: • Contains only letters, digits, hyphens (-), and underscores (_), and must start with a letter or digit. • Contains 1 to 255 characters. • Must be unique and cannot be modified once the topic is created.

Parameter	Description
Display Name	Message sender name which can contain up to 192 characters **NOTE** After you specify a display name, the sender in email messages will be presented as *Display name*<**username@example.com**>. Otherwise, the sender will be **username@example.com**.
Enterprise Project	Centrally manages cloud resources and members by project.
Tag	A tag is a key-value pair. Tags identify cloud resources so that you can easily categorize and search for your resources. • • You can add up to 20 tags to each topic.

7. Click **OK.**

   The topic you created is displayed in the topic list. The system generates a topic URN, which is the unique resource identifier of the topic and cannot be changed.

8. Click the name of the topic to view its details, including the URN, display name, and subscriptions.

1.2 Modifying the Display Name of a Topic

Scenarios

You have created a topic and want to modify its display name.

Modifying the Display Name of a Topic

1. and select **Simple Message Notification** under **Application Service**.

   The SMN console is displayed.

2. In the navigation pane, choose **Topic Management** > **Topics**.

   The **Topics** page is displayed.

3. Locate the topic, click **More** in the **Operation** column, and select **Modify Display Name**.

   Enter a new display name.

   NOTE

   After you specify a display name, the sender in email messages will be presented as *display name<username@example.com>*. Otherwise, the sender will be *username@example.com*.

4. Click **OK**.

1.3 Configuring Topic Policies

1.3.1 Basic Mode

Only the topic creator has the right to configure topic policies. Using topic policies, you can specify which users and cloud services can perform which topic operations, for example, querying topic details and publishing messages. Topic creators always have permissions over a topic even if they grant topic permissions to other users.

Configuring Topic Policies in Basic Mode

1. and select **Simple Message Notification** under **Application Service**.

   The SMN console is displayed.

2. In the navigation pane, choose **Topic Management** > **Topics**.

   The **Topics** page is displayed.

3. Locate a topic, click **More** under **Operation**, and select **Configure Topic Policy**.

4. In the **Configure Topic Policy** dialog box, configure the topic policy in basic mode.

   The basic mode simply specifies which users or cloud services have permissions to publish messages to the topic. For details, see **Table 1-2**.

   The advanced mode provides more flexible permission settings. For details, see **Advanced Mode**.

   Table 1-2 Description for configuring topic policies in basic mode

Item	Parameter	Description
Users who can publish messages to this topic	Topic creator	Only the topic creator has the permission to publish messages to the topic.
	All users	All users have the permission to publish messages to the topic.

Item	Parameter	Description
	Specified user accounts	Only specified users have the permission to publish messages to the topic. Users are specified in the following format: urn:csp:iam::domainId:root. *domainId* indicates the account IDs of the users. You only need to enter the account ID and click **OK**. The system completes all other required information for you. SMN does not limit the number of IDs you enter, but the total size of a topic policy cannot exceed 30 KB. All settings in basic mode are also presented in the advanced policy. You can modify them in advanced mode if necessary. For details, see **Advanced Mode**. To obtain your account ID, log in to the SMN console. In the upper right corner, hover the mouse over your login account, and select from the drop-down list.
Services that can publish messages to this topic	Example: **OBS** The services that can publish messages to a topic vary in different regions.	The selected cloud services have operation permissions of the topic. **NOTE** By default, Cloud Eye and Anti-DDoS have the permission to publish messages to topics created by all users. For details about how to use SMN in other cloud services, see user guides of the related services.

1.3.2 Advanced Mode

The advanced mode provides a more flexible topic policy. You can specify which users and cloud services can perform which topic operations, for example, querying topic details, modifying topics, publishing messages, and deleting topics.

Introduction to Topic Policies

A topic policy is configured by a topic creator to allow or disallow other users or cloud services to perform specified operations to a topic. **Table 1-3** lists the elements consisting of a topic policy.

Table 1-3 Topic policy elements

Item	Description	Constraint
Version	Policy specification version	Only **2016-09-07** is supported.

Item	Description	Constraint
Id	Policy ID, which uniquely identifies a policy	The policy ID must be specified.
Statement	You can use statements to grant topic operation permissions to other users or cloud services. Each topic policy may contain one or more statements.	A policy must contain at least one statement. For details about elements in a statement, see **Statement Elements**.

The following is an example topic policy, which contains two statements, **Statement1** and **Statement2**.

```
{
   "Version": "2016-09-07",
   "Id": "access_policy_01",
   "Statement": [
     {Statement1},
     {Statement2}
   ]
}
```

Statement Elements

The following example shows how to modify a topic policy. **Table 1-4** lists the statement elements.

```
{
   "Version": "2016-09-07",
   "Id": "__default_policy_ID",
   "Statement": [
         //The first statement
     {
         "Sid": "__user_pub_0",
         "Effect": "Allow",
         "Principal": {
            "CSP": [
               "urn:csp:iam::123456789:root",
               "urn:csp:iam::987654321:root"
            ]
         },
         "Action": [
            "SMN:Publish",
            "SMN:QueryTopicDetail"
         ],
         "Resource": "urn:smn:regionId:e23bf08ebb924730b452426c60849564:ECM_BKS_Topic"
     },
     //The second statement
     {
         "Sid": "__service_pub_0",
         "Effect": "Allow",
         "Principal": {
            "Service": [
               "obs"
            ]
         },
```



```
        "Action": [
            "SMN:Publish",
            "SMN:QueryTopicDetail"
        ],
        "Resource": "urn:smn:regionId:e23bf08ebb924730b452426c60849564:ECM_BKS_Topic"
    }
  ]
}
```

Table 1-4 Statement elements description

Element	Description	Constraint
Sid	Statement ID	The statement ID must be unique, for example, **statement01** or **statement02**.
Effect	Statement effect	The effect can be **Allow** or **Deny**.
Principal NotPrincipal	• **Principal**: object the statement applies to • **NotPrincipal**: object the statement does not apply to The following two types of objects are supported: – **CSP** indicates cloud users. – **Service** indicates cloud services.	Either the **Principal** or **NotPrincipal** element must be configured. If you enter **CSP**, you must specify user information in the format **urn:csp:iam::domainId:root**. Obtain the account ID of each user you specify. If you enter **Service**, you must specify the cloud service names in lowercase.
Action NotAction	• **Action**: allowed statement action • **NotAction**: statement action not allowed You can use a wildcard character to configure a type of actions, for example, **SMN:Update*** and **SMN:Delete***. If you only enter a wildcard character (*) in a statement, all supported actions are configured.	Either the **Action** or **NotAction** element must be configured. The following actions are supported: • SMN:UpdateTopic • SMN:DeleteTopic • SMN:QueryTopicDetail • SMN:ListTopicAttributes • SMN:UpdateTopicAttribute • SMN:DeleteTopicAttributes • SMN:DeleteTopicAttributeByName • SMN:ListSubscriptionsByTopic • SMN:Subscribe • SMN:Unsubscribe • SMN:Publish For details about mappings between actions and APIs, see **Mappings Between SMN Actions and APIs**.

Element	Description	Constraint
Resource NotResource	• **Resource**: topic a statement applies to • **NotResource**: topic the statement does not apply to	Either the **Resource** or **NotResource** element must be configured. Then, enter a topic URN.
Condition	(Optional) Condition under which a policy statement takes effect	Enter supported conditional operators and keywords. For details, see **Condition Elements**.

Condition Elements

Conditions determine whether a statement takes effect. They enable you to configure more fine-grained control over topic permissions. **Table 1-5** lists elements in a condition.

Table 1-5 Condition elements

Item	Description	Constraint
Operation	Character strings, digits, date, or time to be matched in the operation	The time you entered must comply with ISO 8601 specifications. For details, see **Table 1-6**.
Operation keyword	Object on which the condition operator takes effect	The operation keyword must be specified. For details, see **Table 1-7**.

A statement allows the requested operation only when all conditions in the statement are met. Otherwise, the operation will be denied.

As shown in **Figure 1-1**, when a condition contains multiple operators, for example, **condition1** and **condition2**, an AND operation is executed.

When the operator **condition1** contains multiple keywords, for example, **conditionKey1** and **conditionKey2**, an AND operation is executed.

When the keyword **conditionKey1** contains multiple values, for example, **value11** and **value12**, an OR operation is executed.

Figure 1-1 Condition logic

An example condition is as follows.

```
"Condition": {
   "DateLessThan":{
       "csp:CurrentTime":"2016-11-07T15:35:00Z"
   },
   "StringLike": {
       "smn:Endpoint":["*@gmail.com","*@hotmail.com"]
   }
}
```

Table 1-6 Condition operators

Category	Operator	Description
String	StringEquals	Match a string (case-sensitive).
	StringNotEquals	Exclude a string (case-sensitive).
	StringEqualsIgnoreCase	Match a string (case-insensitive).
	StringNotEqualsIgnoreCase	Exclude a string (case-insensitive).
	StringLike	Match a string. The value can contain one or more wildcard characters (*).
	StringNotLike	Exclude a string. The value can contain one or more wildcard characters (*).
Numeric	NumericEquals	Match an integer or decimal.
	NumericNotEquals	Exclude an integer or decimal.
	NumericLessThan	Match any number less than an integer or decimal.
	NumericLessThanEquals	Match any number less than or equal to an integer or decimal.
	NumericGreaterThan	Match any number greater than an integer or decimal.
	NumericGreaterThanEquals	Match any number greater than or equal to an integer or decimal.

Category	Operator	Description
Date	DateEquals	Match a date.
	DateNotEquals	Exclude a date.
	DateLessThan	Match any time earlier than a date and time point.
	DateLessThanEquals	Match any time earlier than or equal to a date and time point.
	DateGreaterThan	Match any time later than a date and time point.
	DateGreaterThanEquals	Match any time later than or equal to a date and time point.
Bool	Bool	Match a Boolean value.

Table 1-7 Condition keywords

Keyword	Description
csp:CurrentTime	Current time
smn:Protocol	Protocol of a subscription, which is valid only for the SMN:Subscribe action
smn:Endpoint	Endpoint of a subscription, which is valid only for the SMN:Subscribe action

1.4 Adding a Subscription to a Topic

Scenarios

To deliver messages published to a topic to endpoints, you must add the subscription endpoints to the topic.

To Add a Subscription

1. and select **Simple Message Notification** under **Application Service**.

   The SMN console is displayed.
2. In the navigation pane, choose **Topic Management** > **Topics**.

   The **Topics** page is displayed.
3. Locate the topic that you want to add a subscription to. In the **Operation** column, click **Add Subscription**.

