TECHNICAL SPECIFICATIONS ON

PUSH SMS CONNECTIVITY API

Version 2.5.3

SSL WIRELESS

93 B, New Eskaton Road
Dhaka 1000
Bangladesh
Work: +880 9666 77 6969. +880 2 831 6969.
Fax: +880 2 913 2172.
E-mail: info@sslwireless.com
Web: www.sslwireless.com

Table Of Contents

PURPOSE OF THIS DOCUMENT

This document is for companies/clients who wish to integrate with SSL's Push SMS gateway hereafter referred to as Push Gateway. This document contains detailed information about the methods for integrating this gateway over HTTP/HTTPS protocol using GET/POST method. As a GET/POST client over HTTP/HTTPS protocol can be implemented by using various programming languages, this document is designed both as a getting started guide for the technical staff/developer and a reference document throughout the project implementation.

The client system would require sending SMS messages via GET/POST of parameters as per specified in this document from SSL's Push SMS Gateway. The gateway will access the parameters sent from the client system and send the SMS message(s) to the recipient(s) accordingly. This specification is under active development, any updates to the API specifications may not be notified.

A detailed description of the API parameters is available in this document. However, in order to send the SMS messages via this Gateway, you are required to have the following:

- a client application to send the required parameters
- a registered account with SSL Wireless

Note: Your client application or server need not be limited to any platforms/languages. In this document we have provided typical examples for connecting to our gateway using PHP scripting language in order to get you started.

For more details or examples not included in this specification, please contact us through email at engg@sslwireless.com.

ABOUT SSL WIRELESS

SSL Wireless is the leading Application Service Provider (ASP) in Bangladesh. SSL provide mobility and internet solutions to corporate, financial institutes, and SME segments and is connected to multiple banking and financial institutions. SSL is also connected to all the mobile network operators in the country for provisioning of mobile financial services. SSL specializes in the SMS, MMS, IVR, WAP, JAVA, and WEB based solutions. SSL works with telecommunication and internet technologies and has been the leading ASP to work on SMS, IVR, JAVA, WAP, WEB, and GPRS as a revenue generating tool for enterprises, media channels, and financial institutes.

SSL is connected to multiple leading private Banks and Financial Services Institutes of Bangladesh offering them Mobile Banking (i.e. M-Banking), Merchant Gateway (i.e. SSLCOMMERZ), and Mobile Insurance (i.e. M-Insurance).

SSL establishes physical secured links with all of these financial institutions over VLAN (Virtual LAN) or VPN (Virtual Private Network) tunnels through multiple Wide Area Networks (WANs) such as MetroNet, Acenet, Telnet, Link3, etc.

We also offer Merchant Payment Gateway services through a manageable SSLCOMMERZ front-end as part of the gateway solution. The merchants will be able to view/download/print/email their transactions, payments, and settlement reports from SSLCOMMERZ. SSL offers total Support and Maintenance of the gateway.

We also provide Mobile Airtime Recharge services for end-clients. Any client can connect with SSL's Virtual Recharge Gateway to send mobile airtime recharge (Top-Up) requests for any Bangladeshi mobile phone numbers in Bangladesh.

GATEWAY FUNCTIONALITY

SSL's Push SMS Gateway offers standard methods for connecting any client application through SSL-enabled systems to the client application for sending SMS messages to mobile subscribers in Bangladesh and also in abroad. You are able to identify mobile phone number and SMS Message Body according to these specifications. You can integrate to the gateway via widely used web technology - HTTP.

This API makes it easy to integrate your client applications to the Push SMS Gateway.

The following is a simple architecture reflecting the relationships between Customer/user/subscriber, Your application, SSL's Push SMS Gateway and the telecom operators.

