App Note

VoxStack GSM Gateway API

Rev: 1.0
Date: August 05, 2013, Rev 1.0
From: OpenVox support group
Contact Info: support@openvox.com.cn

OpenVox VoxStack GSM Gateway is a feature-rich, high availability, flexible modular gateway product. This Application Note introduces some methods about "How to use API of OpenVox GSM Gateway". This API is based on Asterisk Management Interface (AMI).

Asterisk Management Interface (AMI) is a powerful telnet-like text based interface allow 3rd party to control almost all the functions of Asterisk. OpenVox GSM Gateway has integrated and enhanced AMI, extended AMI command and AMI event. Next section we will demo the AMI in OpenVox GSM Gateway Step by Step.

TABLE OF CONTENTS

AMI in OpenVox GSM Gateway Introduction

OpenVox GSM Gateway AMI allows client's management program to connect Asterisk in OpenVox GSM Gateway, and allows it to send commands or read events via TCP/IP flow. It is very useful, for example, you can use this feature to send/receive SMS/USSD.

A simple "key: value" protocol is used to convey messages between client's management program and Asterisk in OpenVox GSM Gateway. Every lines use "line break" (\r\n) to end.

Protocol Characteristics

- You need to create a connection before sending commands.
- The packets can be conveyed in both directions at any time.
- When the first line of packets is "Action", it indicates that the packets have been conveyed from client's management program to OpenVox GSM Gateway Asterisk.
- When the first line of packets is "Event" or "Response", it indicates that the packets from OpenVox GSM Gateway Asterisk have been received by client's management program.
- The "break line" is used to isolate every line. Double "return" indicates the end of commands and OpenVox GSM Gateway Asterisk begins to deal with commands.

The Type of Packets

The type of packets is identified by some key words.

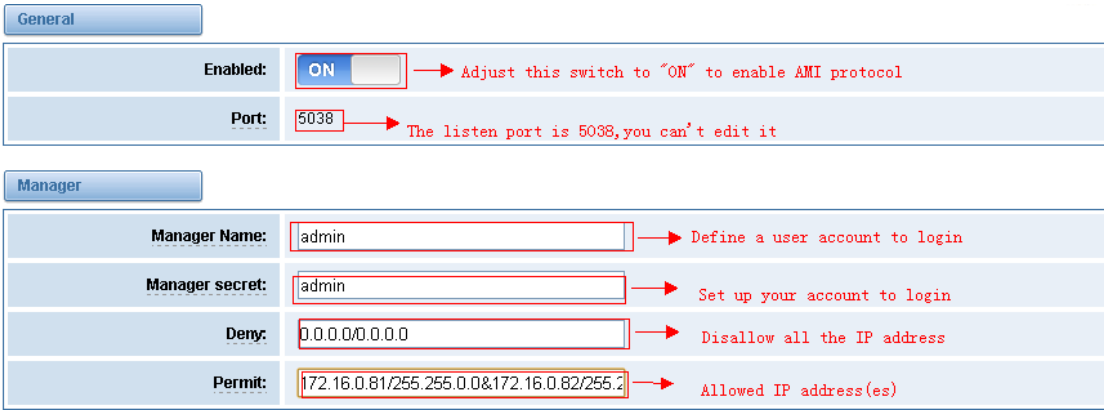
- Action: Client's management program requests Asterisk to execute special action. Just some limited actions can be used.
- Response: The OpenVox GSM Gateway Asterisk responses the action from client's management program.
- Event: Some information about OpenVox GSM Gateway Asterisk core events.

Configure Connection and Authentication

In order to create a session between OpenVox GSM Gateway Asterisk and client's management program, the client's management program must create a TCP/IP connection that listens to the port (5038) of OpenVox GSM Gateway Asterisk, and use "login" action to authentication. So we need to create a user in OpenVox GSM Gateway Asterisk. You can configure it in the file "/etc/asterisk/manager.conf" or by Web interface. The user is composed of user name, password and permissions list.

First Part (Configure It in Web)

Log in your OpenVox GSM Gateway with your user account and password, and please follow this flow to configure: ADVANCED → Asterisk API. Please change the default settings to yours.



