App Note

VoxStack GSM Gateway API

Rev: 2.0

Date: On July 7, 2014

From: OpenVox support group

Contact Info: support@openvox.com.cn

目录

OpenVox GSM 网关的 HTTP 接口

OpenVox 网关提供了丰富的外部接口用来处理客户多样化的短信服务需求，OpenVox 一直致力于更好的满足客户的需求和营造使用方便灵活的客户体验！前期我们一直使用的 AMI 接口和目前刚完成的 API 接口一起构成了目前 OpenVox GSM 网关完整丰富的短信系统和接口类型！

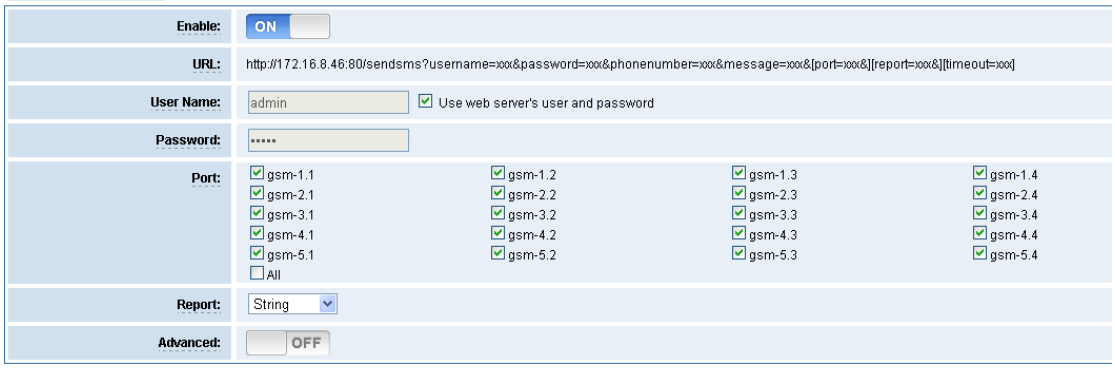
OpenVox 的短信 HTTP 接口是为了满足客户对 GSM 网关与自身短信平台的兼容性和开发的简单易用的要求而开发完成！我们相信您通过使用我们的 HTPP 接口，完全可以花费更少的时间和代价来构建属于您自己的短信平台！当然如果您已经拥有的非常稳定和功能完备的短信平台，我们同样相信您可以做很少的改动，甚至不需改动就可以将 OpenVox GSM 网关的短信功能融合到您的短信平台！

HTTP to SMS 配置

发送短信：

第一步： 在 OpenVox 网关配置 HTTP 发送短信接口！
进入菜单：SMS—>SMS Settings—>HTTP to SMS,如下图所示：

第二步：使能 HTTP to SMS,如下图所示:



Enable: 使能 HTTP to SMS 功能，默认为关闭！

URL: 发送 SMS 的 URL，参数说明如下：

username: 认证用户名，您可以使用 WEB 的登陆用户名作为用户名，也可以自定义用户名！

password: 认证密码，您可以使用 WEB 登陆的认证密码，也可以自定义认证密码！

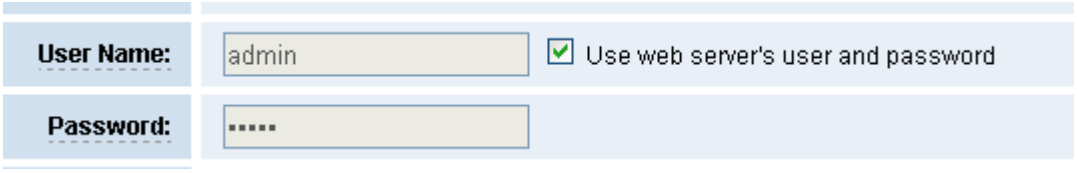
phonenumber: 您要发送的目的地，一般为目的手机号码！

message: 发送的消息内容!

