SMS HTTP API User Guide

Version	Date	Reason
1.0	03.10.2014	Initial version

Overview

This document describes the M800 SMS HTTP API for the purpose of integrating SMS into your applications and web systems. The M800 HTTP API is simple and easy to integrate into applications written in almost any programming language, because most languages have built-in support for making HTTP requests. SMS messages are sent to our servers in the same way that a user submits a website form (POST).

M800's HTTP API is used for one-way messaging only; the API supports HTTP POST requests for submitting messages. The Client issues an HTTP POST request to the M800 HTTP API with a list of required parameters. Then, M800 sends you an HTTP Response that indicates the validity of the transaction. HTTP persistent connections can be used to reduce connection overhead for increased message throughput.

Get Started

Sign Up

1. Go to www.m800.com.
2. Click Login/Sign Up.
3. Complete the form by choosing a username and password. The password must be at least eight characters long and must contain both numbers and letters.
4. Click **Sign Up**.

Validate Account

Once you have chosen your username and password, you will receive a confirmation email. Click the **Active Now** button in the email to complete the process of validating and setting up your M800 account.

Log In

Log in with your username and password (the page defaults to the M800 Dashboard).

- If you forget your password, click the Forgot Password link to reset your password.
- If you forget your username, contact support by live chat or email support@M800.com.

Buy SMS Plan

1. From the left-hand side of your M800 Dashboard, click **SMS** > **Home** > **Buy**.
2. Select the plan you want, and then click **Proceed to Checkout**.
3. Check the box to agree to the M800 Service Terms and Conditions, and click **Confirm**.
4. Enter your payment information. If you choose to pay by credit card, your details will be stored in our system to make it easy for you purchase additional products and numbers in the future.

Get API Key and API Secret

> You must purchase an SMS plan before you can get your API key and API secret.

1. From the left-hand side of your M800 Dashboard, click **SMS** > **API Info**.
2. The Dashboard automatically displays your API key and API secret.

Add Sender Number

A sender number is the number that displays when you send an SMS message. You can add up to five sender numbers to your SMS-enabled M800 account.

> **Notes**
> - You must purchase an SMS plan before you can add a sender number.
> - You must have at least one verified sender number before using the SMS HTPP API to send messages.
> - Due to provider/operator issues, the sender number may not be displayed in the target phone.

1. From the left-hand side of your M800 Dashboard, click **SMS** > **Source Addresses**.
2. Click the **Add New Number** button.
3. Select your country, enter a mobile number, and then click **Confirm**.
4. When you receive your verification code, enter the code in the pop-up window and click **Confirm** to begin using your sender number.

Prepare to Send SMS

The following section will guide you through the process of being able to send SMS messages through M800's API by submitting an authenticated HTTP POST request. This section includes instructions for the two methods of authentication:

1. Authorization header
2. URL-encoded query-string parameter

Prepare Message Content (Body)

Field Name	Description
from	This is your verified sender number in alphanumeric E.164 format with no "00" prefix and no "+" prefix.
type	The type is text.
messageClass	"normal" or "flash" Flash SMS is message that is displayed on the phone screen immediately upon arrival. Unless you choose to save the flash message, it will disappear upon navigating away and will not be saved in your inbox.
body	When sending an SMS message, the message content is in UTF-8 text format. If a message is too long to be sent as a single message, the SMS server will automatically split the long message in to smaller messages and send them one by one.

Example: Text SMS Request

```
 {
    "from": "85291111111",
    "type": "text",
    "messageClass": "normal",
    "body": "This a test SMS."

}
```

Two Methods for Authenticating HTTP API Requests

All requests require your API credentials. There are two methods for sending your API credentials with HTTP API requests:

1. Authorization header (more secure)
2. URL-encoded query-string parameter (less secure)

Method 1: Authorization Header

Using an authorization header is the most secure way to send your signature with an HTTP API request, because the API secret is hidden in the signature. M800 requires the following format for adding your API key and MD5 signature into the authorization header:

```
Authorization: MAAIISDK10 key="{apiKey}", nonce="{nonce}", signature="{signature}"
```