   The **Add Subscription** dialog box is displayed.

4. Specify the subscription protocol and endpoints.

   You can enter up to 10 endpoints, each on a separate line.

   - Email

     Enter a valid email address, for example, **username@example.com**.

     Subscribers will receive a subscription confirmation email valid in 48 hours and must confirm the subscription to receive messages published to the topic.

   - HTTP or HTTPS

     Enter a public network URL, for example, **http://example.com/notification/action**. HTTP/HTTPS subscribers must confirm their subscriptions. For details about HTTP/HTTPS messages, see **Introduction**.

   - SMS

     Enter a valid mobile number, for example, **18512345678**.

     Subscribers will receive a subscription confirmation message valid in 48 hours and must confirm the subscription to receive messages published to the topic.

5. Click **OK**.

   The subscription you added is displayed in the subscription list.

   NOTE

   - To prevent malicious users from attacking subscription endpoints, SMN limits the number of confirmation messages that can be sent to an endpoint within a specified period of time. For details, see **Traffic Control over Subscription Confirmation**.
   - SMN does not check whether subscription endpoints exist when you add subscriptions. However, subscribers will not receive notification messages until they confirm their subscriptions.
   - After you add a subscription, SMN sends a confirmation message to the subscription endpoint. The message contains a link for confirming the subscription. The subscription confirmation link is valid within 48 hours. Confirm the subscription on your mobile phone, mailbox, or other endpoints in time.

1.5 Publishing a Message

1.5.1 Introduction

SMN enables you to publish messages in the following formats:

- Text
- JSON
- Template

After you publish a message to a topic, SMN will deliver the message to all confirmed subscription endpoints in the topic.

For SMS endpoints, if an SMS message is oversized, the system divides it into multiple parts when sending it to subscribers. However, you must note that SMN only sends the first two parts of the SMS message and does not send any additional parts. You are charged based on the actual number of messages sent to the subscribers.

You must ensure that firewall policies of the HTTP or HTTPS endpoints allow SMN to send messages over the Internet. An SMN HTTP or HTTPS message consists of a message header and body. For details, see **HTTP or HTTPS Message Format**.

1.5.2 Publishing a Text Message

Scenarios

After you publish a text message to a topic, SMN will deliver the message to all confirmed subscription endpoints in the topic.

Prerequisites

Subscribers in the topic must have confirmed the subscription, or they will not be able to receive any messages.

Procedure

1. and select **Simple Message Notification** under **Application Service**.

   The SMN console is displayed.
2. In the navigation pane, choose **Topic Management** > **Topics**.

   The **Topics** page is displayed.
3. In the topic list, locate the topic that you need to publish a message to and click **Publish Message** in the **Operation** column.
4. Configure the required parameters based on **Table 1-8**.

   The topic name is provided by default and cannot be changed.

   Table 1-8 Parameter descriptions

Parameter	Description
Subject	(Optional) The message subject must be fewer than 512 bytes.
Message Format	The message format can be **Text**, **JSON**, or **Template**. In this context, select **Text**. • **Text**: common text message • **JSON**: JSON message • **Template**: template message. For details, see **Message Template Management**
Message	This is the message content, which cannot be left blank nor exceed 256 KB.

5. Click **OK**.

   SMN delivers your message to all subscription endpoints. For details about the messages received by each endpoint, see **Messages of Different Protocols**.

1.5.3 Publishing a JSON Message

Scenarios

In a JSON message, you can specify different message content for different protocols, including SMS, email, HTTP, and HTTPS.

Prerequisites

Subscribers in the topic must have confirmed the subscription, or they will not be able to receive any messages.

Procedure

1. and select **Simple Message Notification** under **Application Service**.

   The SMN console is displayed.

2. In the navigation pane, choose **Topic Management** > **Topics**.

   The **Topics** page is displayed.

3. In the topic list, locate the topic that you need to publish a message to and click **Publish Message** in the **Operation** column.

4. Configure the required parameters.

   The topic name is provided by default and cannot be changed.

   Configure the required parameters based on **Table 1-8**.

   Select **JSON** for **Message Format**.

   Manually type the JSON message in the **Message** box or click **Generate JSON Message** to generate it automatically. The total size of a JSON message cannot exceed 256 KB.

   - If you choose to manually type the JSON message, see **JSON Message Format** for detailed requirements.
   - If you choose to automatically generate the JSON message, proceed with steps **5** through **8**.

5. Click **Generate JSON Message**.

6. Enter your message content, for example **This is a default message.**, in the **Message** box and select the desired message protocols.

   The size of a JSON message varies depending on the protocol combinations. As you type in the message content, the system will calculate the number of bytes you have entered, the size of the JSON message, and how many bytes are left. The total size of a JSON message includes braces, quotation marks, spaces, line breaks, and message content. For details about how to calculate the size of a JSON message, see **Calculation on the Size of a JSON Message** in **JSON Message Format**.

7. Click **OK**.

8. Modify the message content for each protocol so that different messages are sent to endpoints of different protocols. The system generates JSON-formatted content that includes a default message and content for each protocol. When SMN fails to match any specific message protocol, it sends the default message. For detailed, see **JSON Message Format**.

9. Click **OK**.

   SMN delivers your message to all subscription endpoints. For details about the messages received by each endpoint, see **Messages of Different Protocols**.

1.5.4 Publishing a Template Message

Scenarios

Message templates contain fixed message content. If you need to send the same or similar messages multiple times, you can create a message template for quick message sending.

You can create different templates for different protocols using the same template name so that each type of subscribers can receive customized messages. Templates contain variables as the placeholders to represent changeable content that you can replace with your own message content. Note that you must create a template whose **Protocol** is **Default**, or the system will not allow you to publish messages using this template name.

When you are creating messages using a template, select a template name. The system will list all variables in the following protocol sequence: **Default**, **SMS**, **Email**, **FunctionGraph (function)**, **HTTP**, and **HTTPS**. The same variables are listed only once even if they are used in multiple protocols, and the protocols they support are listed after each variable. Specify content for each variable in the message template, and SMN replaces them with the content you entered. If you do not enter any content for a variable, the system will treat it as empty when sending messages.

SMN tries to match different types of subscribers to the template protocols. If there is no template for a specified protocol, SMN will use the default template to send messages to subscribers of that protocol.

This section describes how to publish messages using a template. For more details about message templates, see **Message Template Management**.

Prerequisites

Subscribers in the topic must have confirmed the subscription, or they will not be able to receive any messages.

Creating a Message Template

1. Log in to the management console.
2. Click  on the upper left to select the desired region and project.
3. and select **Simple Message Notification** under **Application Service**.

   The SMN console is displayed.
4. In the navigation pane, choose **Topic Management** > **Message Templates**.
5. In the upper right corner, click **Create Message Template**. For details, see **Creating a Message Template** in **Message Template Management**.

   For example, the template information is as follows:

  - **Template Name**: **tem_001**
  - **Protocol**: **Default**
  - **Content**: **The Arts and Crafts Exposition will be held from {startdate} through {enddate}. We sincerely invite you to join us.**

Publishing a Template Message

1. Log in to the management console.
2. Click  on the upper left to select the desired region and project.
3. Under **Application**, select **Simple Message Notification**.

   The SMN console is displayed.
4. In the navigation pane, choose **Topic Management** > **Topics**.

   The **Topics** page is displayed.
5. In the topic list, locate the topic that you need to publish a message to and click **Publish Message** in the **Operation** column.
6. Configure the required parameters.

   The topic name is provided by default and cannot be changed.

   Select **Template** for **Message Format**. Then, manually type the template content in the **Message** box or click **Generate Template Message** to generate it automatically. The template message content cannot exceed 256 KB.

   - If you choose to manually type the template message, see section "Template Message Format" in the *Simple Message Notification User Guide* for detailed requirements.
   - If you choose to automatically generate the template message, proceed with **7** through **10**.
7. Click **Generate Template Message**.
8. Select a template name, for example, **tem_001**, and enter values for the variables.

   The system replaces the variables with the message content you specified. The protocols configured in the template are displayed after each variable. Only the **Default** protocol is specified in **tem_001**. Therefore, all confirmed subscribers in the topic will receive the message content in the default template.
9. Click the **Preview** tab, and click **Message Preview** to preview the message.

   In this example, the message generated is "The Arts and Crafts Exposition will be held from February 10 through February 21. We sincerely invite you to join us."
10. Click **OK**.

    The message that is generated contains the template name and variables.
11. Click **OK**.

    SMN delivers your message to all subscription endpoints. For details about messages for different protocols, see section "Messages of Different Protocols" in *Simple Message Notification User Guide*.

1.6 Deleting a Topic

1. Log in to the management console.
2. Click  on the upper left to select the desired region and project.
3. and select **Simple Message Notification** under **Application Service**.
   The SMN console is displayed.
4. In the navigation pane, choose **Topic Management** > **Topics**.
   The **Topics** page is displayed.
5. Locate a topic, click **More** in the **Operation** column, and select **Delete**.
6. Click **OK**.

NOTE

Deleting a topic deletes all its subscriptions.

2 Subscription Management

2.1 Adding a Subscription

Scenarios

To deliver messages published to a topic to endpoints, you must add the subscription endpoints to the topic. Endpoints can be email addresses, phone numbers, and HTTP/HTTPS URLs. After you add endpoints to the topic and the subscribers confirm the subscription, they are able to receive messages published to the topic.

You can add multiple subscriptions to each topic. This section describes how to add a subscription to a topic you created or a topic that you have permissions for and how to delete a subscription.

Adding a Subscription

1. Log in to the management console.
2. Click on the upper left to select the desired region and project.
3. and select **Simple Message Notification** under **Application Service**.
   The SMN console is displayed.
4. In the navigation pane, choose **Topic Management** > **Subscriptions**.
5. In the upper right corner, click **Add Subscription**.
   The **Add Subscription** dialog box is displayed.
6. Specify the required subscription information.
   a. Click beside the **Topic Name** box to select a topic.
   b. Specify the subscription protocol and endpoints.

Table 2-1 Parameter descriptions

Parameter	Description
Topic Name	Name of the topic
Protocol	The following protocols are supported. They include SMS, Email, HTTP, HTTPS, DingTalk chatbot, WeCom chatbot, and Lark chatbot. **NOTE** • DingTalk chatbot, WeCom chatbot, and Lark chatbot are in the open beta test (OBT). To use these functions, submit a service ticket to apply for the OBT. • After the OBT is enabled, the IAM user token will be invalid. You need to log in to the console again to use the token. When calling an API, you need to obtain a new token. • The number of WeCom, DingTalk, and Lark messages that can be sent is limited, and the recipient system may be faulty. As a result, messages may fail to be sent. In this case, SMN does not ensure successful message delivery.
Endpoint	Subscription endpoint. You can add up to 10 SMS, email, HTTP, or HTTPS endpoints, one in each line. • **SMS**: Enter one or more valid phone numbers. Example: **+353876612345** **+353 876612345** **+353-876612345** **+353/876612345** • **Email**: Enter one or more valid email addresses. Example: **username@example.com** **username2@example.com** • **HTTP**: Enter one or more public network URLs. Example: **http://example.com/notification/action** • **HTTPS**: Enter one or more public network URLs. Example: **https://example.com/notification/action**

7. Click **OK**.

   The subscription you added is displayed in the subscription list.