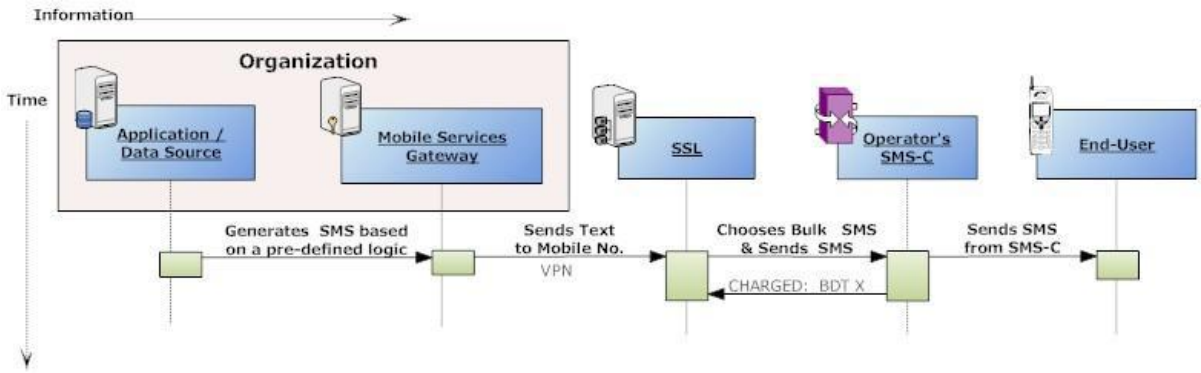

Figure 1: SSL's Push SMS Gateway

The following steps describe the procedure of a customer receiving a SMS text from your application through SSL's Push SMS gateway and the telecom operator's SMS-C.

1. The client application submits the Push SMS information to the SSL's Push SMS Gateway via the given API parameters.

2. The Push SMS Gateway sends the message to the respective telecom operator.

3. The telecom operator sends the messages to the recipient mobile phone number.

API SPECIFICATIONS

1.1 INTRODUCTION

The client application has to open a HTTP connection to send SMS messages to SSL's Push SMS Gateway. The gateway will receive the hits from the client application with detailed information regarding a customer's SMS Message for processing.

This API is only for limited usage; in per API call it is expected to receive only 150 SMS messages.

1.2 LIST OF METHODS

The following is a list of parameters at this API. The different parameters for this method are described in the following table.

1.2.1 Send SMS Request

To send SMS, you have to use post method (You can also use curl library). A sample code in HTML is given below for generating SMS request from a web page.

HTML-FORM Example:-

```
<form action="http://sms.sslwireless.com/pushapi/dynamic/server.php" method="post">

<input  type="hidden" value="UserName" name="user" />
<input  type="hidden" value="Password" name="pass" />
<input  type="hidden" value="XXXXXXXXXXXXXXXXXXXX" name="sid" />

<input  type="hidden" value="880XXXXXXXXX" name="sms[0][0]" />
<input  type="hidden" value="Test SMS One" name="sms[0][1]" />
<input  type="hidden" value="123456789" name="sms[0][2]" />

<input  type="hidden" value="880XXXXXXXXX " name="sms[1][0]" />
<input  type="hidden" value="Test SMS Two" name="sms[1][1]" />
<input  type="hidden" value="123456790" name="sms[1][2]" />
<input type="submit" />
</form>
```

PHP-CURL-POST Example:-

```
<?php

$user = "UserName";
$pass = "Password";
$sid = "XXXXXXXXXXXXXXXXXXXX";
$url="http://sms.sslwireless.com/pushapi/dynamic/server.php";
$param="user=$user&pass=$pass&sms[0][0]= 880XXXXXXXXX &sms[0][1]=".urlencode("Test
SMS 1")."&sms[0][2]=123456789&sms[1][0]= 880XXXXXXXXX &sms[1][1]=".urlencode("Test
SMS &2")."&sms[1][2]=123456790&sid=$sid";
$crl = curl_init();
curl_setopt($crl,CURLOPT_SSL_VERIFYPEER,FALSE);
curl_setopt($crl,CURLOPT_SSL_VERIFYHOST,2);
curl_setopt($crl,CURLOPT_URL,$url);
curl_setopt($crl,CURLOPT_HEADER,0);
curl_setopt($crl,CURLOPT_RETURNTRANSFER,1);
curl_setopt($crl,CURLOPT_POST,1);
curl_setopt($crl,CURLOPT_POSTFIELDS,$param);

$response = curl_exec($crl);
curl_close($crl);
echo $response;

?>
```

