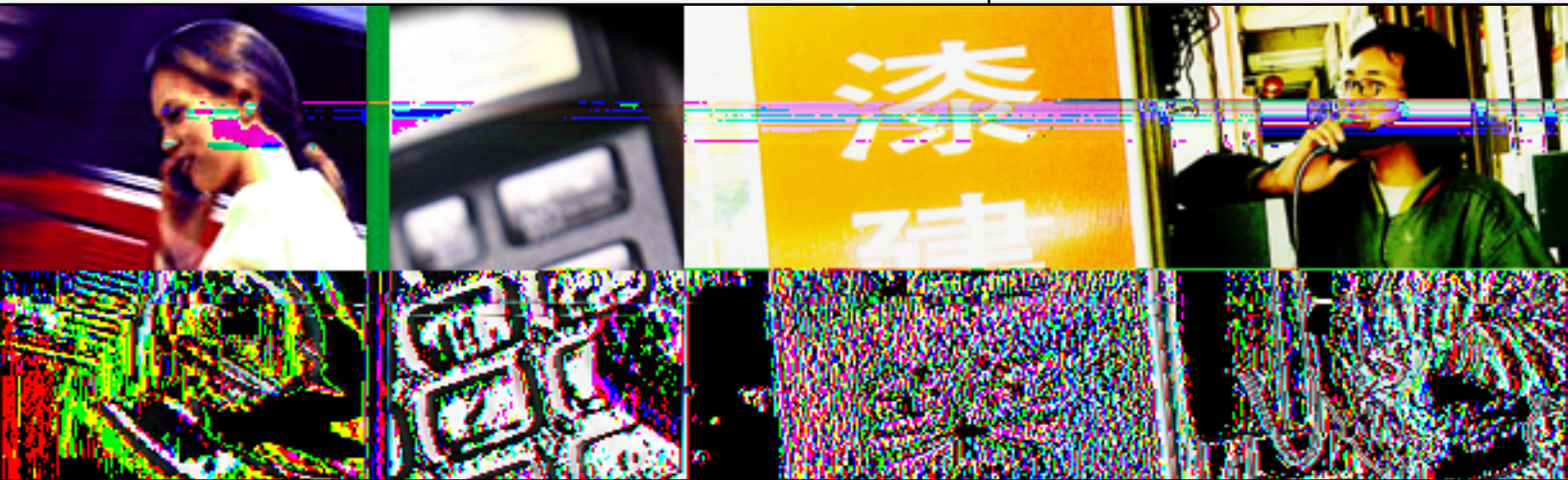


PortaOne Callback



User Guide

Copyright notice & disclaimers

Copyright (c) **2005-2006 PortaOne, Inc.** All rights reserved.

PortaCallback Guide Part I
V.1.10.2, March 2005

Please address your comments and suggestions to: Sales Department,
PortaOne, Inc., Suite 400, 2963 Glen Drive; Coquitlam, BC V3B 2P7
Canada.

Changes may periodically be made to the information in this publication. Such changes will be incorporated in new editions of this guide. The software described in this document is furnished under a license agreement, and may be used or copied only in accordance with the terms of the license agreement. It is against the law to copy the software on any other medium, except as specifically allowed in the license agreement. The licensee may make one copy of the software for backup purposes. No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by electronic, mechanical, photocopied, recorded or any other means, without the prior written permission of PortaOne, Inc.

The software license and limited warranty for the accompanying product are set forth in the information packet supplied with the product, and are incorporated herein by this reference. If you cannot locate the software license, contact your PortaOne representative for a copy.

All product names mentioned in this manual are for identification purposes only, and are either trademarks or registered trademarks of their respective owners.