NOTE

- To prevent malicious users from attacking subscription endpoints, SMN limits the number of confirmation messages that can be sent to an endpoint within a specified period of time. For details, see section "Traffic Control on Subscription Confirmation" in *Simple Message Notification User Guide*.
- SMN does not check whether subscription endpoints exist when you add subscriptions. However, subscribers will not receive notification messages until they confirm their subscriptions.
- After you add a subscription, SMN sends a confirmation message to your subscription endpoint. You can confirm the subscription within 48 hours through the confirmation link via your mobile phone, mailbox, or other endpoints.

2.2 Requesting Subscription Confirmation

Scenarios

If a subscriber does not receive the confirmation message, request confirmation again. You can send a subscription confirmation message to one or more subscription endpoints at a time. For details, see **Traffic Control over Subscription Confirmation**.

Requesting Subscription Confirmation

1. Log in to the management console.
2. Click  on the upper left to select the desired region and project.
3. and select **Simple Message Notification** under **Application Service**.

   The SMN console is displayed.
4. In the navigation pane, choose **Topic Management** > **Subscriptions**.
5. In the subscription list, select one or more subscriptions whose status is **Unconfirmed**.
6. Click **Request Confirmation** above the subscription list to send confirmation messages.
7. The subscribers confirm their subscriptions.

   NOTE

   - To prevent malicious users from attacking subscription endpoints, SMN limits the number of confirmation messages that can be sent to an endpoint within a specified period of time. For details, see **Traffic Control over Subscription Confirmation**.
   - SMN does not check whether subscription endpoints exist when you add subscriptions. However, subscribers will not receive notification messages until they confirm their subscriptions.
   - After you add a subscription, SMN sends a confirmation message to the subscription endpoint. The message contains a link for confirming the subscription. The subscription confirmation link is valid within 48 hours. Confirm the subscription on your mobile phone, mailbox, or other endpoints in time.

2.3 Canceling a Subscription

Scenarios

After you add subscriptions to a topic, the subscribers receive a confirmation message and need to confirm their subscriptions to receive notification messages published to the topic. If the subscribers no longer want to receive notifications from a topic, they can choose to cancel subscriptions.

CAUTION

The subscription management capability of SMN is open to subscribers. You must keep your subscription links secure to avoid being unable to receiving notifications or receiving unexpected notifications.

Canceling a Subscription

A subscriber can choose to cancel the subscription based on the protocol of the subscription endpoint:

- SMS: SMN does not provide a link to unsubscribe in SMS notification messages because of the message length limit. To cancel an SMS subscription, the subscriber needs to access the link provided in the subscription confirmation message and cancels the subscription on the web page.
- Email: SMN encloses a link to unsubscribe in email notifications. The subscriber can cancel the subscription by clicking the link. After the subscriber has canceled the subscription, SMN re-sends a subscription confirmation email which is valid for 48 hours, so that the subscriber can re-subscribe to the topic if they clicked the link by mistake.
- HTTP/HTTPS: SMN provides a link to unsubscribe in the HTTP/HTTPS message body. The subscriber can cancel the subscription by clicking the link. After the subscriber has canceled the subscription, the system returns **200** over HTTP and re-sends a subscription confirmation message which is valid for 48 hours, in case the subscriber has clicked the link by mistake. For details about the HTTP/HTTPS message header and body, see **Introduction**.

2.4 Deleting a Subscription

Scenarios

If one or multiple subscription endpoints do not need to receive messages published to a topic, you can delete them.

Deleting Subscriptions

1. Log in to the management console.
2. Click  on the upper left to select the desired region and project.

3. and select **Simple Message Notification** under **Application Service**.
   The SMN console is displayed.
4. In the navigation pane, choose **Topic Management** > **Topics**.
   The **Topics** page is displayed.
5. Click the topic name.
   The **Topic Details** page is displayed.
6. In the **Subscriptions** area, view the subscriptions to the topic.
7. Select one or more subscription endpoints and click **Delete**.
8. In the displayed **Delete Subscription** dialog box, click **OK**.

3 Message Template Management

Scenarios

Message templates contain fixed and changeable content and can be used to create and send messages more quickly. When you use a template to publish a message, you need to specify values for different variables in the template.

Message templates are identified by name, but you can create different templates with the same name as long as they are configured for different protocols. All template messages must include a **Default** template or they cannot be sent out. The **Default** template is used anytime a template has not been configured for a given protocol, but as long as there is a template for the protocol, then any subscriber who selected that protocol when they subscribed will receive a message using the corresponding template.

This section describes how to publish messages using a template.

Creating a Message Template

1. Log in to the management console.
2. Click  on the upper left to select the desired region and project.
3. and select **Simple Message Notification** under **Application Service**.
   The SMN console is displayed.
4. In the navigation pane, choose **Topic Management** > **Message Templates**.
5. In the upper right corner, click **Create Message Template**.
   The **Create Message Template** dialog box is displayed.
6. Specify the template name, protocol, and content.

Table 3-1 Parameters required for creating a message template

Parameter	Description
Template Name	Template name, which: • Contains only letters, digits, hyphens (-), and underscores (_), and must start with a letter or digit. • Can contain 1 to 64 bytes. • Cannot be modified once the template is created.
Protocol	Endpoint protocol of the template, which cannot be changed once the template is created The protocol can be **Default**, **SMS**, **HTTP**, **HTTPS**, or **Email**. If you do not specify a protocol, **Default** is used.
Content	Template content You can use variables as placeholders. Before you send messages using the template, SMN replaces the variables with the message content you specify. A variable can contain up to 21 characters and must start with a letter or digit. It can contain letters, digits, hyphens (-), underscores (_), and periods (.). The message template must meet the following requirements: • The template supports plain text only. • The template content cannot be left blank and cannot exceed 256 KB. • The template can contain up to 256 variables in total, but that includes redundant variables. For unique variables, there can be no more than 90. • When you send messages using a template, the message content you specify for each variable cannot exceed 1 KB.

For example, the template information is as follows:

- **Template Name**: **tem_001**
- **Protocol**: **Default**
- **Content**: **The Arts and Crafts Exposition will be held from {startdate} through {enddate}. We sincerely invite you to join us.**

7. Click **OK**.

   The template you created is displayed in the template list.

Modifying a Template

1. On the **Message Templates** page, locate the template to be modified in the template list.
2. Click **Modify** in the **Operation** column to change its content.

Deleting a Template

1. On the **Message Templates** page, locate the template to be deleted in the template list.
2. Click **Delete** in the **Operation** column.

4 SMN Operation Recording

4.1 Introduction

You can use Cloud Trace Service (CTS) to record information about SMN-related operations, including request content, source IP addresses, request senders, and when a request was sent, for future query, audit, and backtracking.

CTS can record operations performed on the management console, performed by calling APIs, and triggered within the CTS system.

4.2 Key SMN Operations Recorded by CTS

After you enable CTS, whenever an SMN API is called, the operation is recorded in a log file, which is then dumped to a specified OBS bucket for storage based on time and data changes.

However, if someone makes an API call to cancel a subscription without login, CTS will not record the operation. For example, if a subscriber clicks the link in an email notification to cancel the subscription, the unsubscribe API is called, but CTS does not record the operation.

Table 4-1 lists the SMN operations that will be recorded by CTS.

Table 4-1 SMN operations recorded by CTS

Operation	Resource	Trace Name
Creating a topic	Topic	createTopic
Deleting a topic	Topic	deleteTopic
Updating a topic	Topic	updateTopic
Updating a topic policy	Topic	updateTopicAttribute
Deleting all topic policies	Topic	deleteTopicAttributes

Operation	Resource	Trace Name
Deleting a specified topic policy	Topic	deleteTopicAttributeByName
Adding a subscription	Subscription	subscribe
Deleting a subscription	Subscription	delsubscribe
Creating a message template	Message template	createMessageTemplate
Creating message templates in batches	Message template	batchCreateMessageTemplate
Modifying a message template	Message template	updateMessageTemplate
Deleting a message template	Message template	deleteMessageTemplate

4.3 CTS Traces

Scenarios

After CTS is enabled, it starts recording operations on cloud resources. You can view the operation records of the last seven days on the management console.

This topic describes how to query or export the last seven days of operation records on the CTS console.

Procedure

1. Click **Service List** in the upper left corner. Under **Management & Governance**, select **Cloud Trace Service**.
2. In the left navigation pane, choose **Trace List**.
3. Specify filters as needed. The following filters are available:
   - **Trace Type**, **Trace Source**, **Resource Type**, and **Search By** Select the filter from the drop-down list.

     If you select **Resource ID** for **Search By**, specify a resource ID.

     If you select **Data** for **Trace Type**, you can only filter traces by tracker.
   - **Operator**: Select one or more specific operators from the drop-down list.
   - **Trace Status**: Available options include **All trace statuses**, **Normal**, **Warning**, and **Incident**. You can select only one of them.
   - Time range: You can query traces generated at any time range of the last seven days.
4. Click on the left of the required trace to expand its details.
5. Click **View Trace**.

CTS Log Entries

Each log entry consists of a trace in JSON format. A log entry indicates an SMN API request, including the requested operation, the time, operation parameters, and information about the user who sent the request. The user information is obtained from IAM.

