HTTP API Specification V1.3

11th January, 2024

Contents

Introduction 3

Changelog 3

Getting Started 3

Basic commands 3

- Authentication 4
- Send a message 4

Message parameters 5

- Destination Address (numbers) 5
- Source Address (from) 5
- Delivery acknowledgment 6

Additional commands 6

- Query balance 6

SOAP Examples (HTTP POST) 6

- Send a message 6
- Query Balance 8

Short Message Peer-to-Peer Protocol 8

Appendix A: Message Statuses 9

Appendix B: Terminology 9

Introduction

This technical document is intended for developers who wish to use the Socnet HTTP API for sending messages, and describes the various programming methods and commands used by developers when using this API.

The HTTP API can be used for low or high-volume messaging. As HTTP is a means for relaying information, the HTTP API can be used with practically any web-service application. This is particularly useful for high-volume message sending.

To use this API, you need to register with Socnet Solutions. You will be given a username and password: keep these at hand. We will cover the HTTP method in this document.

This is one of the simpler server-based forms of communication to our gateway. It can be used either in the form of a HTTP POST, or as a URL (GET). We recommend POST for larger data transfer, due to the size limitations of GET. All calls to the API must be URL-encoded.

Note: It is important that the ENTIRE document is read before contacting support.

Changelog

This API is a revision of the second version, 1.2, which has been in production. The major enhancement is the addition of an alternative backup SMPP option.

Getting Started

In order to use the Socnet gateway you need a Socnet account. Obtain this by contacting Socnet Solutions at the offices or via https://www.socnetsolutions.com/projects/bulk/. You may need to use a browser that supports Adobe Flash Player, or to install the same in your browser. Once you have obtained and activated your account, you will receive 50 free credits for testing purposes.

To start sending messages, you need to purchase more credits. Optionally, you might have to supply your originating public IP address for using the API. Confirm this with Socnet Solutions technical team.

Basic commands

In order to send a message, the system will firstly need to authenticate you as a valid user. You will have to pass your account details with every command. The commands are made up of two segments: authentication and call parameters depending on which call is being made.

Basic Command Structure:

https://www.socnetsolutions.com/projects/bulk/amfphp/services/blast.php?username=xxx&passwd=xxx&msg=xxx&numbers=xxx,yyy

Authentication

In order to deliver a message, the system needs to authenticate the request as coming from a valid source. We use a number of parameters to achieve this:

- spname (or sometimes username for GET requests): This is the username of your account.
- sppass (passwd): The current password you have set on your account.
- Optionally, the source IP address of the call (which you do not have to supply).

Additionally we can force an IP lockdown, allowing only requests sent from IP addresses that you have specified.

Send a message

To facilitate sending an SMS with a single command, we have the ability to post *spname* and *sppass* variables in blast (GET) or sendmsg (POST). One can send to multiple destination addresses by delimiting the addresses with commas. The basic parameters required are *to* (the handset number to which the message is being sent) and *msg* (the content of the message). Additionally, you can choose the content-type of the response using the *type* parameter. The maximum number of comma separated destination addresses per blast (GET) or sendmsg (POST) depends on how much GET can allow you to, if you are calling the command via a GET, or alternatively, up to more than 300 destination addresses if you are submitting via a POST.

Each message returns a unique identifier in the form of an API message ID. This can be used to track and monitor any given message. The API message ID is returned after each post.

Sample Command:

https://www.socnetsolutions.com/projects/bulk/amfphp/services/blast.php?spname=xxx&sppass=xxx&type=xml&msg=xxx&numbers=xxx,yyy

Sample Response (xml):

```
<socnetblast>
        <status>Login ok</status>
        <messageId>48133</messageId>
        <info>Send ok: 2 numbers scheduled.</info>
        <credit>2400</credit>
</socnetblast>
```

Or

```
<socnetblast>
        <info>message sent [2/2]</info>
        <status>
                <item>
                        <id>1234</id>
                        <msisdn>256774715632</msisdn>
                </item>
                <item>
                        <id>1235</id>
                        <msisdn>256701608449</msisdn>
                </item>
        </status>
</socnetblast>
```

The number of <item> tags returned depend on how many destination numbers were in the request command. The returned <id> can be used to check the status of (query) the message.

- type: The content-type of the response. May be one of xml (eXtensible Markup Language), json (JavaScript Object Notation), or text; defaults to text.
- msg: Contents of the message, URL encoded as necessary. In case the content is more than 160 characters, your account will be charged depending on the message parts.
- numbers: Phone number of the receiver. To send to multiple receivers, separate each entry with comma (,).

Message parameters

Destination Address (numbers)

SMS messages need to be sent in the standard international format, with country code followed by number. No leading zero to the number and no special characters such as "+" or spaces must be used. For example, a number in Uganda being 0772134567 should be changed to 256772134567.

Mobile numbers starting with a 0 (zero) may not be delivered.

Source Address (from)

The source address (from), also known as the sender ID, can be either a valid international format number between 1 and 16 characters long, or an 11 character alphanumeric string. These must be registered within your account and approved by Socnet before they may be used. In some cases, and once approved, this parameter may not be submitted at the time of using the API but is set by Socnet and once set, it will be the source address of messages sent using your account.

Note that characters such as spaces, punctuation, Unicode and other special characters may not always be supported to all destinations and could interfere with your delivery. We suggest that you refrain from using such characters on the source address. If this is set, then delivery acknowledgements may be unavailable. The use of an alphanumeric source address with 8-bit messaging may cause message failure. This service is not guaranteed across all mobile networks worldwide and may interfere with delivery to certain handsets, such as those with DND (Do not Disturb) activated.