Table of contents

	Preface	3
	Hardware and software requirements.....	4
1.	System concepts	5
	Callback principle.....	6
	Main components	6
2.	PortaOne callback triggers.....	10
	SMS callback trigger	11
	ANI/DNIS callback trigger	14
	Web trigger.....	16
3.	PortaOne callback engines.....	24
	Cisco gateway-based callback engine.....	25
	PortaSwitch-based callback engine.....	29
4.	FAQ	32
	Can I use ANI callback, when my users from country A call my access number in country B?	33
	Can I use another billing package with your callback modules?	33
	Can I use your SMS callback on CDMA-based wireless networks?	33
	I need to connect several GSM modems to my server, but there is only one serial port – what should I do?	34
	Which Cisco gateway can I use for callback?	34
	When using a Cisco gateway-based callback, can I terminate both calls over IP to my termination carrier?	34
5.	Appendixes	35
	Appendix A – SMS callback commands.....	36
	Appendix B – ANI callback application config parameters.....	38
	Appendix C – Web callback application config parameters	41
	Appendix D – Vendor support for re-INVITE messages	43
	Appendix E - Setting-up a back-to-back T1/E1 connection.....	43
	Appendix F – PortaSwitch callback configuration file.....	45

Preface

This document provides a description of system concepts and available modules for the PortaOne callback solution.

Where to get the latest version of this guide

The hard copy of this guide is updated at major releases only, and does not always contain the latest material on enhancements occurring between minor releases. The online copy of this guide is always up to date, and integrates the latest changes to the product. You can access the latest copy of this guide at www.portaone.com/resources/documentation/

Conventions

This publication uses the following conventions:

- Commands and keywords are in **boldface**
- Terminal sessions, console screens and system file names are displayed in `fixed width font`



Caution means ‘reader beware’. You are capable of doing something that might result in a program malfunction or loss of data.

NOTE: Means ‘reader take note’. Notes contain helpful suggestions or references to materials not contained in this manual.



Timesaver means that you can save time by performing the action described in the paragraph.



Tips are information that might help you to solve a problem.

Hardware and software requirements

Server System Recommendations

There are many different callback modules, but from the hardware point of view they can be split into two categories:

- Modules installed on a Unix server
- Modules installed on a Cisco gateway

Server-based modules can in some cases be installed on one of the PortaSwitch servers, so no extra hardware is necessary. If installed on a separate server, you will need a standard PC-based server, i.e.:

- A minimum of 20 GB of available disk space. This space is required for normal system maintenance and storing various log files. RAID is recommended to improve performance and reliability.
- A processor running at 2.4 GHz or greater.
- At least 512 MB of RAM (RDRAM or DDR), 1 GB recommended.
- If you plan to connect multiple GSM modems to this server, an extra board for multiple RS232 ports.

For information about whether particular hardware is supported by FreeBSD from the JumpStart Installation CD, consult the relevant document on the FreeBSD website:

http://www.freebsd.org/doc/en_US.ISO8859-1/books/faq/hardware.html

1. System concepts

Callback principle

The main idea of callback is that when user A wishes to have a phone conversation with user B, he does not make an outgoing call to B (as he would do with normal telephony service); instead:

- He initiates a callback call (we will discuss the available methods to trigger a callback later);
- The system establishes a call to A, so instead of originating the call A actually answers it (in most telephony networks a subscriber does not pay for incoming calls, so this call is free for A);
- The system establishes a call to B and then bridges the two calls together, so A and B can hear each other.

Main components

The previous section described callback service from an end-user perspective. Now we will look at the internal architecture of such a solution. There are three main components of a callback system:

- Callback trigger
- Callback engine
- Authorization and billing

Callback trigger

The user informs the system of his desire to make a callback call in one of several available ways. The part of the system capable of interacting with the end user in some way and, upon receiving certain information from him, initiating the callback process, is called the callback trigger. Several types of callback triggers are available, and of course new ones can be invented and developed.

ANI (missed call) trigger

This is perhaps the oldest and most common trigger type. The user dials a certain access number, this call is forwarded by his telco operator, and the incoming call is delivered to equipment where a callback trigger application is installed (for example, a Cisco gateway). The application does not answer the call, but remembers the phone number from which the call was made (ANI or CLI number) and then disconnects it. Since the call was never connected, the end user is not charged by the telco operator. Now the application has the user's phone number, and so can initiate a call back to him. Following this an IVR is usually initiated, giving the user his current balance, prompting him for the destination number, and so forth.