Notice: Concerning the "Permit" option, if you have one more IP addresses, you can input them with symbol "&". In the demo test, allow servers 172.16.0.81 and 172.16.0.82 to access to the GSM Gateway.

You can define read/write authorization for various events.

Notice: In this illustration, you will be able to define custom authorization for various events, "Read" authorization permits you to receive asynchronous events, in general, "write" authorization permits you to send commands and get responses.

Second Part (Configure It in Configure File)

Please open the file "/etc/asterisk/manage.conf".

Command:

vi /etc/asterisk/manage.conf

```
[general]
bindaddr=0.0.0.0
enabled=yes
port=5038
[admin]
secret=admin
permit=0.0.0.0/0.0.0.0
read=system,call,log,verbose,agent,user,config,dtmf,reporting,cdr,dialplan
write=system,call,log,verbose,command,agent,user,config,reporting,originate
```

You can edit the file and save, then reload the configuration.

Command:

asterisk –r

core reload

Create Connection

Using Telnet to Demonstrate AMI Connectivity over TCP Socket

Here shows how to get access to the gateway and some responds to the actions from OpenVox GSM Gateway AMI.
You must send a "Login" action to log in your GSM Gateway Asterisk server. Then input your username and password.

```
root@Openvox-Wireless-Gateway: # telnet 172.16.127.127 5038
Asterisk Call Manager/1.1
action: Login
username: admin
secret: admin

Response: Success
Message: Authentication accepted

Event: FullyBooted
Privilege: system,all
Status: Fully Booted
```

Notice: If you want to send commands, please press double [Enter] key.

Now we have guided you to the first command:

Login: Start a Manager session

AMI over TCP for Windows

Yes, you can get access to your OpenVox GSM Gateway over AMI protocol in Windows system. Please follow this flow:
Click "Start → Run → Open", at the input text box, type "cmd" to enter a Windows console.

```
C:\Documents and Settings\Administrator>telnet 172.16.127.127 5038
```

Hit the [Enter] key, and it will automatically skip to the next illustration.

Using Program to Demonstrate AMI Connectivity over TCP Socket

This is an example that using C program to login the asterisk server, it just guides you to login the OpenVox GSM Gateway asterisk server.

```
#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",username,secre
t);

        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");
                }
        }
}

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);
    //connect
        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_fun(client_socket);
        return 0;
}
```

Introduction of SMS/USSD Sending/Receiving

OpenVox GSM Gateway extends AMI commands and AMI events to enable SMS/USSD support. Developer can integrate USSD/SMS application through OpenVox GSM Gateway AMI. Next section we will demo the AMI in OpenVox VoxStack GSM Gateway Step by Step.

Introduction of SMS/USSD Sending Commands

Command:

GSM send sms <destination> <message> <timeout> [id]

GSM send ussd <destination> <message> <time out> [id]

Arguments:

span: Which GSM channel will you select to send SMS

destination: the number that the short message will be sent to

timeout: How long it will try to send SMS before time is up

id: Identifier of SMS.id is an optional parameter.

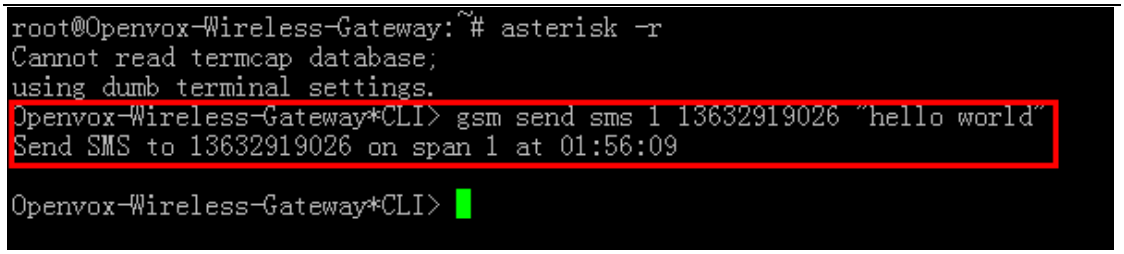
For example:

Step 1: Please login your OpenVox GSM Gateway via ssh.