port: 可选参数，需要指定哪个端口发送，比如：port: gsm-1.2

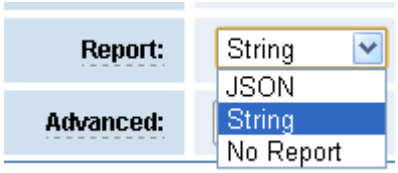
report:可选参数，发送消息报告的格式，比如：report=String

timeout: 超时时间，单位为毫秒，比如：timeout=5000



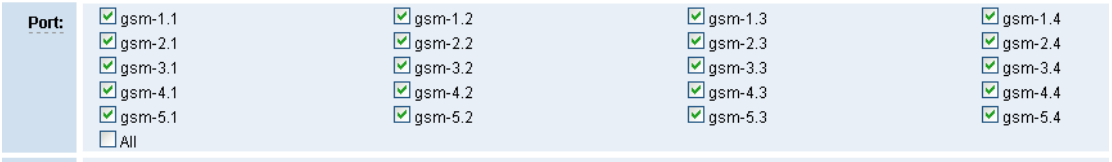
User Name: 自定义用户名，注意：如果勾选“Use web server’s user and password”则会使用 WEB 登陆用户名和密码；

Password: 自定义密码，注意：如果勾选“Use web server’s user and password”则会使用 WEB 登陆用户名和密码；



Report: 消息报告的格式，目前一共支持 2 种消息报告的合适，分别是 String,JSON;如果选择 No Report,则表示不需要消息报告！

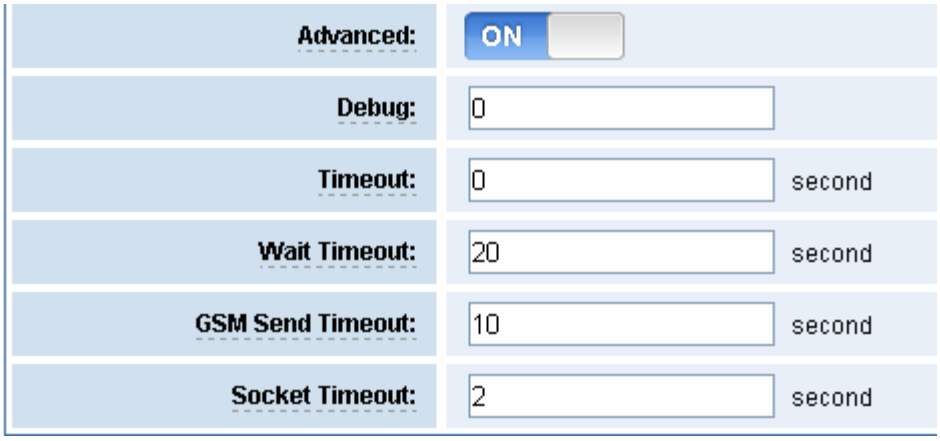
Port:



您想选择哪个端口发送短信，当网关收到一个 HTTP 请求发送短信时将会从您选

择的端口中选择一个端口发送短信，为降序排列，比如：您第一次发送为 gsm-1.1，那么第二次则会选择 gsm-1.2 发送，依次类推！

Advance 选项：



Debug: 调试消息的级别（注意：该功能主要为技术支持人员调试错误使用，建议您保持默认值）；

Timeout: 发送超时时间，默认为 0，单位为秒；

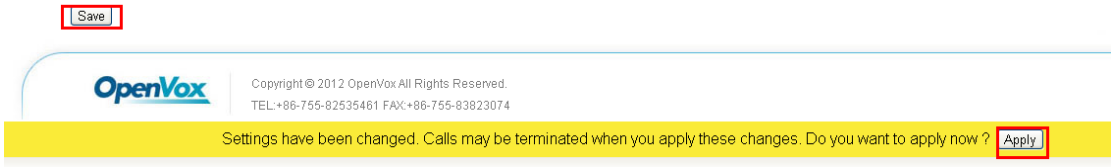
Wait Timeout: 等待消息报告超时时间，默认 20 秒，单位为秒;

GSM Send Timeout: GSM 端发送超时时间，默认 10 秒，单位为秒；

Socket Timeout: Socket 套接字链接超时时间，默认 2 秒，单位为秒;

注意： 建议保持设置 advance 值为默认值或者不开启 Advance 选项！

(当您配置好之后，请点击保存按钮保存，并且应用，如下图：)



第三步：使用 HTTP 接口发送短信！

尝试在浏览器输入一个 URL，然后敲击 ENTER 键，网关会将消息报告返回给当前页！如下图所示：

HTTP to SMS 示例代码

使用 PHP 调用网关的 HTTP 接口，如下示例：

```
#!/usr/bin/php -q
<?php
        function SendSMS()
        {
          // 1. 初始化一个 cURL 会话(Init an CURL session)
                $ch = curl_init();

               // 2. 设置请求选项，包括具体的 url(Setting the requestion options)
                curl_setopt($ch,CURLOPT_URL,"http://172.16.8.46:80/sendsms?username=admin&password=admin&phonenumber=13632919026&message=test&port=gsm-4.2&report=String&timeout=5");
            // 3. 执行一个 cURL 会话并且获取相关回复(Get the reply)
                  $response = curl_exec($ch);

            // 4. 释放 cURL 句柄,关闭一个 cURL 会话(Release the Curl handle)
                              curl_close($ch);
            }
            SendSMS();
 ?>
```