PHP-CURL-GET Example:-

You can send only one sms by each request through GET Method.

```
<?php
$curl = curl_init();
 curl_setopt_array($curl, array( CURLOPT_RETURNTRANSFER => 1, CURLOPT_URL =>
'http://sms.sslwireless.com/pushapi/dynamic /server.php
?user=USER NAME&pass=PASSWORD&sid=STAKEHOLDER&sms='.urlencode("Test SMS
3").'&msisdn=8801672270833&csmsid=123456789', CURLOPT_USERAGENT => 'Sample cURL Request' ));
 $resp = curl_exec($curl);
 curl_close($curl);
echo $resp; ?>
```

C#- Example One:-

```
using System.Web.UI;
using System.Web.UI.WebControls;

namespace TestPushAPI
{
   public partial class Default : System.Web.UI.Page
   {
      protected void Page_Load(object sender, EventArgs e)
      {

         //Procedure one
         String sid = "STAKEHOLDER";
         String user = "USER NAME";
         String pass = "PASSWORD";

         String URI = "http://sms.sslwireless.com/pushapi/dynamic/server.php";
         String myParameters = "user=" + user + "&pass=" + pass +"&sms[0][0]=88**********&sms[0][1]="
+System.Web.HttpUtility.UrlEncode("Test SMS1\nTest SMS2\nTest SMS API3") + "&sms[0][2]=" + "1234567890" +
"&sms[1][0]=88**********&sms[1][1]=" +System.Web.HttpUtility.UrlEncode("TESTSMS2\nTESTSMS3") +
"&sms[1][2]=" + "1234567890" + "&sid=" + sid;

         using (WebClient wc = new WebClient())
         {
            wc.Headers[HttpRequestHeader.ContentType] = "application/x-www-form-urlencoded";

            string HtmlResult = wc.UploadString(URI, myParameters);

            Console.Write(HtmlResult);
         }
```

C#- Example Two:-

```
         String sid = "STAKEHOLDER";
         String user = "USER NAME";
         String pass = "PASSWORD";
         string URI = "http://sms.sslwireless.com/pushapi/dynamic/server.php";
         string myParameters = "user=" + user + "&pass=" + pass + "&sms[0][0]=8801913900620&sms[0][1]=" +
System.Web.HttpUtility.UrlEncode("Test sms 2") + "&sms[0][2]=123456789&sms[1][0]=8801913900620&sms[1][1]=" +
System.Web.HttpUtility.UrlEncode("Test sms 2") + "&sms[1][2]=123456790&sid=" + sid;
         HttpWebRequest request = (HttpWebRequest)WebRequest.Create(URI);
         byte []data = Encoding.ASCII.GetBytes(myParameters);
         request.Method = "POST";
         request.ContentType = "application/x-www-form-urlencoded";
         request.ContentLength = data.Length;
         using (Stream stream = request.GetRequestStream())
         {
            stream.Write(data, 0, data.Length);
         }
         HttpWebResponse response = (HttpWebResponse)request.GetResponse();
         String responseString = new StreamReader(response.GetResponseStream()).ReadToEnd();
         Response.Write(responseString);
```

Java-POST Example:-

```
import  java.io.BufferedReader;
import  java.io.DataOutputStream;
import  java.io.InputStreamReader;
import  java.net.URL;
import  java.net.URLEncoder;
import  java.util.UUID;
import  sun.net.www.protocol.https.HttpsURLConnectionImpl;
import  java.net.HttpURLConnection;

public  static  String  sendSMS(String  msisdn,  String  sms_string)
    {
       try
       {
         String  url  =  "https://sms.sslwireless.com/pushapi/dynamic/server.php";

         URL  obj  =  new  URL(url);
         HttpsURLConnectionImpl  con  =  (HttpsURLConnectionImpl)  obj.openConnection();

      //add  reuqest  header
      con.setRequestMethod("POST");
      con.setRequestProperty("User-Agent",  "USER_AGENT");
      con.setRequestProperty("Accept-Language",  "en-US,en;q=0.5");
      String  args  =  "user=username&pass=password&sid=SID  &sms[0][0]="  +  msisdn  +  "&sms[0][1]="  +  sms_string
+  "&sms[0][2]="+UUID.randomUUID();

      //  Send  post  request
      con.setDoOutput(true);
      DataOutputStream  wr  =  new  DataOutputStream(con.getOutputStream());
      wr.writeBytes(args);
      wr.flush();
      wr.close();

      BufferedReader  in  =  new  BufferedReader(
            new  InputStreamReader(con.getInputStream()));
      String  inputLine;
      StringBuffer  response  =  new  StringBuffer();

      while  ((inputLine  =  in.readLine())  !=  null)  {
       response.append(inputLine);
      }
      in.close();
      return  response.toString();
        }
        catch(Exception  ex){
         return  "<SMS_STATUS>ERROR</SMS_STATUS>";
        }
    }
```