Needless to say, this service will only work if the correct caller info is delivered to the gateway. If information about the user's phone number is not available at all during the incoming call, or if it is garbled (for instance, only the four last digits available) then there is no way to call the user back. Usually you can only rely on ANI information for calls made within the same country. Thus if you have an access number in the Czech Republic, and your users make calls from within the Czech Republic (fixed or mobile phones), you will receive correct ANI information. But if someone calls your access number from Slovakia or France, there is a high chance that either no ANI will be provided at all, or else it will be provided in a local format such as 02345464 (no country code), in which case it is impossible to determine whether this number is actually in the Czech Republic, Slovakia or France.

DNIS (DID) callback

This is a slightly different approach to the callback function described above. In a situation where a correct ANI number is not likely to be available, one of the remaining options is to allocate an access number to each of your callback users. Each user will enter his actual phone number (the one he prefers to be called on) into his account's properties. Then, when he wishes to make a callback call, he simply needs to make a call to his access number. Every incoming call to this number will be dropped, but the system will know that it must now originate a call to the phone number associated with this access number. After that, the call is handled similar to an ANI callback.

One definite advantage of this method is that it can be used by callers in one country with an access number in a different country. A disadvantage is that you will need a relatively large amount of access numbers.

SMS callback

Most mobile phones on the market have the ability to send short text (SMS) messages. These messages can also be used to initiate callback. When a user needs to make a callback call, he just sends a message with the relevant information to your access number. This message travels on the wireless network and is then delivered to your callback server. The information in the message is processed and the callback is initiated.

There are several advantages of this method:

- It is possible to include the destination (B) number in the text message. Thus the call can be connected immediately, and the user does not have to enter this information in the IVR.
- It is possible to include other extra information in the text message. For instance, the user can include a password in the message and initiate a call from a friend's phone. Or, he can initiate a call from his mobile phone, but tell the system to connect his hotel phone in Singapore with a destination number in the USA.

- SMS messages have a fixed low cost, so there is no risk of being charged for an outgoing call. Also, while traveling in foreign countries outgoing calls may be disabled unless a special roaming service has been activated, while SMS messages are normally allowed.

Web callback

When a user wants to initiate a callback call, he just goes to your website, enters all the required information (username, password, phone number, etc.) and clicks the Submit button to initiate a call. This request is delivered to the callback trigger running on the web server, and the callback call is initiated.

An advantage of this method is that (similar to SMS callback) the user can enter all the required information in one place. A disadvantage is that the user needs access to the Internet in order to use the service.

Email callback

This is simply a different variety of web callback. Instead of filling in a form in his browser, the user sends an email message to a specific address (e.g. callback@yourdomain.com) with information such as account number, password and destination number inside the message. One small advantage of this method over web callback is that, in some office environments, users are not permitted to browse the Web, but can still send and receive emails.

Callback engine

The component of the VoIP network which actually establishes calls to A and B and bridges them together is known as the callback engine. There are different possible implementations of the callback engine (to be discussed in the section), but all must perform the following functions:

- Include an interface for the callback trigger to transfer the required call information to it;
- Perform call authorization in the billing, to ensure that the user is allowed to make the call and has a sufficient balance to cover it;
- Correctly establish and connect outgoing calls;
- Disconnect calls when a caller has exceeded his credit limit;
- Report call duration and other relevant call information to the billing so the call may be charged.