The following example shows CTS log entries for the **CreateTopic**, **DeleteTopic**, and **UpdateTopic** actions:

```
{
  "time": "2017-02-15 14:21:50 GMT+08:00",
  "user": "xxx",
  "request": "xxx",
  "response": "xxx",
  "code": 200,
  "service_type": "SMN",
  "resource_type": "topic",
  "resource_id": "topicUrn instance",
  "source_ip": "127.0.0.1",
  "trace_name": "createTopic",
  "trace_rating": "normal",
  "trace_type": "ApiCall",
  "api_version": "2.0",
  "project_id": "tenantId instance",
  "record_time": "2017-02-15 14:21:50 GMT+08:00",
  "trace_id": "xxx"
}

{
  "time": "2017-02-15 14:12:15 GMT+08:00",
  "user": "xxx",
  "response": "xxx",
  "code": 200,
  "service_type": "SMN",
  "resource_type": "topic",
  "resource_id": "topicUrn instance",
  "source_ip": "127.0.0.1",
  "trace_name": "deleteTopic",
  "trace_rating": "normal",
  "trace_type": "ApiCall",
  "api_version": "2.0",
  "project_id": "tenantId instance",
  "record_time": "2017-02-15 14:12:15 GMT+08:00",
  "trace_id": "xxx"
}

{
  "time": "2017-02-13 15:38:30 GMT+08:00",
  "user": "xxx",
  "request": "xxx",
  "response": "xxx",
  "code": 200,
  "service_type": "SMN",
  "resource_type": "topic",
  "resource_id": "topicUrn instance",
  "source_ip": "127.0.0.1",
  "trace_name": "updateTopic",
  "trace_rating": "normal",
  "trace_type": "ApiCall",
  "api_version": "2.0",
  "project_id": "tenantId instance",
  "record_time": "2017-02-13 15:38:30 GMT+08:00",
  "trace_id": "xxx"
}
```


5 Logs

Scenarios

You can use logs when you want to know the statuses of messages published to a topic. Protocols including SMS, email, HTTP and HTTPS are supported. Before configuring logs, you need to interconnect SMN with Log Tank Service (LTS) and have created a log group and log stream to be associated.

Configuring Cloud Logs

1. Create a log group.
   a. Log in to the management console.
   b. Click  on the upper left to select the desired region and project.
   c. Select **Log Tank Service** under **Management & Governance**.
      The LTS console is displayed.
   d. In the navigation pane on the left, choose **Log Management**.
      The **Log Management** page is displayed.
   e. Click **Create Log Group**. In the displayed dialog box, enter a log group name.
   f. Click **OK**.
2. Create a log stream.
   a. Locate the created log group and click its name.
   b. Click **Create Log Stream**. In the displayed dialog box, enter a name for the log stream.
   c. Click **OK**.

Configuring Message Transmission Logs

On the SMN console, configure logs.

1. Set this parameter on the **Create Topic** page.
   a. Log in to the management console.
   b. In the upper left corner of the page, click  and select the desired region and project.

c. In the upper left corner of the page, click . Select **Simple Message Notification** under **Management & Governance**.

   The SMN console is displayed.

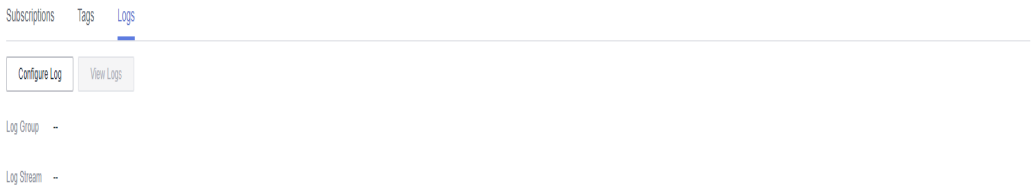
d. In the navigation pane on the left, choose **Topic Management** > **Topics**.

   The **Topics** page is displayed.

e. On the **Topics** page, click the name of the topic to be configured.

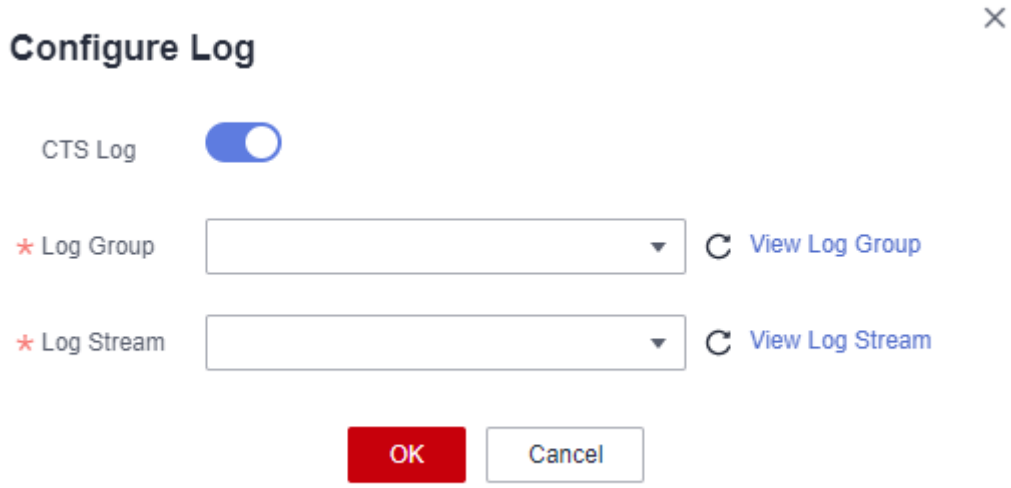
f. On the displayed page, click the **Message Transfer Logs** tab, and click **Configure Access Logs**.

   Figure 1 Message transmission log



g. Click **Start Logging** and select the log group and log stream that you have created on the LTS console.

   Figure 2 Configuring logs



h. Click **OK**.

Viewing Access Logs

You can view details about logs you configured.

The log format is as follows.

```
{"message_id":"$message_id","project_id":"$project_id","topic_urn":"$topic_urn","subscriber_urn":"$subscriber_urn","protocol_name":"$protocol_name","endpoint":"$endpoint","status":"$status","http_code":$http_code,"create_time":"$create_time","send_time":"$send_time"}
```

The log format cannot be modified. Table 1 describes the log fields.

Table 5-1 Parameters in a FunctionGraph (function) message

Parameter	Type	Description
message_id	String	Message ID
project_id	String	Project ID
topic_urn	String	Resource identifier of a topic, which is unique
subscriber_urn	String	Resource identifier of a subscription, which is unique
protocol_name	String	Specifies the subscription protocol. (Different protocols indicate different types of endpoints to receive messages.) The following protocols are supported: **Email**: The endpoints are email addresses. **SMS**: The endpoints are phone numbers. **FunctionGraph**: FunctionGraph (function) transmission protocol. The endpoint is a function. **FunctionStage**: FunctionStage (workflow) transmission protocol. The endpoint is a function workflow. **HTTP** and **HTTTPS**: The endpoints are URLs.
endpoint	String	Message receiving endpoint
status	String	Message status. The options are as follows: **DELIVERED**: The message has been delivered. **FAIL_DELIVERED**: The message fails to be sent. **REJECTS**: The message has been rejected. The flow control mechanism is triggered.
http_code	Integer	HTTP return code. Only HTTP/HTTPS messages are supported.

Parameter	Type	Description
create_time	String	Time when a message was created. The UTC time is in **YYYY-MM-DDTHH:MM:SSZ** format.
send_time	String	Specifies the time when the message was sent. The UTC time is in **YYYY-MM-DDTHH:MM:SSZ** format.

Example Log

{"message_id":"1ae49922602a42fc83acb9689a2eb5f4","project_id":"5a9f32e4f1ec4bbe9695ff9da51c2925","topic_urn":"urn:smn:cn-north-1:5a9f32e4f1ec4bbe9695ff9da51c2925:demo","subscriber_urn":"urn:smn:cn-north-1:5a9f32e4f1ec4bbe9695ff9da51c2925:demo:b55c3c6fa7cd471b9f24818d530a8740","protocol_name":"https","endpoint":"https://127.0.0.1:443/https","status":"DELIVERED","http_code":200,"create_time":"2022-11-01T00:00:00Z","send_time":"2022-11-01T00:00:10Z"}

The following table describes the fields in the log.

Table 5-2 Example values of fields in the log

Parameter	Example Value
message_id	1ae49922602a42fc83acb9689a2eb5f4
project_id	5a9f32e4f1ec4bbe9695ff9da51c2925
topic_urn	urn:smn:cn-north-1:5a9f32e4f1ec4bbe9695ff9da51c2925:demo
subscriber_urn	urn:smn:cn-north-1:5a9f32e4f1ec4bbe9695ff9da51c2925:demo:b55c3c6fa7cd471b9f24818d530a8740
protocol_name	https
endpoint	https://127.0.0.1:443/https
status	DELIVERED
http_code	200
create_time	2022-11-01T00:00:00Z
send_time	2022-11-01T00:00:10Z

6 Permissions Management

6.1 Creating a User and Granting SMN Permissions

Use **IAM** to implement fine-grained permissions control over your SMN resources. With IAM, you can:

- Create IAM users for employees based on your enterprise's organizational structure. Each IAM user will have their own security credentials for accessing SMN resources.
- Grant only the permissions required for users to perform a specific task.
- Entrust a Huawei Cloud account or cloud service to perform efficient O&M on your SMN resources.

If your Huawei Cloud account does not require individual IAM users, skip this chapter.

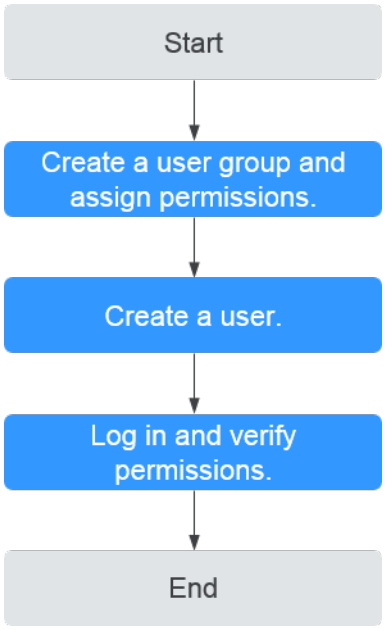
This section describes the procedure for granting permissions (see **Figure 6-1**).

Prerequisites

Learn about the system permissions (see **Permissions Management**) supported by SMN and choose policies or roles according to your requirements. For system permissions of other cloud services, see **System Permissions**.

Process Flow

Figure 6-1 Process for granting SMN permissions



1. **Create a user group and assign permissions**.

   Create a user group on the IAM console and assign the **SMN Administrator** permissions to the group.

2. **Create a user and add it to the user group**

   Create a user on the IAM console and add the user to the group created in **1**.

3. **Log in as the created user** and verify the **SMN Administrator** permissions.

   Log in to the SMN console by using the created user, and verify that the user only has read permissions for SMN.

   - Choose **Service List** > **Simple Message Notification**. On the SMN console, choose **Topic Management** > **Topics**, and click **Create Topic** in the upper right corner. If the topic is successfully created, the **SMN Administrator** permissions have already taken effect.
   - Choose any other service in **Service List**. If a message appears indicating that you have insufficient permissions to access the service, the **SMN Administrator** permissions have already taken effect.