执行结果如下：

```
[root@Centos test]# ./test.php
message: test

record start

--record 1 start--
port: gsm-4.2
phonenumber: 13632919026
time: 2014-07-07 12:12:57
result: success
--record 1 end--

record end
[root@Centos test]#
```

SMS to HTTP 配置

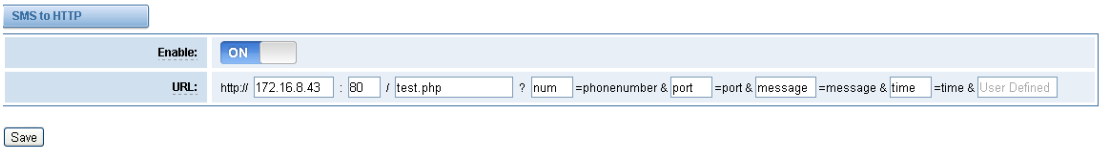
接收短信：

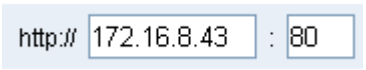
目前我们采用的是主动投送的方式将网关收到的短信投送到一个具体的 URL 链接地址！

第一步： 在 OpenVox 网关配置 HTTP 接收短信接口！

进入菜单：SMS—>SMS Settings—>HTTP to SMS,如下图所示：

第二步：使能 SMS to HTTP,如下图所示：

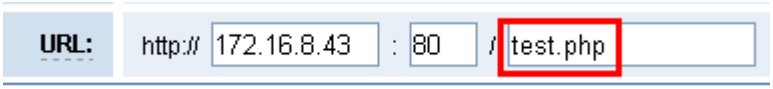




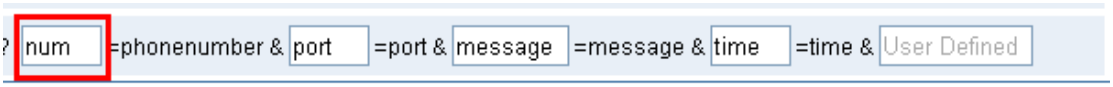
Hostname: 存放资源的服务器的域名系统(DNS) 主机名或 IP 地址。有时，在主机名前也可以包含连接到服务器所需的用户名和密码；

整数，可选，省略时使用方案的默认端口，各种传输协议都有默认的端口号，如

http 的默认端口为 80。如果输入时省略，则使用默认端口号。有时候出于安全或其他考虑，可以在服务器上对端口进行重定义，即采用非标准端口号，此时，URL 中就不能省略端口号这一项。



由零或多个“/”符号隔开的字符串，一般用来表示主机上的一个目录或文件地址。

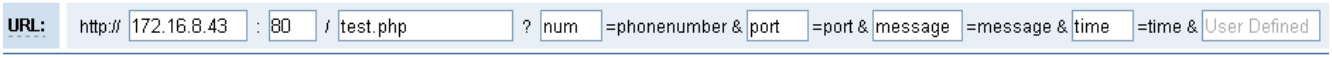


用于给动态网页（如使用 CGI、ISAPI、PHP/JSP/ASP/ASP。NET 等技术制作的网页）传递参数，可有多个参数，用“&”符号隔开，每个参数的名和值用“=”符号隔开。

第三步：使用 HTTP 接口接受短信

注意：使用本实例前请确保您已经正确安装并配置了 WEB 服务器！本实例在 apach 服务器下运行！

GSM 网关配置如下：



SMS to HTTP 示例代码

在/var/www/html 下创建文件 test.php

```
<?php
        $num=$_GET['num'];
        $port=$_GET['port'];
        $message=$_GET['message'];

        $Begin = "----------Received Message----------\n";
        $Message = "Received Message:".$message."\n";
        $MyPhonenumber  = "Phomenumber:".$num."\n";
        $MyTime = "Time:".$time."\n";
        $MyPort = "Port:".$port."\n";

        $Time = date("Y-m-d H:i:s");
        $MyTime = "Time:".$Time;
```

```
$MyStr = $Begin.$Message.$MyPhonenumber.$MyPort.$MyTime."\n";

$file_pointer = fopen("/var/www/html/message.txt","a+");
fwrite($file_pointer,$MyStr);
fclose($file_pointer);

?>
```