Billing

Call authorization

Billing must support a type of call authorization which is very specific to callback services. For normal service (e.g. prepaid cards) the question in authorization is: “If account 123 has \$5 available, and tries to call 861234567, is he allowed to talk at all and, if yes, for how long?” In this case, the billing system finds the applicable rate for 861234567 (let us assume it is \$0.10/min, rounded to 60-second increments) and then calculates the allowed maximum duration as $5/0.10=50$ minutes. This task becomes more complicated as the rating formula is more complex, including multiple intervals and special surcharges (a.k.a. billing tricks). With callback, it becomes more complicated still. Now the question is: “If account 123 has \$5 available, and tries to call 861234567 and 42029876543 at the same time, for how long he should be allowed to talk?”. Thus the maximum allowed call duration must be calculated in a such a way that the sum of the charges for the first and the second call do not exceed \$5. Since rates for China and the Czech Republic will most certainly be different, and the rating will include different intervals and surcharges, this task becomes very complicated, and so is not supported by most billing packages. However, PortaBilling100 supports special callback authorization, and can be used to provide authorization for advanced callback services.

Charging a call

After the call has been completed, billing should take appropriate action to calculate call charges, write a CDR into the database, and modify the account’s balance, as well as any other applicable tasks. PortaBilling100 supports all of the callback features described in this manual, and allows you to implement flexible rating for callback services.

2. PortaOne callback triggers

SMS callback trigger

Receiving an SMS message

This module allows a callback call to be initiated upon receiving an SMS message from the user. The most unclear part always is how the SMS message sent from the user's mobile phone will reach the PortaOne server. There are several available options:

GSM modem

It is possible to install a GSM modem and connect it to the PortaOne callback server via a serial interface. When a SIM card is installed in the modem, it can act as a mobile phone (i.e. receive SMS messages), the only difference being that instead of appearing on the phone's LCD these messages will be delivered to the server for further processing.

Advantages:

- Can be used with any SIM card.
- Not dependent on a wireless carrier.

Disadvantages:

- GSM modem and individual SIM cards have a limited performance capacity. Usually you can only receive 6-10 messages per minute.
- GSM modem must be physically connected to the server, so there could be issues with server location (e.g. bad wireless coverage in your collocation center).

Direct connection to wireless carrier

You may contact a carrier directly and discuss an option whereby every incoming SMS message to a certain GSM number will be delivered to your server. The protocol to be used in this case depends on the carrier's capabilities, but will usually be an IP-based protocol (HTTP, SMPP or the like).

Advantages:

- Virtually no limitation on the number of SMS messages per second.
- Stable relationship with wireless carrier.

Disadvantages:

- Wireless carrier may not have this service, or may not be willing to provide it to you.
- Difficulty in dealing with wireless carriers from other countries.
- Interconnection details (e.g. protocol) are not known in advance.

Hosted SMS provider

This method allows you to avoid most of the problems discussed above in the *Direct connection to wireless carrier* section. A company (hosted SMS provider) will take care of interconnecting with wireless carriers around the world, and will then provide you with consolidated SMS flow (using a well-documented API) and a centralized management and billing interface. One good example of such a company is Connection Software, <http://www.csoft.co.uk/>.

Advantages:

- Well-documented API and good management tools.
- Ability to have multiple access numbers (from different countries) and different services from the same vendor.
- Not dependent on a wireless carrier.

Disadvantages:

- Usually the cost of such a service is higher than direct connection to a wireless carrier.

SMS options in PortaOne callback

Currently, the PortaOne SMS callback trigger module contains several components which allow you to receive SMS messages:

1. Using a GSM modem connected directly to the PortaOne callback server. The following GSM modems are currently supported:
 - o WaveCom M1206B (for Europe & Asia GSM networks)
 - o MultiTech MTCBA-G-F2 (for US & Canada GSM networks)
2. Using a connection to a hosted SMS provider. We currently support the following providers:
 - o Connection Software (<http://www.csoft.co.uk>)

If you would like to use:

- a GSM modem not listed above,
- a hosted SMS provider not listed above, or
- a direct connection to a wireless carrier,

please provide us with a detailed description of the interface available to read incoming SMS messages, and we will provide you with a quote for such custom development.