Delivery acknowledgment

In order to determine whether an SMS has been received by a handset or not, we request delivery acknowledgement for every message we send. The ability to receive reliable delivery acknowledgements varies between mobile networks. Please test to a specific mobile network first, before assuming that you will receive handset acknowledgments for messages that are delivered.

If a GSM handset is unavailable, e.g. switched off or out of coverage, the SMS will be delivered according to a retry cycle once the handset is back in coverage. A delivery receipt will only be returned if and when the retry is delivered. If the validity period or retry cycle (typically 24 hours) is exceeded, the SMS will fail and may simply show "Submitted" or status 8.

Additional commands

Query balance

This will return the number of credits available on this particular account. The account balance is returned as a numeric value. You will still need to authenticate with *spname* and *sppass*.

Sample Command:

https://www.socnetsolutions.com/projects/bulk/amfphp/services/blast_bal.php?spname=xxx&sppass=xxx&type=xml

Sample Response:

```
<socnetblast>
        <credits>54</credits>
</socnetblast>
```

SOAP Examples (HTTP POST)

Send a message

Request posted at https://www.socnetsolutions.com/projects/bulk/http/sendmsg.php

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"  xmlns:SOAP-
ENC="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"                    xmlns:ns1="SOCNETPlatForm"
xmlns:ns2="HTTPSendMsg">
<SOAP-ENV:Body>
        <ns:receiveMTSMS xmlns:ns="HTTPSendMsg">
                <mtSMS>
                        <spname>xxx</spname>
                        <sppass>xxx</sppass>
                        <msg>xxx</msg>
                        <numbers>
                                <number>xxx</number>
                                <number>xxx</number>
                        </numbers>
                </mtSMS>
        </ns:receiveMTSMS>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Sample Response:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<SOAP-ENV:Envelope          SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"                                   xmlns:SOAP-
ENC="http://schemas.xmlsoap.org/soap/encoding/">
<SOAP-ENV:Body>
        <ns1:receiveMTSMSResponse xmlns:ns1="HTTPSendMsg">
                <info xsi:type="xsd:string">message sent [2/2]</info>
                <status xsi:type="SOAP-ENC:Array" SOAP-ENC:arrayType=":[2]">
                        <item xsi:type="xsd:">
                                <id>1234</id>
                                <msisdn xsi:type="xsd:string">xxx</msisdn>
                        </item>
                        <item xsi:type="xsd:">
                                <id>1235</id>
                                <msisdn xsi:type="xsd:string">xxx</msisdn>
                        </item>
                </status>
        </ns1:receiveMTSMSResponse>
</SOAP-ENV:Body>
```

```
</SOAP-ENV:Envelope>
```

Query Balance

Request posted at: https://www.socnetsolutions.com/projects/bulk/http/getcredits.php

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:ns1="SOCNETPlatForm" xmlns:ns2="HTTPQueryBal">
<SOAP-ENV:Body>
        <ns:creditsQry xmlns:ns="HTTPSendMsg">
                <qrysms>
                        <spname>xxx</spname>
                        <sppass>xxx</sppass>
                </qrysms>
        </ns:creditsQry>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Sample Response:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<SOAP-ENV:Envelope SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:SOAP-
ENC="http://schemas.xmlsoap.org/soap/encoding/">
<SOAP-ENV:Body>
        <ns1:creditsQryResponse xmlns:ns1="HTTPSendMsg">
                <credits xsi:type="xsd:string">52</credits>
        </ns1:creditsQryResponse>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Short Message Peer-to-Peer Protocol

SMPP is an open, industry standard protocol designed to provide a flexible data communication interface for the transfer of short message data between ESMEs, REs and SMSC. Whereas the primary connection is done via HTTP, this connection-oriented option is provided only as a backup. This should only be used when the primary (production) route is, for one reason or another, unavailable.

Upon request and depending on the nature of your account, your technical team will be provided with the following in order to make this connection (when necessary):

- Host IP address (SMS-C),
- Port number (Transmit only),
- SMS-C Username, and
- SMS-C Password.

Your technical team is advised to code (program) and automate this switch over should an attempt fail through the primary HTTP connection. It is however strongly advised to only make the switch only when the primary route becomes unavailable.

Appendix A: Message Statuses

These are message statuses that are generated after the Socnet gateway has accepted your message for delivery:

Code	Description	Detail
1	Delivered	Delivered to phone
2	Failed	Non-Delivered to Phone
4	Queued	Queued on SMSC
8	Submitted	Delivered to SMSC
16	Rejected	Non-Delivered to SMSC
32	Pending	SMPP unbound but retrying
64	Undelivered	SMPP client timeout
128	Expired	Retried long enough
256	Unknown	Unknown network
512	DND	Do not disturb

Appendix B: Terminology

API: Application programming interface.

ESME: External Short Messaging Entity.

GSM: Global System for Mobile Communications.

GET: Requests a representation of the specified resource. Requests using GET (and a few other HTTP methods) “SHOULD NOT have the significance of taking an action other than retrieval”.

HTTP: Hypertext Transfer Protocol.

POST: Submits data to be processed to the identified resource. The data is included in the body of the request. This may result in the creation of a new resource or the updates of existing resources or both.

RE: Routing Entity.

SOAP: Simple Object Access Protocol.

SMPP: Short Message Peer-to-Peer.

SMSC or **SMS-C**: Short Message Service Center.