在/var/www/html 下创建文件 message.txt

尝试发送一条短信给网关，然后打开 message.txt 查看，结果如下图所示：

vim message.txt

由结果可以看出，网关在 2014-07-07 日收到短信两条，结果都来自 gsm-4.2

补充一: AMI 接口发送长短信

目前 OpenVox 网关能支持发送长短信，不管您是采用网关本身的 WEB GUI，或者 AMI 还是 HTTP 接口都可以发送长短，一下主要针对 AMI 接口发送长短信进行说明。

第一步：尝试使用命令方式发送长短信！(详细可查看本公司 AMI 文档进行参考)，如下图所示：

第二步：发送长短信

注意：当您使用 AMI 接口发送长短信（超过 160 个字节）时您需要对您的短信内容进行拆分(分条发送)，然后再进行发送！这样能保证您所受到的长短信为一整条！

发送命令：

gsm send sync csms <destination> <message> <flag> <smscount> <smssequence> <timeout>

span: 您发送短信所使用的端口，比如填写: gsm-1.1 表示利用第一个端口发送长短信

destination: 接受短信的手机号码

message: 消息内容

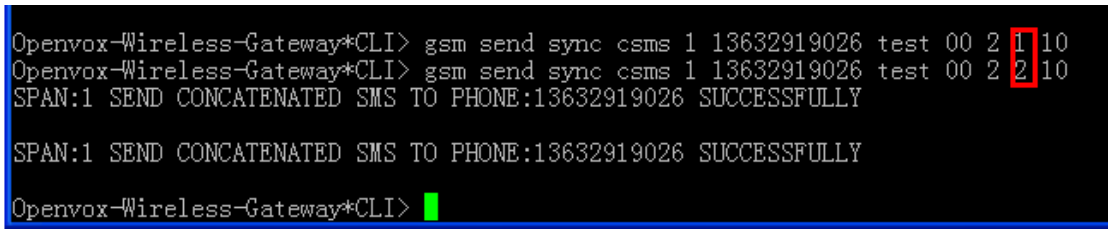
flag: 当前发送长短信的标识，比如您需要将当前一条超过 160 个字节的短信分两条发送，那么您必须保证这两条命令拥有同样的标识，标识范围为(00--ff)

smscount: 拆分的数量，比如您需要拆分 4 条发送，那么您应该填写 4

smssequence: 发送的队列，比如您需要拆分 2 条发送，那么第一条命令的 smssequence 为 1，第二条命令的 smssequence 参数应该为 2，以此类推

timeout: 发送超时时间，单位为毫秒

示例：



说明：该示例将 8 个字节的内容拆分为两次发送，第一次发送一个 test,第二次再次发送一个 test，您可以看到您只收到了一条短信，短信内容为:testtest!