6.2 SMN Custom Policies

You can create custom policies to supplement the system-defined policies of SMN. For the actions supported by custom policies, see **Permissions Policies and Supported Actions** in *Simple Message Notification API Reference*.

You can create custom policies in either of the following ways:

- Visual editor: Select cloud services, actions, resources, and request conditions. This does not require knowledge of policy syntax.

- JSON: Create a JSON policy or edit an existing one.

The following are examples of custom policies created for SMN. For details, see **Creating a Custom Policy**.

Example SMN Custom Policies

- Example 1: allowing topic creation

```
{
   "Version": "1.1",
   "Statement": [
      {
         "Effect": "Allow",
         "Action": [
            "smn:topic:create"
         ]
      }
   ]
}
```

- Example 2: denying topic deletion

  A policy with only **Deny** permissions must work with other policies. If the policies assigned to a user contain both **Allow** and **Deny** actions, the **Deny** actions take precedence over the **Allow** actions.

  You can assign a system policy of **SMN FullAccess** and a custom policy of denying topic deletion to the user group which the user belongs to at the same time. Thus the user can perform all operations on SMN except deleting topics. The example is as follows.

```
{
   "Version": "1.1",
   "Statement": [
      {
         "Effect": "Deny",
         "Action": [
            "smn:topic:delete"
         ]
      }
   ]
}
```

- Example 3: defining multiple actions in a policy

  A custom policy can contain multiple actions that belong to any global or project-level services. The example is as follows.

```
{
   "Version": "1.1",
   "Statement": [
      {
         "Effect": "Allow",
         "Action": [
            "smn:topic:create",
            "smn:tag:create",
            "smn:application:create"
         ]
      },
      {
         "Effect": "Allow",
         "Action": [
            "elb:certificates:create",
            "elb:whitelists:create",
            "elb:pools:create",
            "elb:members:create",
            "elb:healthmonitors:create",
            "elb:l7policies:create",
            "elb:listeners:create",
```



```
            "elb:loadbalancers:create"
        ]
      }
   ]
}
```


7 Quotas

What Is Quota?

Quotas can limit the number or amount of resources available to users, such as the maximum number of ECSs or EVS disks that can be created.

If the existing resource quota cannot meet your service requirements, you can apply for a higher quota.

How Do I View My Quotas?

1. Log in to the management console.
2. Click  in the upper left corner and select the desired region and project.
3. In the upper right corner of the page, choose **Resources** > **My Quotas**.

   The **Service Quota** page is displayed.
4. View the used and total quota of each type of resources on the displayed page.

   If a quota cannot meet service requirements, apply for a higher quota.

How Do I Apply for a Higher Quota?

1. Log in to the management console.
2. In the upper right corner of the page, choose **Resources** > **My Quotas**.

   The **Service Quota** page is displayed.
3. Click **Increase Quota** in the upper right corner of the page.
4. On the **Create Service Ticket** page, configure parameters as required.

   In the **Problem Description** area, fill in the content and reason for adjustment.
5. After all necessary parameters are configured, select **I have read and agree to the Ticket Service Protocol and Privacy Statement.** and click **Submit**.

A Appendix

A.1 JSON Message Format

Description

The JSON format allows you to specify different message content for different subscription protocols, including **Default**, **SMS**, **HTTP**, **HTTPS**, and **Email**. The message content you specify will be sent to subscription endpoints using applicable protocols.

```
{
  "default": "Dear Sir or Madam, this is a default message.",
  "email": "Dear Sir or Madam, this is an email message.",
  "http": "{'message':'Dear Sir or Madam, this is an HTTP message.'}",
  "https": "{'message':'Dear Sir or Madam, this is an HTTPS message.'}"
  "sms": "This is an SMS message."



   }
```

It is recommended that you specify general message content for all subscription types in the **Default** protocol and enter customized content for specific protocols.

In the following example, you enter a shorter message for the SMS protocol because of the length limit on SMS messages. SMS subscribers in the topic receive the message "This is an SMS message.", while other types of subscribers (email, HTTP, and HTTPS) receive the one "Dear Sir or Madam, this is a default message."

```
{
  "sms": "This is an SMS message.",
  "default": "Dear Sir or Madam, this is a default message."
 }
```

Constraints

- The content must be in JSON format.
- You must configure the **Default** protocol in the JSON message.
- The size of a JSON message cannot exceed 256 KB.

Calculation on the Size of a JSON Message

The size of a JSON message, including braces, quotation marks, spaces, line breaks, protocols, and message content, cannot exceed 256 KB. The size of a JSON message generated for each protocol may vary.

For example, message content "This is a default message." contains 26 bytes.

The system automatically adds the **Default** protocol when generating a JSON message.

```
{
  "default": "This is a default message.",
  "protocol1": "This is a default message.",
  "protocol2": "This is a default message.",
  ...
}
```

The total number of protocols is *N*, including the **Default** protocol and those you selected.

The size of the message is calculated as follows:

- Three spaces in each of the *N* protocols: 3 x *N* = 3*N* bytes
- Four quotation marks in each of the *N* protocols: 4 x *N* = 4*N* bytes
- One colon in each of the *N* protocols: 1 x *N* = *N* bytes
- Message content "This is a default message." in each of the *N* protocols: 26 x *N* = 26*N* bytes
- Commas in (*N* – 1) protocols: 1 x (*N* – 1) = (*N* – 1) bytes
- Line breaks in (*N* + 1) protocols: 1 x (*N* + 1) = (*N* + 1) bytes
- Two braces: 2 bytes
- Protocol name **Default**: 7 bytes

Bytes of protocols you selected:

- **HTTP**: 4 bytes
- **HTTPS**: 5 bytes
- **Email**: 5 bytes
- **SMS**: 3 bytes

Total size = 36*N* + 9 + Bytes of protocols you selected

For example, you selected the HTTP, HTTPS, and email protocols, and the message is as follows:

```
{
  "default": "This is a default message.",
  "email": "This is a default message.",
  "http": "This is a default message.",
  "https": "This is a default message."
}
```

The system adds a **Default** protocol, and the value of *N* is 4. The size of this JSON message is:

- Fixed length: 36 x 4 + 9 = 153 bytes
- **http**: 4 bytes

- **https**: 5 bytes
- **email**: 5 bytes

The total size is 167 bytes (153 + 4 + 5 + 5 = 167).

A.2 Template Message Format

Message templates are used to publish messages with fixed content and use variables as placeholders to represent content that you can change.

The size of template message cannot exceed 256 KB. The following is an example of how to format a template when you manually type the template message content:

```
"message_template_name":"confirm_message",
"tags":{
   "topic_urn":"urn:smn:regionId:xxxx:SMN_01"
    }
```

Table A-1 Parameters description and setting

Parameter	Description
message_template_name	Specifies the template name, which must be specified. You can query the template name in the template list. You must create a template of the default protocol so that SMN can send messages using the default template once it fails to match a specified protocol.
tags	Variables in the template, which are presented as JSON mappings. You can create templates for different protocols using the same template name and configure different variables in each template.

A.3 Messages Using Different Protocols

Message contents delivered to endpoints using different protocols differ.

- Email or HTTP/HTTPS endpoints will receive the message subject, content, and a link to unsubscribe.
- SMS endpoints receive only the message content.

A.4 Traffic Control over Subscription Confirmation

To prevent malicious users from harassing subscribers, SMN limits the number of subscription confirmation messages a user can send to an individual subscriber within a specified period of time. Traffic control policies apply to confirmation requests issued both from the SMN console and by API calling. Traffic control policies for different subscription protocols are as follows:

- Email: A user can send up to 20 confirmation messages within one hour or 40 within two days. When the threshold is met, SMN will not send any more confirmation messages to that email address in the next seven days. After the subscriber confirms the subscription, SMN clears the count in the traffic control policy.
- SMS: A user can send up to 10 confirmation messages within one hour or 20 within two days. If the user exceeds this threshold, SMN will not send any more confirmation messages to that phone number in the next seven days. After the subscriber confirms the subscription, SMN clears the count in the traffic control policy.
- HTTP or HTTPS: A user can send up to 200 confirmation messages within 10 minutes.

A.5 Country or Region Codes

Country or Region	Chinese	Code
Afghanistan	阿富汗	93
Albania	阿尔巴尼亚	355
Algeria	阿尔及利亚	213
American Samoa	美属萨摩亚	1684
Andorra	安道尔	376
Angola	安哥拉	244
Anguilla	安圭拉	1264
Antigua and Barbuda	安提瓜和巴布达	1268
Argentina	阿根廷	54
Armenia	亚美尼亚	374
Aruba	阿鲁巴	297
Australia	澳大利亚	61
Austria	奥地利	43
Azerbaijan	阿塞拜疆	994
Bahamas	巴哈马	1242
Bahrain	巴林	973
Bangladesh	孟加拉国	880
Barbados	巴巴多斯	1246
Belarus	白俄罗斯	375
Belgium	比利时	32

Country or Region	Chinese	Code
Belize	伯利兹	501
Benin	贝宁	229
Bermuda	百慕大群岛	1441
Bhutan	不丹	975
Bolivia	玻利维亚	591
Bosnia and Herzegovina	波斯尼亚和黑塞哥维那	387
Botswana	博茨瓦纳	267
Brazil	巴西	55
British Indian Ocean Territory	英属印度洋领地	246
Brunei	文莱	673
Bulgaria	保加利亚	359
Burkina Faso	布基纳法索	226
Burundi	布隆迪	257
Cambodia	柬埔寨	855
Cameroon	喀麦隆	237
Canada	加拿大	1
Cape Verde	佛得角	238
Cayman Islands	开曼群岛	1345
Central African Republic	中非共和国	236
Chad	乍得	235
Chile	智利	56
Colombia	哥伦比亚	57
Comoros	科摩罗	269
Republic Of The Congo	刚果共和国	242
Democratic Republic of the Congo	刚果民主共和国	243
Cook Islands	库克群岛	682
Costa Rica	哥斯达黎加	506

Country or Region	Chinese	Code
Croatia	克罗地亚	385
Curacao	库拉索	599
Cyprus	塞浦路斯	357
Czechia	捷克	420
Denmark	丹麦	45
Djibouti	吉布提	253
Dominica	多米尼克	1767
Dominican Republic	多米尼加共和国	1809
Ecuador	厄瓜多尔	593
Egypt	埃及	20
El Salvador	萨尔瓦多	503
Equatorial Guinea	赤道几内亚	240
Eritrea	厄立特里亚	291
Estonia	爱沙尼亚	372
Eswatini	斯威士兰	268
Ethiopia	埃塞俄比亚	251
Falkland Islands	福克兰群岛	500
Faroe Islands	法罗群岛	298
Fiji	斐济	679
Finland	芬兰	358
France	法国	33
French Guiana	法属圭亚那	594
French Polynesia	法属波利尼西亚	689
Gabon	加蓬	241
Gambia	冈比亚	220
Georgia	格鲁吉亚	995
Germany	德国	49
Ghana	加纳	233
Gibraltar	直布罗陀	350
Greece	希腊	30