Step 2: Run the command "asterisk -r".

Step 3: Send SMS using this command

We use the first span to send SMS to 13632919026, and the content is "hello word".



Using Telnet to Demonstrate Send/Receive SMS via AMI

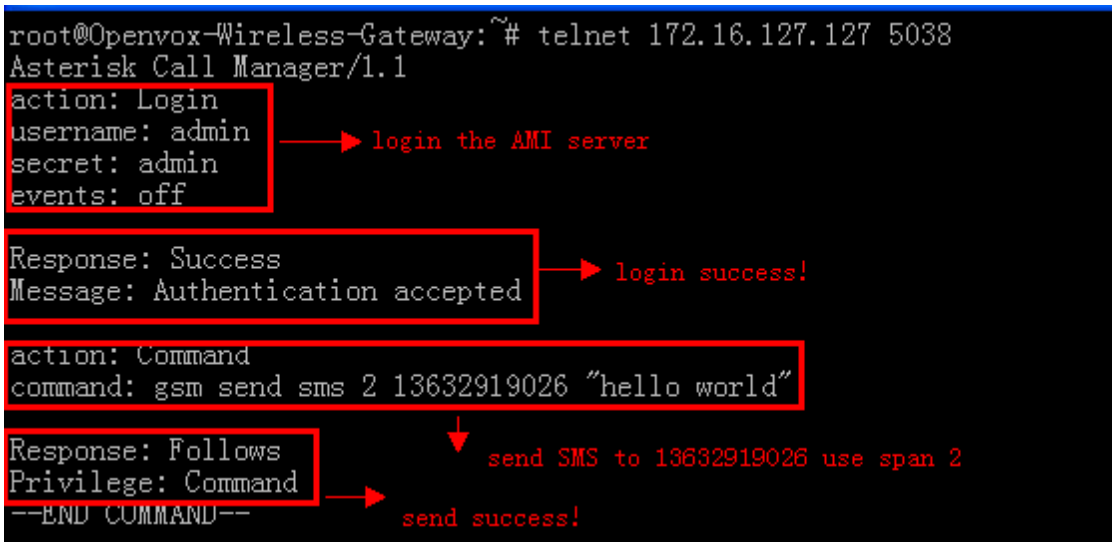
At first we introduce you an OpenVox GSM Gateway AMI action to you.

Action: command.

Description: Execute a CLI command

Step 1: Login your AMI server.

Step 2: Use the AMI action to send SMS



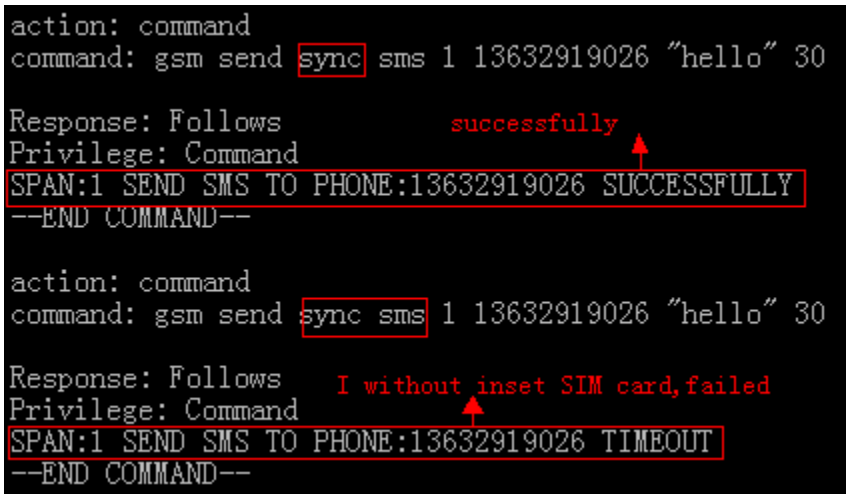
The client manager will receive an event repot when you have message comes in.