补充二: AMI 接口发送长短信示例代码

注意：为了满足客户的多样化需求，本示例采用 C 语言所写！

运行环境：linux

编译命令：gcc sendsms.c –o sendsms

运行命令: ./sendsms

完整代码如下：

```
#include<stdio.h>
#include<string.h>
#include<stdlib.h>
#include<sys/socket.h>
#include<netinet/in.h>
#include<unistd.h>

//Login
void login_fun(int sock_fd)
{
	char username[20];
	char secret[20];
	int login_len=0;
	int secret_len=0;
	int res = 0;
	char receive_buf[4096];
	char login_buf[40];

	memset(username,'\0',20);
	memset(secret,'\0',20);
	printf("please input your username\n");
	scanf("%s",username);
	fflush(stdin);
	printf("please input your secret\n");
	scanf("%s",secret);
	fflush(stdin);

	memset(login_buf,'\0',40);
	sprintf(login_buf,"action:Login\r\nusername:%s\r\nsecret:%s\r\n\r\n",usernam
e,secret);

	login_len = strlen(login_buf);

	if(res = write(sock_fd,login_buf,login_len) == login_len)
	{
		sleep(1);
```

```
            memset(receive_buf,'\0',4096);
            if(res = read(sock_fd,receive_buf,sizeof(receive_buf))<0)
            {
                perror("login failed\n");
                return;
            }
            printf("%s\n",receive_buf);
            if(NULL != strstr(receive_buf,"Authentication accepted"))
            {
                printf("login success\n");
            }
        }
}



void sendsms_fun(int sock_fd)
{
    char send_buf[4096];
    char span_num[3];
    char destination[12];
    char message[2048];
    int res;
    int send_len;
    char receive_buf[4096];

    int SMS_Lengh = 160;
    int Message_Lengh = 0;
    int Divided_Lengh =152;
    char segmentation_message[153];
    int temp_count = 0;
    int smscount = 0;
    int smssequence = 1;
    int i = 0;
    char flag[3] = "00";
    int timeout = 10;

    memset(send_buf,'\0',4096);
    memset(destination,'\0',11);
    memset(message,'\0',2048);
    memset(span_num,'\0',3);

    printf("please input the span you want used\n");
    scanf("%s",span_num);
```

```
fflush(stdin);
printf("please input the destination num you want send\n");
scanf("%s",destination);
fflush(stdin);
printf("Please input the message you want send\n");
scanf("%s",message);
fflush(stdin);

Message_Lengh = strlen(message);
printf("Input  Message Lengh is:%d\n",Message_Lengh);
if(Message_Lengh > SMS_Lengh)                    //Send Long SMS
{
        if(Message_Lengh%Divided_Lengh == 0)
            {
                smscount = Message_Lengh/Divided_Lengh;
            }
            else
            {
                smscount = Message_Lengh/Divided_Lengh+1;
            }
            printf("The SMS will be divided into %d part
send\n",smscount);
            while(temp_count!=smscount)
            {
                for(i = 0;i<Divided_Lengh;i++)
                {

                segmentation_message[i]=message[temp_count*Divid
                ed_Lengh+i];
                }

                printf("The %d part is:\n",temp_count+1);
                printf("%s\n",segmentation_message);

sprintf(send_buf,"action:Command\r\ncommand:GSM send sync
csms %s %s %s %s %d %d %d\r\n\r\n",span_num,destination,segmentation_messag
e,flag,smscount,smssequence+temp_count,timeout );
                printf("send_buf:%s\n",send_buf);
              send_len = strlen(send_buf);
               memset(receive_buf,'\0',4096);
              printf("%s\n",send_buf);
             if(res = write(sock_fd,send_buf,send_len) == send_len)
             {
                sleep(1);
```

```
                                        if(res                                                  =
read(sock_fd,receive_buf,sizeof(receive_buf))<0)
                                        {
                                                perror("send sms error\n");
                                               return;
                                        }
                                        printf("%s\n",receive_buf);
                                        if(NULL != strstr(receive_buf,"Response: Follows"))
                                        {
                                                printf("send SMS success\n");
                                        }
                                    }
                                     temp_count++;
                            }

        }
        else
        {
                sprintf(send_buf,"action:Command\r\ncommand:GSM                         send
sms %s %s %s\r\n\r\n",span_num,destination,message);
                send_len = strlen(send_buf);

                memset(receive_buf,'\0',4096);
                printf("%s\n",send_buf);
                if(res = write(sock_fd,send_buf,send_len) == send_len)
                {
                        sleep(1);
                        if(res = read(sock_fd,receive_buf,sizeof(receive_buf))<0)
                        {
                                perror("send sms error\n");
                                return;
                        }
                        printf("%s\n",receive_buf);
                        if(NULL != strstr(receive_buf,"Response: Follows"))
                        {
                                printf("send SMS success\n");
                        }
                }
        }
}

int main(void)
{
        int client_socket;
```

```
    struct sockaddr_in client_addr;
    char ServerIp[20];

    memset(ServerIp,'\0',20);
    printf("Please input your server ip address\n");
    scanf("%s",ServerIp);
    fflush(stdin);

    client_socket = socket(AF_INET,SOCK_STREAM,0);
    if(client_socket < 0)
    {
        perror("create socket error\n");
        return -1;
    }
    client_addr.sin_family = AF_INET;
    client_addr.sin_port      = htons(5038);
    client_addr.sin_addr.s_addr = inet_addr(ServerIp);

    if(connect(client_socket,(struct sockaddr *)&client_addr,sizeof(client_addr))<0)
    {
        perror("connect error\n");
        return -1;
    }
    else
    {
        printf("connect to %s success\n",ServerIp);
    }
    //login
    login_fun(client_socket);
    //send SMS
    sendsms_fun(client_socket);
    return 0;
}
```