Country or Region	Chinese	Code
Greenland	格陵兰岛	299
Grenada	格林纳达	1473
Guadeloupe	瓜德罗普岛	590
Guatemala	瓜地马拉	502
Guernsey	根西	44
Guinea	几内亚	224
Guinea-Bissau	几内亚比绍	245
Guyana	圭亚那	592
Haiti	海地	509
Honduras	洪都拉斯	504
Hong Kong (China)	中国香港	852
Hungary	匈牙利	36
Iceland	冰岛	354
India	印度	91
Indonesia	印度尼西亚	62
Iraq	伊拉克	964
Ireland	爱尔兰	353
Isle of Man	马恩岛	44
Israel	以色列	972
Italy	意大利	39
Ivory Coast	科特迪瓦	225
Jamaica	牙买加	1876
Japan	日本	81
Jersey	泽西岛	44
Jordan	约旦	962
Kazakhstan	哈萨克斯坦	7
Kenya	肯尼亚	254
Kuwait	科威特	965
Kyrgyzstan	吉尔吉斯斯坦	996
Laos	老挝	856

Country or Region	Chinese	Code
Latvia	拉脱维亚	371
Lebanon	黎巴嫩	961
Lesotho	莱索托	266
Liberia	利比里亚	231
Libya	利比亚	218
Liechtenstein	列支敦士登	423
Lithuania	立陶宛	370
Luxembourg	卢森堡	352
Macao (China)	中国澳门	853
North Macedonia	北马其顿	389
Madagascar	马达加斯加	261
Malawi	马拉维	265
Malaysia	马来西亚	60
Maldives	马尔代夫	960
Mali	马里	223
Malta	马耳他	356
Marshall Islands	马绍尔群岛	692
Martinique	马提尼克	596
Mauritania	毛里塔尼亚	222
Mauritius	毛里求斯	230
Mexico	墨西哥	52
Moldova	摩尔多瓦	373
Monaco	摩纳哥	377
Mongolia	蒙古	976
Montenegro	黑山	382
Montserrat	蒙特塞拉特岛	1664
Morocco	摩洛哥	212
Mozambique	莫桑比克	258
Myanmar	缅甸	95
Namibia	纳米比亚	264

Country or Region	Chinese	Code
Nauru	瑙鲁	674
Nepal	尼泊尔	977
Netherlands	荷兰	31
New Caledonia	新喀里多尼亚	687
New Zealand	新西兰	64
Nicaragua	尼加拉瓜	505
Niger	尼日尔	227
Nigeria	尼日利亚	234
Niue	纽埃岛	683
Norfolk Island	诺福克岛	672
North Cyprus	北塞浦路斯	90
Norway	挪威	47
Oman	阿曼	968
Pakistan	巴基斯坦	92
Palau	帕劳	680
Palestinian Territory	巴勒斯坦	970
Panama	巴拿马	507
Papua New Guinea	巴布亚新几内亚	675
Paraguay	巴拉圭	595
Peru	秘鲁	51
Philippines	菲律宾	63
Poland	波兰	48
Portugal	葡萄牙	351
Qatar	卡塔尔	974
Réunion Island	留尼汪	262
Romania	罗马尼亚	40
Rwanda	卢旺达	250
Saint Kitts and Nevis	圣基茨和尼维斯	1869
Saint Lucia	圣卢西亚	1758

Country or Region	Chinese	Code
Saint Pierre and Miquelon	圣皮埃尔和密克隆	508
Saint Vincent and The Grenadines	圣文森特和格林纳丁斯	1784
Samoa	萨摩亚	685
San Marino	圣马力诺	378
Sao Tome and Principe	圣多美和普林西比	239
Saudi Arabia	沙特阿拉伯	966
Senegal	塞内加尔	221
Serbia	塞尔维亚	381
Seychelles	塞舌尔	248
Sierra Leone	塞拉利昂	232
Singapore	新加坡	65
Sint Maarten (Dutch Part)	荷属圣马丁	1721
Slovakia	斯洛伐克	421
Slovenia	斯洛文尼亚	386
Solomon Islands	所罗门群岛	677
Somalia	索马里	252
South Africa	南非	27
South Korea	韩国	82
Spain	西班牙	34
Sri Lanka	斯里兰卡	94
Suriname	苏里南	597
Sweden	瑞典	46
Switzerland	瑞士	41
Taiwan (China)	中国台湾	886
Tajikistan	塔吉克斯坦	992
Tanzania	坦桑尼亚	255
Thailand	泰国	66
Timor L'este	东帝汶	670

Country or Region	Chinese	Code
Togo	多哥	228
Tonga	汤加	676
Trinidad and Tobago	特立尼达和多巴哥	1868
Tunisia	突尼斯	216
Türkiye	土耳其	90
Turkmenistan	土库曼斯坦	993
Turks and Caicos Islands	特克斯和凯科斯群岛	1649
Tuvalu	图瓦卢	688
Uganda	乌干达	256
Ukraine	乌克兰	380
United Arab Emirates	阿拉伯联合酋长国	971
United Kingdom	英国	44
United States	美国	1
Uruguay	乌拉圭	598
Uzbekistan	乌兹别克斯坦	998
Vanuatu	瓦努阿图	678
Venezuela	委内瑞拉	58
Vietnam	越南	84
Virgin Islands, British	英属维尔京群岛	1284
Wallis and Futuna	瓦利斯和富图纳	681
Yemen	也门	967
Zambia	赞比亚	260
Zimbabwe	津巴布韦	263

A.6 Mappings Between SMN Actions and APIs

Table A-2 Mappings between SMN actions and APIs

Action	API	Function
SMN:UpdateTopic	UpdateTopic	Modify the topic. Only the **display_name** value can be changed.
SMN:DeleteTopic	DeleteTopic	Delete a topic and its subscribers. If a topic is deleted, any pending messages may fail to send to the topic subscribers.
SMN:QueryTopicDetail	QueryTopicDetail	Query details about a topic.
SMN:ListTopicAttributes	ListTopicAttributes	Query topic attributes.
SMN:UpdateTopicAttribute	UpdateTopicAttribute	Modify an attribute of a topic.
SMN:DeleteTopicAttributes	DeleteTopicAttributes	Delete all attributes of a topic.
SMN:DeleteTopicAttributeByName	DeleteTopicAttributeByName	Delete an attribute of a specified topic.
SMN:ListSubscriptionsByTopic	ListSubscriptionsByTopic	Query the subscription list of a specified topic by page. The list is sorted by time when the subscriptions are added in ascending order. You can specify values of **offset** and **limit**. If no subscription has been added, an empty list is returned.
SMN:Subscribe	Subscribe	Add a subscription to a specified topic and send a confirmation message to the subscriber. After confirming the subscription, the subscriber can receive notification messages published to the topic.
SMN:Unsubscribe	Unsubscribe	Delete a subscription. This operation requires identity authentication. Only the subscriber or the topic owner can delete a subscription.

Action	API	Function
SMN:Publish	Publish	Publish messages to a topic. After a message ID is returned, the message has been saved and is to be delivered to subscribers of the topic. The message form varies depending on the protocol of each subscription.

A.7 Restrictions on SMS Messaging

- Messaging time requirements are as follows:
  - Except for user-triggered messaging, your enterprise should not send SMS messages beyond 07:00 to 21:00.
  - If you do need to send urgent events from 21:00 to 07:00, SMN allows you to send 100 messages containing the same content at most.
- Maximum number of messages sent to each phone number

  You can send a maximum of 10 messages to each phone number within a day. If you need to send more, provide a phone number whitelist.
- Message length

A.8 HTTP/HTTPS Messages

A.8.1 Introduction

HTTP/HTTPS messages can be classified as management messages and service messages. The former includes subscription messages and subscription cancellation messages, while the latter includes notification messages. An HTTPS/HTTPS message is composed of a message header and body, which are illustrated in detail in this topic.

A.8.2 HTTP or HTTPS Message Format

Scenarios

CAUTION

When receiving HTTP or HTTPS messages sent by SMN, refer to the industry standards for the CN name of the terminal certificate. Some special characters may cause HTTPS message sending failures.

This section describes the format of messages sent to HTTP or HTTPS endpoints. You can identify messages based on message types in the headers. HTTP/HTTPS message types include: subscription confirmation messages, notification messages, and subscription cancellation messages. POST is used for HTTP/HTTPS messages.

The header of an SMN HTTP/HTTPS message contains the following parameters: **X-SMN-MESSAGE-TYPE**, **X-SMN-MESSAGE-ID**, **X-SMN-TOPIC-URN**, and **X-SMN-SUBSCRIPTION-URN**.

Table A-3 HTTP/HTTPS header parameters

Parameter	Description
X-SMN-MESSAGE-TYPE	Indicates the message type, which can be: • SubscriptionConfirmation • Notification • UnsubscribeConfirmation
X-SMN-MESSAGE-ID	Indicates the unique message ID.
X-SMN-TOPIC-URN	Indicates the URN of the topic the message belongs to.
X-SMN-SUBSCRIPTION-URN	Identifies the subscription endpoint. This parameter is required only when messages are pushed over HTTP/HTTPS and when you cancel your HTTP/HTTPS subscriptions.

HTTP/HTTPS Subscription Confirmation Message Format

After you add an HTTP/HTTPS endpoint, SMN sends a subscription confirmation message to the subscriber. The message body is composed of JSON character strings. The subscriber must obtain the subscription URL (**subscribe_url**) to confirm the subscription. **Table A-4** describes the JSON field in detail.

Table A-4 HTTP/HTTPS subscription confirmation message body

Parameter	Description
type	Indicates the message type. Its value is **SubscriptionConfirmation**.
signature	Indicates the signature information. The signature includes the **message**, **message_id**, **subscribe_url**, **timestamp**, **topic_urn**, and **type** fields. For details about signature verification, see **Message Signature Verification**.
topic_urn	Indicates the URN of the topic the message belongs to.
message_id	Indicates the unique message ID.
signature_version	Indicates the signature version, which is **V1**.
message	Indicates the message content.

Parameter	Description
subscribe_url	Indicates the URL to be accessed for subscription confirmation.
signing_cert_url	Indicates the certificate URL for a message signature. It can be directly accessed without authentication.
timestamp	Indicates the time stamp when the message was initially sent.