Notes

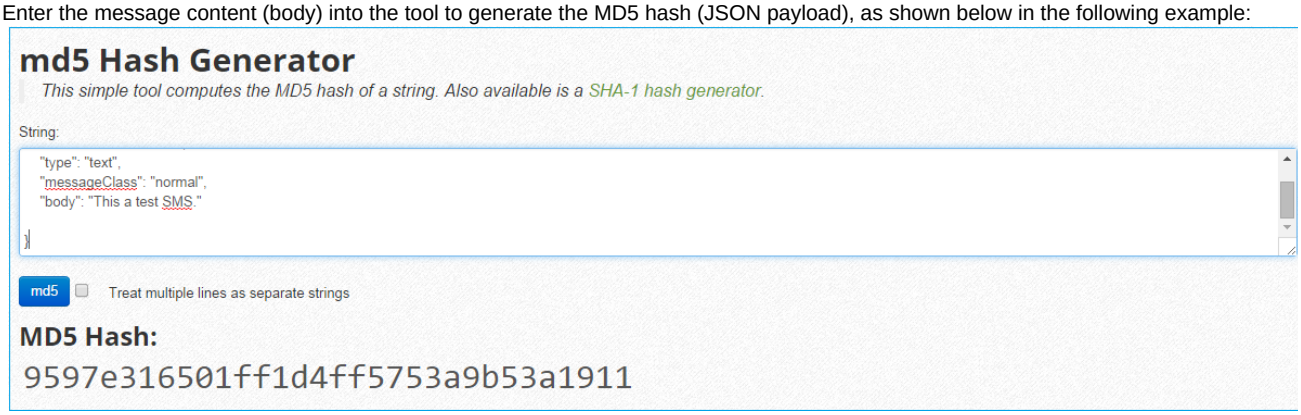
- Use an MD5 generator to obtain the MD5 hash for the Content-Md5 header.
- Once you've obtained the MD5 hash, use an SHA-256 HMAC generator to obtain the signature for the authorization header.
- The nonce is usually a number that is supplied by the developer. The developer should ensure the uniqueness of the nonce. If a nonce is reused, the request will be rejected.

Field	Description
apiKey	Get the API key from the M800 Dashboard.
nonce	This is the number that makes the request unique
signature	Use an SHA-256 HMAC signature generator to compute the signature for the authorization header as follows: `HEX(HMAC_SHA256($API_SECRET, $HTTP_METHOD + "\n" + $REQUEST_URI + "\n" + $CONTENT_MD5 + "\n" + $CONTENT_TYPE + "\n" + $DATE + "\n" + $NONCE ))`

Generate Signature

Use the following steps generate a signature:

1. Go to an MD5 generator tool, such as http://www.miraclesalad.com/webtools/md5.php, to generate the MD5 hash for the Content-Md5 header.
2. Enter the message content (body) into the tool to generate the MD5 hash (JSON payload), as shown below in the following example:



3. Go to an SHA-256 HMA generator tool, such as http://jetcityorange.com/hmac/. Enter the HTTP method, URI, MD5 hash, and body content type into the tool in order to generate the SHA-256 HMAC signature, as shown below in the following example:

Method 1: Submit HTTP POST Request

Once the authentication is ready, the HTTP POST request should contain the following information, which includes five headers (Content-Md5, Content-Type, Content-Length, X-M-Date, and Authorization):

HTTP Method	POST
URI	/api/1.0/sms/{destination number}
Content-Md5	The Content-MD5 entity-header field, as defined in RFC 1864, is an MD5 digest of the entity-body for the purpose of providing an end-to-end message integrity check (MIC) of the entity-body.
Content-Type	application/json; charset=utf-8
Content-Length	The length of the JSON contents
X-M-Date	Date and timestamp of the signature in dd/mm/yyyy HH:MM:SS format
Authorization	MAAIISDK10 key="{apiKey}", nonce="{unique custom ID}", signature="{signature generated by the SHA-256 HMAC signature generator}"

Method 1 Example: HTTP POST Request

```
POST /api/1.0/sms/85261111111 HTTP/1.0
Content-Md5: 9597e316501ff1d4ff5753a9b53a1911
Content-Type: application/json; charset=utf-8
Content-Length: 115
X-M-Date: Thu, 18 Sep 2014 18:49:58 GMT
Authorization: MAAIISDK10 key="a-demo-key", nonce="123456789",
signature="880657b2adef3630db5ee6d628157d83baa543e8422903cbb62a2a7d832ccb51"
{
    "from": "85291880678",
    "type": "text",
    "messageClass": "normal",
    "body": "This is a test."
}
```

Method 2: URL-Encoded Query-String Parameter

Using a URL-encoded query-string parameter is the least secure way to send your signature with an HTTP API request, because the API secret is exposed in the URL. Other end users can easily use the API, because the authentication details are specified in the URL when sending a request.