Java-GET Example:-

```
import  java.io.BufferedReader;
import  java.io.DataOutputStream;
import  java.io.InputStreamReader;
import  java.net.URL;
import  java.net.URLEncoder;
import  java.util.UUID;
import  sun.net.www.protocol.https.HttpsURLConnectionImpl;
import  java.net.HttpURLConnection;

 public  static  String  sendSMS(String  msisdn,  String  sms_string)
    {
       try
       {
         String  url  =  "http://sms.sslwireless.com/pushapi/dynamic  /server.php";
```

```
        String args = "?msisdn="+msisdn+"&sms="+URLEncoder.encode(sms_string)+"& user=USER
NAME&pass=PASSWORD&sid=STAKEHOLDER&csmsid=123456789";

        URL obj = new URL(url+args);
        HttpURLConnection con = (HttpURLConnection) obj.openConnection();
     con.setRequestMethod("GET");
    con.setRequestProperty("User-Agent", "USER_AGENT");
    int responseCode = con.getResponseCode();

    BufferedReader in = new BufferedReader(
          new InputStreamReader(con.getInputStream()));
    String inputLine;
    StringBuffer response = new StringBuffer();

    while ((inputLine = in.readLine()) != null) {
     response.append(inputLine);
    }
    in.close();

     return response.toString();
       }
       catch(Exception ex){
        return "<SMS_STATUS>ERROR</SMS_STATUS>";
       }
    }



     }
  }
}
```

Oracle Example:

```
CREATE OR REPLACE FUNCTION test_func
return varchar2
is
var_userid varchar2(20);
var_pass varchar2(20);
var_sms varchar2(1000);
var_msisdn varchar2(50);
var_uniqueid varchar2(50);
var_sid varchar2(200);
var_response varchar2(1600);
var_url varchar2(2000);
req   utl_http.req;
resp  utl_http.resp;
begin
 var_userid:='USER NAME';
 var_pass:=' PASSWORD ';
 var_sid:='XXXXXXXXXXXXXXXXXXXX';
 var_msisdn:='8801XXXXXXXXX';
 var_sms:='TESTSMS';
 var_uniqueid:='54212367';

var_url:='http://sms.sslwireless.com/pushapi/dynamic/server.php?user='||var_userid||'&pass='||var_pass||'&sid='||var_sid||'&s
ms='||var_sms||'&msisdn='||var_msisdn||'&csmsid='||var_uniqueid;
 Utl_Http.set_response_error_check (ENABLE => TRUE);
 Utl_Http.set_detailed_excp_support (ENABLE => TRUE);
 req := Utl_Http.begin_request(url => var_url, method => 'GET');
 Utl_Http.set_header (r => req, NAME => 'User-Agent', VALUE => 'Mozilla/4.0');
 resp := Utl_Http.get_response (r => req);
  BEGIN
     LOOP
```

```
    Utl_Http.read_text (r => resp, DATA => var_response);
    END LOOP;
  EXCEPTION
     WHEN Utl_Http.end_of_body
    THEN
      var_response:='NULL';
  END;
Utl_Http.end_response (r => resp);
var_response:=trim(upper(var_response));
RETURN var_response;
END test_func;
```