Other features of SMS callback

In addition to using SMS messages to initiate calls, PortaOne callback supports many other features, for instance:

- New user registration via SMS
- Balance top-up via SMS
- Balance check via SMS

All these features are available if PortaBilling100 is used as your back-end billing.

Requirements

An SMS trigger is installed on a PC-based server, typically on the PortaBilling100 slave server. If installation on a separate server is required, this server must be installed from the PortaBilling100 installation CD (as the PortaBilling100 slave server). If using a GSM modem, the server must have a serial port (or ports) so the modem may be connected.

Installation

Since this module requires tight integration with billing for tasks such as new account registration, please contact the PortaOne support team for assistance with installation.

Configuration

There are a few settings which may be changed in the configuration file located in `/home/porta-callback/etc/porta-callback.conf`. Normally, the PortaOne support team will be able to pre-configure the file for you, but you can find a description of the config file syntax in *Appendix F – PortaSwitch callback configuration file* if you need to make changes on your own.

Usage

After the SMS trigger is configured, your customers can send SMS messages to your access number containing any of the commands described in

ANI/DNIS callback trigger

Since this module must handle an actual call traveling on the PSTN network, it must reside on the gateway which processes the call. This module is only available as a TCL script for the Cisco gateway. The task of this module is to:

- Intercept the incoming call and collect information about the ANI number,
- Authenticate this number in the billing,
- Disconnect the incoming call,
- Establish an outgoing call to the destination (the number of the person who called),
- After this call is connected, hand over call processing to some other application (callback engine) which will do the rest (give the balance, ask for destination, etc.).

Once again, this module is merely a callback trigger – in order to provide the service, you need a second TCL application which will serve as a callback engine. This application can be made using a normal debit card application, with only certain modifications.

For DNIS callback, the logic is as follows:

- Intercepts the incoming call and gets information about the DNIS number (access number which was dialed by the customer),
- Authenticates this number in the billing,
- Disconnects the incoming call,
- Establishes an outgoing call to the number specified in the “redirect number” property for the account in the billing,
- After the call is connected, hands over call processing to some other application (callback engine) which will do the rest (give the balance, ask for destination, etc.).

Installation and configuration

Place the script on the TFTP server

After you have obtained the archive containing the application, extract the files from it. Place the script on your TFTP server, making sure that it is accessible via TFTP. PortaBilling100 users should do the following:

- Copy `app_ani_callback.tcl` to the PortaBilling slave server (using secure copy – `scp`) in your user's home directory

- Login to the PortaBilling slave server using ssh. All of the following commands must be executed as a super-user, or via the sudo command.
- Make sure you have a tcl directory on your TFTP server – do
mkdir /tftpboot/tcl
- Move the file into the TFTP directory
mv app_ani_callback.tcl /tftpboot/tcl
- Make sure the file is accessible via TFTP:
\$tftp
tftp> connect <your-server-IP>
tftp> get /tcl/app_ani_callback.tcl
Received 16589 bytes in 6.8 seconds
tftp>

Define an application

Enter configuration mode on your Cisco gateway (conf term). The application is created with the following command:

```
call application voice ani_callback tftp://<SERVER>/tcl/app_ani_callback.tcl
```

where <SERVER> is either the IP or domain name of your TFTP server.

Add the configuration parameters for the application, for instance:

```
call application voice ani_callback authenticate-by ani
```

The most typical configuration will look similar to this:

```
call application voice ani_callback tftp://<SERVER>/tcl/app_ani_callback.tcl
call application voice ani_callback authenticate-by ani
call application voice ani_callback skip-password yes
call application voice ani_callback handoff-to my_debit_app
```

Note the “my_debit_app” configuration parameter. This means that after the call is established control will be passed on to the “my_debit_app” application. Make sure you have defined it in the gateway configuration!

Associate the callback application with one of the incoming dial-peers

Create a dial-peer which will be matched for incoming calls to the access number for ANI callback. On this dial-peer, enter the application ani_callback command so that your dial-peer configuration looks similar to the one below (this is just an example, your actual configuration will look a bit different):

```
dial-peer voice 2 pots
  application ani_callback
  incoming called-number 012345678
  voice-port 2:D
```

Save the configuration

Exit config mode and type wri mem to save your configuration to non-volatile memory.