The following is an example HTTP/HTTPS subscription confirmation message:


```
{
    "signature": "ViE96uGbBkl
+S8eWqgebi5KdmRht2U8+Rs88yuyMHq1k4h3jUkcDZ6HCqTqdpJ8nrLcdqETyyEiOQyTszJdU05z
+LhfE8jerCCdSbL4zeInVkydHh0kcCRWmORye0/EuQ/gLC1UIXwvUsqbUCPnBRhNFXOeXMOPPCzK
+d04xjy4QHd1H/bHxgsY3AlTe0gCFT068Zru7OK6w9aQaY44mXnN3OWGmBmoHyFab5TRXLSQNz/9u/
Vj646cQMMaI0PPQ30QzGYD0MtzgDZi12m8jMTHAnMkTEcbLaEgaqmaoEnATSpEcspFKNXv2skwk7rsVakMOI
SpMH3+qC6RzhETA2A==",
    "topic_urn": "urn:smn:region01:0553db98c800d5192f9bc01232b89622:vpc_status_report_topic",
    "message_id": "d86c201092574e71a3ca85826652c58b",
    "signature_version": "v1",
    "type": "SubscriptionConfirmation",
    "message": "{\"enterpriseProjectId\": \"0\", \"eventTime\": \"2019-08-12 22:40:55.040632\",
\"chargingMode\": \"postPaid\", \"cloudserviceType\": \"xxx.service.type.bandwidth\", \"eventType\": 1,
\"regionId\": \"region01\", \"tenantId\": \"057eefe55400d2742f8cc0017870ceef\", \"resourceType\":
\"xxx.resource.type.bandwidth\", \"resourceSpecCode\": \"19_bgp\", \"resourceSize\": 10, \"resourceId\":
\"e091f1b1-08ef-4e2b-a27e-f85e4c19026a\", \"resouceSizeMeasureId\": 15, \"resourceName\":
\"elbauto_2019_08_13_06_40_46\"}",
    "subscribe_url": "https://console.xxx.com/smn/subscription/unsubscribe?
subscription_urn=urn:smn:region01:0553db98c800d5192f9bc01232b89622:vpc_status_report_topic:
653e212a43884f7188ca656c537e31ce",
    "signing_cert_url": "https://smn.cn-north-9.myhuaweicloud.com/smn/SMN_cn-
north-9_94f60ccdfbee4588aa4d555935a56ba3.pem",
    "timestamp": "2019-08-12T22:40:56Z"
}
```


HTTP/HTTPS Notification Message Format

After an HTTP/HTTPS subscriber confirms the subscription, the subscriber can receive notification messages published to the topic. The notification message body is composed of JSON character strings, which are described in **Table A-5**.

Table A-5 HTTP/HTTPS notification message body

Parameter	Description
type	Indicates the message type. Its value is **Notification**.
signature	Indicates the signature information. The signature includes the **message**, **message_id**, **subject**, **timestamp**, **topic_urn**, and **type** fields. If the **subject** field is empty, the signature is not verified. For details about signature verification, see **Message Signature Verification**.
subject	Indicates the message subject.

Parameter	Description
topic_urn	Indicates the URN of the topic the message belongs to.
message_id	Indicates the unique message ID.
signature_version	Indicates the signature version, which is **V1**.
message	Indicates the message content.
unsubscribe_url	Indicates the URL for canceling a subscription.
signing_cert_url	Indicates the certificate URL for generating the message signature.
timestamp	Indicates the time stamp when the message was initially sent.

The following is an example HTTP(S) notification message:

```
{
    "signature": "ViE96uGbBkl
+S8eWqgebi5KdmRht2U8+Rs88yuyMHq1k4h3jUkcDZ6HCqTqdpJ8nrLcdqETyyEiOQyTszJdU05z
+LhfE8jerCCdSbL4zeInVkydHh0kcCRWmORye0/EuQ/gLC1UIXwvUsqbUCPnBRhNFXOeXMOPPCzK
+d04xjy4QHd1H/bHxgsY3AlTe0gCFT068Zru7OK6w9aQaY44mXnN3OWGmBmoHyFab5TRXLSQNz/9u/
Vj646cQMMaI0PPQ30QzGYD0MtzgDZi12m8jMTHAnMkTEcbLaEgaqmaoEnATSpEcspFKNXv2skwk7rsVakMOI
SpMH3+qC6RzhETA2A==",
    "topic_urn": "urn:smn:region01:0553db98c800d5192f9bc01232b89622:vpc_status_report_topic",
    "message_id": "d86c201092574e71a3ca85826652c58b",
    "signature_version": "v1",
    "type": "Notification",
    "message": "{\"enterpriseProjectId\": \"0\", \"eventTime\": \"2019-08-12 22:40:55.040632\",
\"chargingMode\": \"postPaid\", \"cloudserviceType\": \"xxx.service.type.bandwidth\", \"eventType\": 1,
\"regionId\": \"region01\", \"tenantId\": \"057eefe55400d2742f8cc0017870ceef\", \"resourceType\":
\"xxx.resource.type.bandwidth\", \"resourceSpecCode\": \"19_bgp\", \"resourceSize\": 10, \"resourceId\":
\"e091f1b1-08ef-4e2b-a27e-f85e4c19026a\", \"resouceSizeMeasureId\": 15, \"resourceName\":
\"elbauto_2019_08_13_06_40_46\"}",
    "unsubscribe_url": "https://console.xxx.com/smn/subscription/unsubscribe?
subscription_urn=urn:smn:region01:0553db98c800d5192f9bc01232b89622:vpc_status_report_topic:
653e212a43884f7188ca656c537e31ce",
    "signing_cert_url": "https://smn.cn-north-9.myhuaweicloud.com/smn/SMN_cn-
north-9_94f60ccdfbee4588aa4d555935a56ba3.pem",
    "timestamp": "2019-08-12T22:40:56Z"
}
```

HTTP/HTTPS Subscription Cancellation Message Format

After an HTTP/HTTPS subscription is canceled, the subscriber receives a subscription cancellation message sent by SMN. The message body is composed of JSON character strings, which are described in **Table A-6**.

Table A-6 HTTP/HTTPS subscription cancellation message body

Parameter	Description
type	Indicates the message type. Its value is **UnsubscribeConfirmation**.

Parameter	Description
signature	Indicates the signature information. The signature includes the **message**, **message_id**, **subscribe_url**, **timestamp**, **topic_urn**, and **type** fields. For details about signature verification, see **Message Signature Verification**.
topic_urn	Indicates the URN of the topic the message belongs to.
message_id	Indicates the unique message ID.
signature_version	Indicates the signature version, which is **V1**.
message	Indicates the message content.
subscribe_url	Indicates the URL for a re-subscription.
signing_cert_url	Indicates the certificate URL for generating the message signature.
timestamp	Indicates the time stamp when the message was initially sent.

The following is an example HTTP(S) message for canceling a subscription:

```
{
    "signature": "ViE96uGbBkl
+S8eWqgebi5KdmRht2U8+Rs88yuyMHq1k4h3jUkcDZ6HCqTqdpJ8nrLcdqETyyEiOQyTszJdU05z
+LhfE8jerCCdSbL4zeInVkydHh0kcCRWmORye0/EuQ/gLC1UIXwvUsqbUCPnBRhNFXOeXMOPPCzK
+d04xjy4QHd1H/bHxgsY3AlTe0gCFT068Zru7OK6w9aQaY44mXnN3OWGmBmoHyFab5TRXLSQNz/9u/
Vj646cQMMaI0PPQ30QzGYD0MtzgDZi12m8jMTHAnMkTEcbLaEgaqmaoEnATSpEcspFKNXv2skwk7rsVakMOI
SpMH3+qC6RzhETA2A==",
    "topic_urn": "urn:smn:region01:0553db98c800d5192f9bc01232b89622:vpc_status_report_topic",
    "message_id": "d86c201092574e71a3ca85826652c58b",
    "signature_version": "v1",
    "type": "UnsubscribeConfirmation",
    "message": "{\"enterpriseProjectId\": \"0\", \"eventTime\": \"2019-08-12 22:40:55.040632\",
\"chargingMode\": \"postPaid\", \"cloudserviceType\": \"xxx.service.type.bandwidth\", \"eventType\": 1,
\"regionId\": \"region01\", \"tenantId\": \"057eefe55400d2742f8cc0017870ceef\", \"resourceType\":
\"xxx.resource.type.bandwidth\", \"resourceSpecCode\": \"19_bgp\", \"resourceSize\": 10, \"resourceId\":
\"e091f1b1-08ef-4e2b-a27e-f85e4c19026a\", \"resouceSizeMeasureId\": 15, \"resourceName\":
\"elbauto_2019_08_13_06_40_46\"}",
    "subscribe_url": "https://console.xxx.com/smn/subscription/unsubscribe?
subscription_urn=urn:smn:region01:0553db98c800d5192f9bc01232b89622:vpc_status_report_topic:
653e212a43884f7188ca656c537e31ce",
    "signing_cert_url": "https://smn.cn-north-9.myhuaweicloud.com/smn/SMN_cn-
north-9_94f60ccdfbee4588aa4d555935a56ba3.pem",
    "timestamp": "2019-08-12T22:40:56Z"
}
```

A.8.3 Message Signature Verification

Scenarios

To ensure message security, SMN provides signature authentication for HTTP/HTTPS subscription confirmation messages, subscription cancellation messages, and notification messages. After you receive HTTP/HTTPS messages, check them based on the signatures.

Procedure

After receiving an HTTP/HTTPS message, check it with the following procedure:

1. Verify the key-value pairs (which vary depending on the message type) contained in the message signature. For details, see **Signature Strings for Different Message Types**.
2. Download the X509 certificate from the certificate URL (**signing_cert_url**) contained in the message.

   NOTE

   The request to download the certificate is always sent over HTTPS. When you download a certificate, verify the identity of the certificate server.
3. Extract the public key from the X509 certificate for verifying the message reliability and integrity.
4. Determine which method will be used to verify the signature based on the message type (the **type** field in the message).
5. Create signature strings. Obtain the signature parameters from the message and sort them in alphabetical order. Each parameter occupies a line, with its value following in the next line.