For example:

Notice： If you need know the status that send sms. Please use

gsm send sync sms <destination> <message> <timeout> [id]

for example:



```
Event: VarSet
Privilege: dialplan,all
Channel: EXTRA-SMS/3-1
Variable: SMSSRC
Value: +8613632919026
Uniqueid: 1377584220.2

Event: VarSet
Privilege: dialplan,all
Channel: EXTRA-SMS/3-1
Variable: SMSTXT
Value: test
Uniqueid: 1377584220.2

Event: VarSet
Privilege: dialplan,all
Channel: EXTRA-SMS/3-1
Variable: SMSPDU
Value: 0891683108705505F0040D91683136929120F60000318072416193230 4F4F29C0E
Uniqueid: 1377584220.2

Event: VarSet
Privilege: dialplan,all
Channel: EXTRA-SMS/3-1
Variable: SMSTIME
Value: 2013/08/27 14:16:39
Uniqueid: 1377584220.2

Event: VarSet
Privilege: dialplan,all
Channel: EXTRA-SMS/3-1
Variable: SMSTZ
Value: GMT+8
Uniqueid: 1377584220.2

Event: VarSet
Privilege: dialplan,all
Channel: EXTRA-SMS/3-1
Variable: DIALSTATUS
Value: SMS_END
Uniqueid: 1377584220.2
```

```
Event: Newexten
Privilege: dialplan,all
Channel: EXTRA-SMS/3-1
Context: gsm-2
Extension: sms
Priority: 1
Application: System
AppData: /my_tools/process_sms "2" "+8613632919026" "2013/08/27 14:16:39" "test"
Uniqueid: 1377584220.2
```

In the whole transaction, you can find a code segment above. This section is the most important when you want to monitor the incoming short messages. Asterisk (Gateway core) will report a new event to the client.

Arguments:

Event: Newexten

When a new short message comes in, Asterisk (Gateway core) will report a new event to client.

Priviledge: Dialplan

Allowed event

Channel: EXTRA-SMS/3-1

The channel to be used

Context: gsm-2

Context name

Extension: sms

Transaction type. When the short message comes in, the gateway will invoke sms extension.

Priority: 1

Executed priority first while shore message coming in.

AppData: This is a short message from my mobile phone. Data which will be saved in CDR

Notice: If TCP socket connection is still alive, and you receive parameters Newexten and sms, that indicates there is a new short message coming in. You can use Ping action to check if your connection is alive or not, and monitor the incoming short message by these two events.

Using the Program to Create a Simple SMS/USSD Center

This program works with C language sending and receiving SMS, based on our first program.

Compile command:

gcc send_sms.c –o send_sms

```
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];
```

```
    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);

    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;
}
```

Now use the program receive SMS.

This program base on our first program.

Compile command: gcc send_sms.c –o send_sms

```
void readsms_fun(int sock_fd)
{
    struct Message
    {
        char sender_num[12];
        char span[3];
        char message_buf[4096];
        char receive_time[30];
    };
    char receive_buf[4096];
    char *receive_AppData = NULL;
    struct Message SMS_buf;
    int res;
    int i = 0;
    char *temp = NULL;
    char temp_buf[1024];

    while(1)
    {
        sleep(1);
        memset(receive_buf,'\0',4096);
        memset(SMS_buf.sender_num,'\0',12);
        memset(SMS_buf.span,'\0',3);
        memset(SMS_buf.message_buf,'\0',4096);
        memset(SMS_buf.receive_time,'\0',30);
        memset(temp_buf,'\0',1024);

        if(res = read(sock_fd,receive_buf,sizeof(receive_buf))<0)
        {
            perror("read  failed\n");
        }
        else
        {
            if((receive_AppData = strstr(receive_buf,"AppData"))!=NULL)
            {
                printf("%s\n",receive_AppData);
```

```
                strcpy(temp_buf,receive_AppData);
            }
            if((temp=strstr(temp_buf,"process_sms"))!=NULL)
            {
                for(i = 0;i<13;i++)
                {
                    temp++;
                }
                strncpy(SMS_buf.span,temp,1);
                for(i=0;i<7;i++)
                {
                    temp++;
                }
                strncpy(SMS_buf.sender_num,temp,11);
                for(i =0;i<14;i++)
                {
                    temp++;
                }
                strncpy(SMS_buf.receive_time,temp,19);
                for(i = 0;i<22;i++)
                {
                        temp++;
                }
                strcpy(SMS_buf.message_buf,temp);

                printf("span = %s\n",SMS_buf.span);
                printf("num=%s\n",SMS_buf.sender_num);
                printf("time  = %s\n",SMS_buf.receive_time);
                printf("message = %s\n",SMS_buf.message_buf);
            }
            //printf("span = %s\n",SMS_buf.span);
        }
    }
}

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_fun(client_socket);
//sendsms_fun(client_socket);
readsms_fun(client_socket);
return 0;
 }
```