Input parameters descriptions are given below:-

Name	Parameter	Description	Type	Length	Requirement
URL	Action	To submit the value	Alphanumeric	30-100	Mandatory
User Name	user	For authentication. Provided by SSL.	Alphanumeric	4-20	Mandatory
Password	pass	For authentication. Provided by SSL.	Alphanumeric	4-20	Mandatory
Stakeholder ID	sid	This is a unique id for the specific brand/masking name for the client, provide by SSL	Alphanumeric	20-30	Mandatory
Mobile Number	sms[0][0]	This is the mobile phone number of the customer/user to receive the SMS message.	Numeric	13	Mandatory
Message Body	sms[0][1]	This is the SMS message body to be received by the customer/user.	Alphanumeric	1-450	Mandatory
Client Ref ID	sms[0][2]	For each sms there will be unique reference ID from Client	Alphanumeric	1-25	Mandatory

The aforementioned parameters will be sent as per specified here from client application to Push SMS Gateway for sending the appropriate SMS Message to customer/end-user's mobile number.

Response:

Push API response will be in XML. Sample XML response is given below.

```
<?xml version='1.0' encoding='ISO-8859-1' ?>
<REPLY>
<PARAMETER>OK</PARAMETER>
<LOGIN>SUCESSFULL</LOGIN>
<PUSHAPI>ACTIVE</PUSHAPI>
<STAKEHOLDERID>OK</STAKEHOLDERID>
<PERMITTED>OK</PERMITTED>
<SMSINFO>
<MSISDN>8801913900620</MSISDN>
<SMSTEXT>Test SMS 1</SMSTEXT>
<CSMSID>123456789</CSMSID>
<REFERENCEID>20130511070404629309006 20</REFERENCEID>
</SMSINFO>
<SMSINFO>
<MSISDN>8801913900620</MSISDN>
<SMSTEXT>Test SMS 2</SMSTEXT>
<CSMSID>123456790</CSMSID>
<REFERENCEID>20130511070404540689006 20</REFERENCEID>
</SMSINFO>
</REPLY>
```

Response parameters descriptions are given below:

The response will be received as XML in the HTTP Response body with following tags.

Name	Parameter	Description	Output
PARAMETER	PARAMETER	If all parameters are exists with correct fromate it will return "OK" , else it will return "All PARAMETERS ARE NOT EXISTS"	OK/ All PARAMETERS ARE NOT EXISTS
Login	LOGIN	It's define the authentication is Successful or Fail.	SUCCESSFUL / FAIL
Pushapi	PUSHAPI	Its defines user id and stakeholder id both are "ACTIVE" or "INACTIVE" for pushapi	ACTIVE / INACTIVE
sid	STAKEHOLDERID	Active or Inactive for respective	OK / INVALID
PERMITTED	PERMITTED	If provided stakeholder id is not permitted for this user id it returns "FAIL" else "OK"	OK / FAIL
Each sms information	SMSINFO	Under this tag, each sms information will be Show.	
Mobile Number	MSISDN	Your provided mobile number will be here.	880XXXXXXXXX
MSISDNSTATUS	MSISDNSTATUS	Checking your provided mobile number is valid or invalid.	Invalid Mobile No
Message Body	SMSTEXT	Your provided sms text will be here.	
Client Ref ID	CSMSID	Your provided reference ID will be here	
Reference ID	REFERENCEID	For each sms SSL will provide a	20-30 digit

		reference ID. This means, we have successfully received your request.	

Note: Do not resend any sms, if you do not get response. You need to inform us with your CSMSID.

1.2.2 Send Bangla SMS

To send Bangla sms you need create a stakeholder by communicating with us. You cannot send Bangla sms using the stakeholder, which is created for sending English SMS. Sms sending procedure is same as English sms, which is all ready describe in 1.2.1 section. For Bangla sms you need to convert the Bangla text to Unicode. Here sample converter function is given below.

c#:

```
public string convertBanglatoUnicode(string banglaText) {
    StringBuilder sb = new StringBuilder();
    foreach (char c in banglaText)
    {
        sb.AppendFormat("{1:x4}", c, (int)c);
    }
   string unicode = sb.ToString().ToUpper();
   return unicode;
}
```

PHP:

```
function convertBanglatoUnicode($BanglaText)
{
   $unicodeBanglaTextForSms = strtoupper(bin2hex(iconv('UTF-8', 'UCS-2BE', $BanglaText)));
   return $unicodeBanglaTextForSms;
}
```

Reports

We will provide you a web based report panel where you can view all sent SMS report.

END OF DOCUMENT