How to debug

When troubleshooting, be sure to check that:

- The incoming call is delivered to your gateway – check the ISDN debug
- The correct incoming dial-peer is matched – check the CCAPI debug
- The correct application (ANI callback) is launched and there are no errors in the application – check the IVR debug
- The incoming call is authorized – check IVR debug and your billing server logs
- The outgoing call is established successfully – check the CCAPI debug
- The call is handed off to the main application – check the IVR debug
- Correct functioning of the main application – check the IVR debug

To activate the ISDN debug, enter
`debug isdn q931`
in exec mode.

To activate the CCAPI debug, enter
`debug ccapi inout`
in exec mode.

To activate the IVR debug, enter
`debug voip ivr all`
`no debug voip ivr dynamic`
in exec mode

Web trigger

The purpose of this module is to allow end users to initiate callback calls by filling in a form on a web interface. Thus the workflow is as follows:

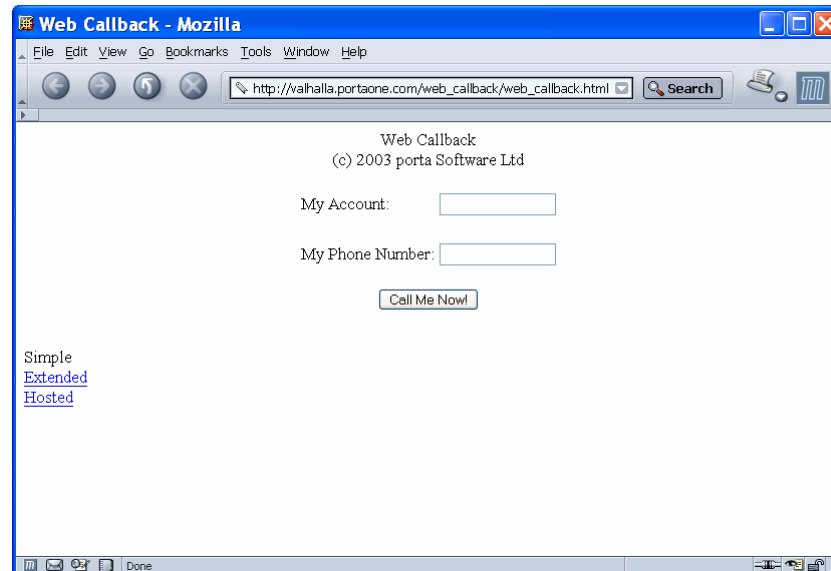
- The customer retrieves the web form, in which he fills in values such as account ID, password, first number, second number, etc.
- By pressing “submit”, the customer’s request goes to the CGI script (hosted on the PortaOne callback server).
- The script processes the input parameters and then passes on the callback request.

You will receive an HTML document containing a form with callback parameters as part of the web callback package. This document is only an example of the required parameters; you can customize it (change text and colors, insert pictures, etc.) or even design this form to be a part of your existing customer portal. However, it is important that all of the form’s parameters and their names be left unchanged.

You may also entirely omit such a web form where users fill in data and press a submit button. In this case, the HTTP request for callback may be sent from another application.

Types of web callback

Simple



- The user provides his account ID (PIN) and the phone number.
- The second number is pre-defined in the application configuration.
- The system checks with billing that the account provided is valid and that it is allowed to establish a call to the given number.
- When the user answers, the second call is established. This provides a convenient way for the user to get connected to the helpdesk or central office.

Extended

The screenshot shows a Mozilla browser window titled 'Web Callback - Mozilla'. The address bar displays 'http://valhalla.portaone.com/web_callback/web_callback_ext.htm'. The page content includes the title 'Web Callback' and copyright '(c) 2003 porta Software Ltd'. There are three input fields: 'My Account:', 'My Phone Number:', and 'Another Phone Number:'. Below these fields is a button labeled 'Connect Us Now!'. On the left side of the page, there are three links: 'Simple', 'Extended' (which is underlined), and 'Hosted'.