OpenVox GSM Gateway AMI Actions

Client manage program can send OpenVox GSM Gateway AMI Actions package to request Gateway to execute a specific action. We have made a demo for you in the previous statement. Now we will do a comprehensive introduction for you about OpenVox GSM Gateway AMI Actions.

OpenVox GSM Gateway AMI Action Package

Extra key words can provide more actions information when client manage program sends actions. For example, maybe you want to disconnect a channel, you can pass a variable to the dialplan if your operations may cause asterisk to execute the entry of dialplan. This is same for passing the key words.

When you send action packages to OpenVox GSM Gateway, you can do it as follows:

Action: <action type> <CRLF>

<Key 1>: <Value1> <CRLF>

<Key 2>: <Value 2> <CRLF>

…….

Variable: <Variable 1>=<Value 1><CRLF>

Variable: <Variable 2>=<Value 2><CRLF>

…….

<CRLF>

The Action List

Action	Privilege	Synopsis
waitEvent	<none>	wait for an event to occur
SIPnotify	system,all	send a SIP notify
SIPshowregistry	system,reportin	reporting Show SIP registrations (text format)
SIPqualifypeer	system,reportin	reporting Qualify SIP peers
SIPshowpeer	system,reportin	reporting show SIP peer (text format)
SIPpeers	system,reportin	reporting List SIP peers (text format)
AGI	agi,all	add an AGI command to execute by Async AGI
DBDelTree	system,all	delete DB Tree
DBDel	system,all	delete DB entry

DBPut	system,all	put DB entry
DBGet	system,reportin	reporting Get DB Entry
Bridge	call,all	bridge two channels already in the PBX
Park	call,all	park a channel
ParkedCalls	<none>	list parked calls
ShowDialPlan	config,reportin	reporting Show dialplan contexts and extensions
AOCMessage	aoc,all	generate an advice of charge message on a channel
ModuleCheck	system,all	check if module is loaded
ModuleLoad	system,all	module management
CoreShowChannel	system,reportin	reporting List currently active channels
Reload	system,config	send a reload event
CoreStatus	system,reportin	reporting Show PBX core status variables
CoreSettings	system,reportin	reporting Show PBX core settings (version etc)
UserEvent	user,all	Send an arbitrary event
UpdateConfig	config,all	Update basic configuration
SendText	call,all	Send text message to channel
ListCommands	<none>	List available manager commands
MailboxCount	call,reporting,	Check Mailbox Message Count
MailboxStatus	call,reporting,	Check mailbox
AbsoluteTimeout	system,call,all	Set absolute timeout
ExtensionState	call,reporting,	Check Extension Status
Command	command,all	Execute Asterisk CLI Command
Originate	originate,all	Originate a call
Atxfer	call,all	Attended transfer
Redirect	call,all	Redirect (transfer) a call
ListCategories	config,all	List categories in configuration file
CreateConfig	config,all	Creates an empty file in the configuration directory
Status	system,call,	List channel status
GetConfigJSON	system,config	Retrieve configuration (JSON format)
GetConfig	system,config,a	Retrieve configuration
Getvar	call,reporting,	Gets a channel variable
Setvar	call,all	Set a channel variable
Ping	<none>	Keepalive command
Hangup	system,call,all	Hangup channel.
Challenge	<none>	Generate Challenge for MD5 Auth
Login	<none>	Login Manager
Logoff	<none>	Logoff Manager
Events	<none>	Control Event Flow
DataGet	<none>	Retrieve the data api tree

Introduction of OpenVox GSM Gateway AMI Actions

You can browse all OpenVox GSM Gateway manage actions by the command "manager show commands".