With this method, the API key and secret are specified in the query string.

The URL with authentication is:

```
https://api.m800.com/api/1.0/sms/{destination
number}?apiKey={a-demo-key}&apiSecret={a-demo-secret}
```

Protocol	HTTPS
API Server Host	api.m800.com
API URI	/api/1.0/sms/{destination number}
API Key	a-demo-key
API Secret	a-demo-secret

Method 2: Submit HTTP POST Request

Once the authentication is ready, the HTTP POST request can be submitted as follows:

HTTP method	POST
URI	/api/1.0/sms/{destination number}
Content-Type	application/json; charset=utf-8
Field	from, type, messageClass, and body parameters

Method 2 Example: HTTP POST Request

```
POST /api/1.0/sms/85261111111?apiKey=a-demo-key&apiSecret=a-demo-secret HTTP/1.0
Content-Md5: 9597e316501ff1d4ff5753a9b53a1911
Content-Type: application/json; charset=utf-8
Content-Length: 110
X-M-Date: Thu, 18 Sep 2013 18:49:58 GMT
{
    "from": "85291880678",
    "type": "text",
    "messageClass": "normal",
    "body": "This is a test."
}
```

Send SMS

If you are planning on sending more than 20 messages per minute, we recommend that you make use of HTTP persistent connections to avoid TCP connection overhead for every submission. For achieving even higher throughputs such as above 10 per second, HTTP pipelining provides support for asynchronous requests and responses.

Maximum Length

The maximum length of each SMS message depends on the encoding and the destination country. For long messages, we allow a maximum of five segments per message.

Destination country: China	
Short message:	
ASCII	135 characters
UCS-2	65 characters
Long message:	
ASCII	135 * 5 segments = 675 characters
UCS-2	65 * 5 segments = 325 characters

Destination country: All countries except China	
Short message:	
ASCII	160 characters
UCS-2	70 characters
Long message:	
ASCII	153 * 5 segments = 765 characters
UCS-2	67 * 5 segments = 335 characters

Response Parameters

Body Content Type: application/json; charset=utf-8

Field Name	Type	Required	Description
success	Boolean	true	True if all the message segments are sent successfully.

totalCost	Double	true	Total cost for sending all the messages in USD.
segmentCount	Integer	true	Number of segments created for the message in the request.
segments	SegmentDetails []	true	Information about each of the message segments (displayed in order of the body).
messageID	String	true	Message ID of this message segment.
status	String	true	Status of the message segment which states whether the message has been sent or not.
error	ErrorDetails	optional	Error details are displayed if the segment has errors.
code	Integer	optional	Error code
message	String	optional	Error message

Response Example (Success)

```
{
    "success": true,
    "totalCost": 0.05,
    "segmentCount": 1,
    "segments": [{
        "messageId": "456123",
        "status": "SENT"
    }]
}
```

Response Example (Failure)

```
{
    "success": false,
    "totalCost": 0.00,
    "segmentCount": 1,
    "segments": [{
        "messageId": "456123",
        "status": "FAIL",
        "error": {
            "code": 10001,
            "message": "SMSC is Down."
        }
    }]
}
```

Concatenated Messages

For incoming messages that exceed the maximum length allowed for the destination country, the sending carrier will break up the message behind the scenes before delivering it. M800's SMS server will treat the segmented message as separate incoming messages and deliver them to your application in the order we receive them.

> Concatenated messages are billed by the number of individual SMS messages used. For example, if you send a 459 character message, you will be charged for three SMS messages.

HTTP Error Codes

HTTP error codes are returned before and after each SMS request.