- The user provides his account ID (PIN) and two phone numbers.
- The system checks with billing that the account provided is valid and that it is allowed to establish a call to the first number.
- When the user answers, the message: “Wait until you are connected” is played to him, while the second number is dialed.
- When the second call is answered, the two calls are bridged together.

Hosted

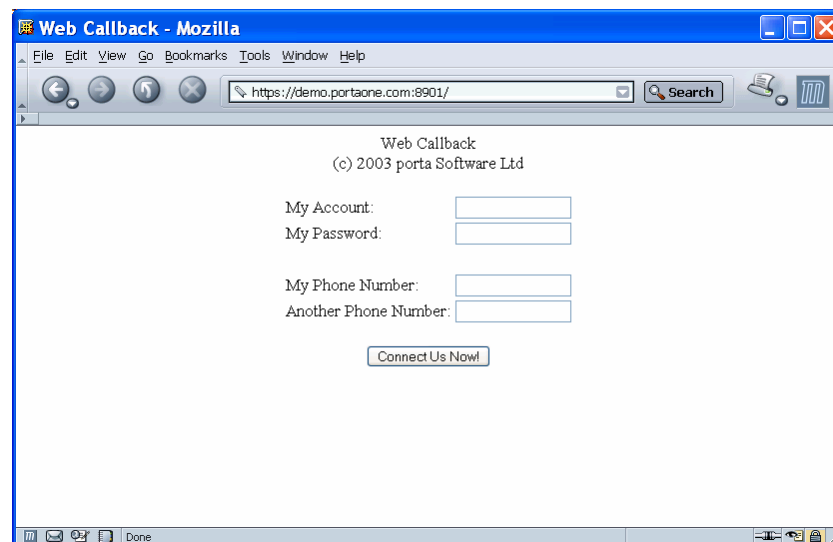
The screenshot shows a Mozilla browser window titled 'Web Callback - Mozilla'. The address bar displays 'http://valhalla.portaone.com/web_callback/web_callback_host.htm'. The page content includes the title 'Web Callback' and copyright '(c) 2003 porta Software Ltd'. There is one input field labeled 'My Phone Number:'. Below this field is a button labeled 'Call Me Now!'. On the left side of the page, there are three links: 'Simple', 'Extended' (which is underlined), and 'Hosted'.

- The user provides the phone number at which he should be called.

- The system does not perform any authentication, but simply establishes an outgoing call to the specified number.
- After the outgoing call is established, control is handed off to another application - not included with PortaOne web callback! This application may report the balance, ask for a destination number, or provide any other functionality (e.g. play a current stock market quote).

NOTE: Since the first outgoing call is established without any authentication, to prevent service abuse make sure that unauthorized users are not allowed to access the web callback trigger for hosted mode.

PortaSwitch-based



The screenshot shows a web browser window titled "Web Callback - Mozilla". The address bar displays "https://demo.portaone.com:8901/". The page content includes the title "Web Callback" and the copyright notice "(c) 2003 porta Software Ltd". Below this, there are four input fields: "My Account:", "My Password:", "My Phone Number:", and "Another Phone Number:". A "Connect Us Now!" button is positioned below the input fields. The browser's status bar at the bottom shows "Done".

This is the only method available for PortaSwitch-based callback, and is very similar to extended mode. The main difference is that the user must provide a password.

- The user provides his account ID (PIN), password and two phone numbers.
- The system checks with billing that the account provided is valid and that it is allowed to establish a call to the given number.
- When the user answers, the second number is dialed.
- When the second call is answered, the two calls are bridged together.

Installation and configuration – PortaSwitch users

Most of the configuration information in this chapter is intended for non-PortaOne users who have purchased a standalone web callback for the Cisco gateway and plan to use it with their custom billing solution.