Signature Strings for Different Message Types

1. Notification messages
   - A notification message signature must contain the following parameters (If the value of **subject** is empty, do not include it in the signature):

     ```
     message
     message_id
     subject
     timestamp
     topic_urn
     type
     ```
   - For example, the signature information for a notification message is as follows:

     ```
     message
     My test message
     message_id
     88c726942175432bac921eafd0036163
     subject
     demo
     timestamp
     2016-08-15T07:29:16Z
     topic_urn
     urn:smn:regionId:74dc9e44d0cc4573adfce91cdfdd3ba9:xxxx
     type
     Notification
     ```

     NOTE

     Each parameter occupies a line, with its value following in the next line.
2. Subscription confirmation and subscription cancellation messages
   - A subscription confirmation or subscription cancellation message signature must contain the following parameters:

     ```
     message
     message_id
     subscribe_url
     timestamp
     ```



```
topic_urn
type
```

- For example, the signature information for a subscription confirmation message is as follows:

```
message
You are invited to subscribe to topic:
urn:smn:regionId:d91989905b8449b896f3a4f0ad57222d:demo. To confirm this subscription,
please visit the following SubscribeURL in this message.
message_id
def5c309cbff44d5a870787ed937edf8
subscribe_url
https://IP address/smn/subscription/confirm?Region ID&Token&Topic URN:demo
timestamp
2016-08-15T07:29:16Z
topic_urn
urn:smn:regionId:d91989905b8449b896f3a4f0ad57222d:demo
type
SubscriptionConfirmation
```

NOTE

Each parameter occupies a line, with its value following in the next line.

A.8.4 Sample Code

Java

Verify **signing_cert_url**, **signature** that obtained in **HTTP or HTTPS Message Format**, and **message** (contained in the message signature) to check the message validity, as shown in the following:

```
private static void isMessageValid(String signing_cert_url,
        String signature, Map<String, String> message) {
    InputStream in = null;
    try {
        URL url = new URL(signing_cert_url);
        in = url.openStream();
        CertificateFactory cf = CertificateFactory.getInstance("X.509");
        X509Certificate cert = (X509Certificate) cf.generateCertificate(in);
        Signature sig = Signature.getInstance(cert.getSigAlgName());
        sig.initVerify(cert.getPublicKey());
        sig.update(buildSignMessage(message).getBytes("UTF-8"));
        byte[] sigByte = Base64.getDecoder().decode(signature);
        if (sig.verify(sigByte)) {
            System.out.println("Verify success");
        } else {
            System.out.println("Verify failed");
        }
    } catch (Exception e) {
        throw new SecurityException("Verify method failed.", e);
    } finally {
        if (in != null) {
            try {
                in.close();
            } catch (IOException e) {
                e.printStackTrace();
            }
        }
    }
}
```


NOTE

If your Java version is earlier than 8, use the third-party package **commons-codec.jar** to perform Base64 decoding, and replace **byte[] sigByte = Base64.getDecoder().decode(signature);** with **byte[] sigByte = Base64.decodeBase64(signature);** in the preceding code.

The following is an example of the code to create the message verification signature:

```
private static String buildSignMessage(Map<String,String> msg) {
    String type = msg.get("type");
    String message = null;
    if ("Notification".equals(type)){
        message = buildNotificationMessage(msg);
    } else if ("SubscriptionConfirmation".equals(type) ||
    "UnsubscribeConfirmation".equals(type)){
        message = buildSubscriptionMessage(msg);
    }
    return message;
}

private static String buildSubscriptionMessage(Map<String, String> msg) {
    String stringMessage = "message\n";
    stringMessage += msg.get("message") + "\n";
    stringMessage += "message_id\n";
    stringMessage += msg.get("message_id") + "\n";
    stringMessage += "subscribe_url\n";
    stringMessage += msg.get("subscribe_url") + "\n";
    stringMessage += "timestamp\n";
    stringMessage += msg.get("timestamp") + "\n";
    stringMessage += "topic_urn\n";
    stringMessage += msg.get("topic_urn") + "\n";
    stringMessage += "type\n";
    stringMessage += msg.get("type") + "\n";
    return stringMessage;
}

private static String buildNotificationMessage(Map<String, String> msg)
    {
        String stringMessage = "message\n";
        stringMessage += msg.get("message").toString() + "\n";
        stringMessage += "message_id\n";
        stringMessage += msg.get("message_id").toString() + "\n";
        if (msg.get("subject") != null){
            stringMessage += "subject\n";
            stringMessage += msg.get("subject").toString() + "\n";
        }
        stringMessage += "timestamp\n";
        stringMessage += msg.get("timestamp").toString() + "\n";
        stringMessage += "topic_urn\n";
        stringMessage += msg.get("topic_urn").toString() + "\n";
        stringMessage += "type\n";
        stringMessage += msg.get("type").toString() + "\n";
        return stringMessage;
    }
```

Node.js

```
const fs = require('fs');
const crypto = require('crypto');
const jsrsag = require('jsrsasign');

/**
* Message signature verification
* @param pemFile: path for storing the signature file (path for storing the certificate downloaded to your local computer)
* @param signature: signature to be verified
```



```
* @param message: content of the message to be verified
* @returns {boolean} true: The signature passes the verification. false: The signature fails the verification.
 */
function verifyMessage(pemFile, signature, message) {
    const pubPem = fs.readFileSync(pemFile);
    const verify = crypto.createVerify(signatureAlgorithm(pubPem));
    verify.update(buildSignMessage(message));
    const verifyResult = verify.verify(pubPem, signature, 'base64');
    if (verifyResult) {
        console.log("verify success");
        return true;
    } else {
        console.log('verify failed, result: ' + verifyResult);
        return false;
    }
}

/**
* Obtain the signature algorithm from the certificate.
 */
function signatureAlgorithm(pubPem) {
    const certObject = new jsrsag.X509();
    certObject.readCertPEM(pubPem.toString());
    let algorithm = certObject.getSignatureAlgorithmField();
    if (algorithm.split('with').length > 1) {
        algorithm = algorithm.split('with')[1] + '-' + algorithm.split('with')[0];
    }
    return algorithm;
}

function buildSignMessage(msg) {
    const type = msg.type;
    let message = '';
    if (type === 'Notification') {
        message = buildNotificationMessage(msg);
    } else if (type === 'SubscriptionConfirmation') {
        message = buildSubscriptionMessage(msg);
    }
    return message;
}

function buildNotificationMessage(msg) {
    let signMessage = 'message\n' + msg.message + '\n';
    signMessage += 'message_id\n' + msg.message_id + '\n';
    if (msg.subject) {
        signMessage += 'subject\n' + msg.subject + '\n';
    }
    signMessage += 'timestamp\n' + msg.timestamp + '\n';
    signMessage += 'topic_urn\n' + msg.topic_urn + '\n';
    signMessage += 'type\n' + msg.type + '\n';
    return signMessage;
}

function buildSubscriptionMessage(msg) {
    let signMessage = 'message\n' + msg.message + '\n';
    signMessage += 'message_id\n' + msg.message_id + '\n';
    signMessage += 'subscribe_url\n' + msg.subscribe_url + '\n';
    signMessage += 'timestamp\n' + msg.timestamp + '\n';
    signMessage += 'topic_urn\n' + msg.topic_urn + '\n';
    signMessage += 'type\n' + msg.type + '\n';
    return signMessage;
}
```

NOTE

The sample code has passed the test on Nodejs v14.17.5.

Go

```
package demo

import (
    "bytes"
    "crypto"
    "crypto/rsa"
    "crypto/x509"
    "encoding/base64"
    "encoding/json"
    "encoding/pem"
    "fmt"
    "io/ioutil"
)

type Message struct {
    Signature        string  `json:"signature"`
    Subject          *string `json:"subject"`
    TopicUrn         string  `json:"topic_urn"`
    MessageId        string  `json:"message_id"`
    SignatureVersion string  `json:"signature_version"`
    Type             string  `json:"type"`
    Message          string  `json:"message"`
    SubscribeUrl     string  `json:"subscribe_url"`
    UnsubscribeUrl   string  `json:"unsubscribe_url"`
    SigningCertUrl   string  `json:"signing_cert_url"`
    Timestamp        string  `json:"timestamp"`
}

func VerifyMessage(pemFile string, message string) bool {
    msg := Message{}
    err := json.Unmarshal([]byte(message), &msg)
    if err != nil {
        fmt.Println("Convert json to struct failed")
        return false
    }
    pemContent, err := ioutil.ReadFile(pemFile)
    if err != nil {
        fmt.Println("Read pem file failed")
        return false
    }
    certDerblock, _ := pem.Decode(pemContent)
    if certDerblock == nil {
        fmt.Println("Decode pem file failed")
        return false
    }
    cert, err := x509.ParseCertificate(certDerblock.Bytes)
    if err != nil {
        fmt.Println("Parse cert failed")
        return false
    }

    msgString := buildMessage(&msg)
    msgHash := crypto.SHA256.New()
    msgHash.Write([]byte(msgString))
    msgHashSum := msgHash.Sum(nil)

    decodeSign, _ := base64.StdEncoding.DecodeString(msg.Signature)

    publicKey := cert.PublicKey.(*rsa.PublicKey)
    err = rsa.VerifyPKCS1v15(publicKey, crypto.SHA256, msgHashSum, decodeSign)
    if err != nil {
        fmt.Println("Verify failed")
        return false
    } else {
        fmt.Println("Verify success")
        return true
    }
}
```

```
func buildMessage(msg *Message) string {
    if msg.Type == "Notification" {
        return buildNotificationMessage(msg)
    } else if msg.Type == "SubscriptionConfirmation" || msg.Type == "UnsubscribeConfirmation" {
        return buildSubscriptionMessage(msg)
    }
    return ""
}

func buildNotificationMessage(msg *Message) string {
    buf := bytes.Buffer{}
    buf.WriteString("message\n" + msg.Message + "\n")
    buf.WriteString("message_id\n" + msg.MessageId + "\n")
       //The Subject field does not exist in msg, and this issue needs to be addressed.
    if msg.Subject != nil {
        buf.WriteString("subject\n" + *msg.Subject + "\n")
    }
    buf.WriteString("timestamp\n" + msg.Timestamp + "\n")
    buf.WriteString("topic_urn\n" + msg.TopicUrn + "\n")
    buf.WriteString("type\n" + msg.Type + "\n")
    return buf.String()
}

func buildSubscriptionMessage(msg *Message) string {
    buf := bytes.Buffer{}
    buf.WriteString("message\n" + msg.Message + "\n")
    buf.WriteString("message_id\n" + msg.MessageId + "\n")
    buf.WriteString("subscribe_url\n" + msg.SubscribeUrl + "\n")
    buf.WriteString("timestamp\n" + msg.Timestamp + "\n")
    buf.WriteString("topic_urn\n" + msg.TopicUrn + "\n")
    buf.WriteString("type\n" + msg.Type + "\n")
    return buf.String()
}
```

NOTE

The sample code has passed the test on Go 11.5

B Change History

Released On	Description
2023-01-31	This issue is the second official release, which incorporates the following change: Added **Logs**.
2022-09-30	This issue is the first official release.