HTTP Error Response Parameters

Field Name	Type	Required	Description
error	ErrorDetails	true	Contains error information
status	Integer	true	HTTP Status Code
code	Integer	true	Error Code
message	String	true	Error Message

HTTP Error Example

```
{
    "error": {
    "status":400,
    "code":20000,
    "message":"Bad Request"
   }
}
```

HTTP Error Code List

Code	HTTP Code	Error	Message	Description
10000	401	UNAUTHORIZED	Request Unauthorized	No permission to login.
11000	403	FORBIDDEN	Request Unauthorized	Refused to use this feature.
11002	403	DEVELOPER NOT FOUND	Developer not found	Developer cannot be found in our datastore.
11004	403	CARRIER NOT ALLOWED USING SMS HTTP API	Carrier does not allow to use SMS HTTP API.	Carrier provided in the request is not allowed to use the SMS HTTP API.
11019	403	NOT ENOUGH BALANCE	Not enough balance. Failed to send SMS.	Not enough balance is left in the sender's wallet. Failed to be charged when sending SMS.
20000	400	BAD REQUEST	Bad Request	The request is unable to be processed.
20002	400	INVALID RECEIVER NUMBER	Invalid receiver phone number.	Receiver phone number in the URI is invalid.
20003	400	INVALID SENDER NUMBER	Invalid sender phone number.	Sender phone number in the request is invalid.
20004	400	INVALID MESSAGE CLASS	Invalid message class.	Message class provided in the request is invalid.
20006	400	INVALID MESSAGE LENGTH	Invalid message length. Failed to send SMS.	Message length provided in the body in the request is invalid.
21000	404	NOT FOUND	Not Found	The request is sent to the wrong API.
23000	429	TOO MANY REQUEST	Too Many Requests	There are too many requests are handling in this service. Please try again later.

30000	500	INTERNAL SERVER ERROR	Internal Server Error	Error found in our server. Need to troubleshoot at our side.
30001	500	FAILED TO SEND SMS	Failed to send SMS.	SMS server replies error after sending request to the SMS server.

HTTP API Authentication Error Code Examples

Code	HTTP Code	Error	Message	Description
10000	401	UNAUTHORIZED	Request Unauthorized	No permission to login.
11000	403	FORBIDDEN	Request Unauthorized	Not allowed to login.
11007	403	INVALID_MD5	Invalid MD5 is given in the request.	Wrong MD5 is provided in the header "Content-MD5" which does not match with the one that calculated with the POST contents.
11008	403	INVALID_DEVELOPER_KEY	Invalid Developer Key is given in the request.	Invalid Developer key is given in the authorization header.
11009	403	INVALID_NONCE	Invalid nonce is given in the request.	Invalid nonce is given in the authorization header.
11010	403	INVALID_SIGNATURE	Invalid signature is given in the request.	• Empty signature is given in the authorization header. • The signature calculated from the given parameters in the request is different from the one given in the authorization header. Therefore, it is invalid for this request.
11011	403	INVALID_DATE_FORMAT	Invalid date format is given in the request.	Date format provided is invalid in the header.
11012	403	INVALID_DATE	Invalid date is given in the request.	Date given in the header is out of range when comparing to the current date. Current date means the date in the moment that the HTTP API receiving the request.
11013	403	INVALID_POST_DATA	Invalid POST data is given in the request.	POST data is invalid to be parsed in the request.
11014	403	EMPTY_AUTHORIZATION_HEADER	No authorization header is provided in the request.	No authorization header can be found in the request.
11015	403	INVALID_AUTHORIZATION_SCHEME	Invalid scheme provided in the authorization header.	Wrong scheme is provided in the authorization header. Maybe the scheme is not match with the HTTP API authentication.
11016	403	INVALID_AUTHORIZATION_HEADER	Invalid authorization header.	Authorization header cannot be parsed by the API server.
11017	403	INVALID_API_KEY	Invalid API Key provided.	API key provided in the query parameter "apiKey" is not valid for authentication.
11018	403	INVALID_API_SECRET	Invalid API Secret provided.	API secret provided in the query parameter "apiSecret" is not valid for authentication.
30000	500	INTERNAL_SERVER_ERROR	Internal Server Error	Miscellaneous server error has occurred that cannot be exposed to the end user. Please contact M800 for further assistance.