A. **(1)Absolute Timeout**: This command will request OpenVox GSM Gateway to drop a specified channel after specified seconds.

Parameters:

Channel: which channel you want to drop.

Timeout: drop channel after specify times.

Example:

Request information

Action: AbsoluteTimeout

Channel: SIP/1001-00000001

Timeout: 5

Return information

Response: Success

Message: Timeout Set

Notice: If you want to know the active channels, please run CLI command:

Core show channel SIP/[tab]

C. **(1)CoreShowChannles**: List current defined channels and some information about them.

Action: CoreShowChannels

Example:

Action: CoreShowChannels

Return information:

Response: Success

EventList: start

Message: Channels will follow

(2)CoreStatus: Query Core pbx status

Action: CoreStatus

Example:

Action: CoreStatus

Response: Success

CoreStartupDate: 2013-08-20

CoreStartupTime: 15:17:45

CoreReloadDate: 2013-08-20

CoreReloadTime: 18:00:56

CoreCurrentCalls: 0

(3)CoreSettings: Query core PBX settings

Action: CoreSettings

Example:

Action: CoreSettings

Response: Success

AMIversion: 1.1

AsteriskVersion: 1.8.20.0

SystemName:

CoreMaxCalls: 0

CoreMaxLoadAvg: 0.000000

CoreRunUser:

CoreRunGroup:

CoreMaxFilehandles: 0

CoreRealTimeEnabled: No

CoreCDRenabled: Yes

CoreHTTPenabled: No

(4)Command: Execute Asterisk CLI Command

Action: Command

Command: <value>

- Command - Asterisk CLI command to run.

Example:

Action: Command

Command: <value> (asterisk CLI commands)

(5)CreateConfig: Create an empty file in the configuration directory. This action is intended to be used before an updateconfig action.

Action: CreateConfig

Filename: <value> (The configuration filename to create)

Example:

Action: CreateConfig

Filename: test.conf

Notice: you can find the test.conf under /etc/asterisk/

E. **(1)Events**: Enable/Disable sending of events to this manager client.

Action: Events

EventMask: <value>

Arguments:

EventMask

on - If all events should be sent.

off - If no events should be sent.

system,call,log,... - To select which flags events should have to be sent.

Example:

Action: Events

EventMask: on

(2)ExtensionState: Report the extension state for given extension. If the extension has a hint, it will use devicestate to check the status of the device connectivity to the extension. Return an Extension Status message. The response will include the hint for the extension and the status.

Action: ExtensionState

Exten: <value>

Context: <value>

Example:

Action: ExtensionState

Context: default

Exten: 1001

Return:

Responese: success

Message: Extension Status

Exten: 1001

Context: default

Hint:

Status: -1

Status:

-1: can't find exten

0: ready

1: be used

2: busy

4: no available

8: ring

16:waiting

G. (**1)GetVar**: Get the value of a global or local channel variable

Example:

Action: GetVar

Channel: <value>

Variable: <value>

Arguments:

Channel: channel to read variable from

Variable: Variable name

H. **(1)Hungup**: Hang up a channel.

Example:

Action: Hangup

Channel: <value>

Cause: <value>

Arguments

ActionID - ActionID for this transaction. Will be returned.

Channel - The channel name to be hangup.

Cause - Numeric hangup cause.

Example:

ACTION: Hangup

Channel: SIP/x7065558529-99a0

L. **(1)ListCommands:** Returns the action name and synopsis for every action that is available to the user.

Example:

*Action: ListCommand*s

(2)Logoff: Log off the current manager session.

Example:

Action: Logoff

S. **(1)SIPpeers:** List SIP peers in text format, details on current status.peerlist will follow as separate events, followed by a final event called peerlistComplete.

Example:

Action: SIPpeers

Notice： The use of the details please refer to the link：

https://wiki.asterisk.org/wiki/display/AST/AMI+Actions