1. If you are a PortaOne customer and wish to use web callback for the Cisco gateway, contact the PortaOne support team for installation and configuration of the web front-end and proceed to *Installation and configuration*; they will do the configuration for you.
2. If you are a PortaOne customer and wish to use web callback for PortaSwitch, simply contact the PortaOne support team for installation and configuration.

Installation and configuration – others

Extract the web subdirectory from the archive. Here you will find a set of HTML files and a CGI script (in the cgi-bin subdirectory). The simplest way to configure the web trigger is the one described below (an Apache server on Unix is assumed).

Web server configuration

- Copy the web subdirectory to your web server (using secure copy – scp) in your user's home directory
- Login to the web server using ssh. All of the following commands must be executed as a super-user, or via the sudo command.
- Move the directory to its permanent location;
/usr/local/web_callback is assumed in the examples below

```
mv web /usr/local/web_callback
```
- Make sure the CGI scripts are executable

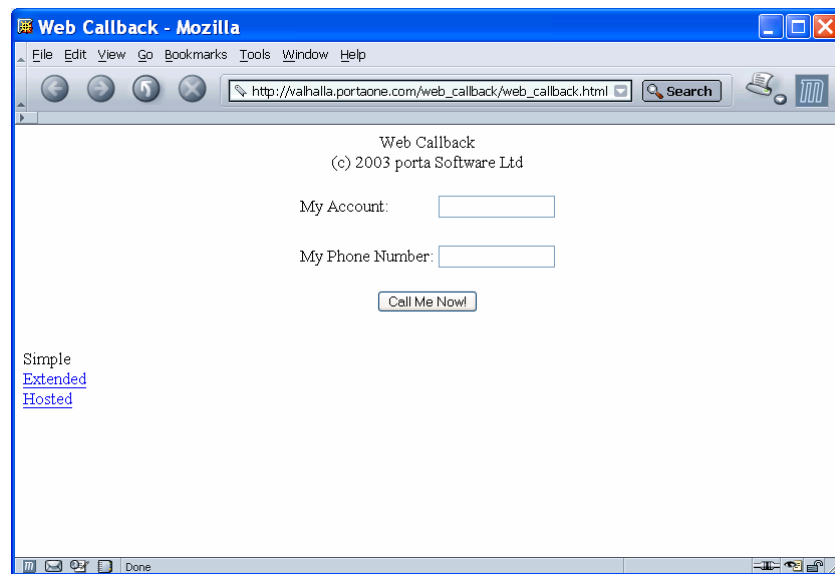
```
chmod 755 /usr/local/web_callback/cgi-bin/*
```
- Include the following section in your Apache config file

```
Alias /web_callback/ /usr/local/web_callback/  
AddHandler cgi-script .cgi  
<Directory /usr/local/web_callback/cgi-bin>  
    Options ExecCGI  
</Directory>
```
- Restart the Apache web server

```
apachectl restart
```
- Now you may open the following URL in your browser:

```
http://<Your-Server-Address>/web_callback/  
web_callback.html
```

You will see a screen similar to the one below:



Note that there are several different web forms available for different types of callback:

- o web_callback.html – simple
- o web_callback_ext.html – extended
- o web_callback_host.html – hosted

Sendmail configuration

The callback trigger will use email messages to deliver a callback request to your Cisco gateway. For this, two or three things are required:

1. Sendmail should be installed on your server. On PortaBilling servers (or any other FreeBSD servers) this is installed by default. For other systems (e.g. Linux) please consult the system administrator guide.
2. Sendmail on your server should be properly configured to send outgoing emails. Please consult the available documentation at www.sendmail.org, FAQs, how-tos and other available information.
3. A certain domain name (e.g. callback.mytelecom.net) should be registered in the DNS and MX (mail exchange records) for this domain point on your Cisco gateway.

For the last item, you should contact your hostmaster or the company that provides your domain hosting. For your convenience, we have provided an example of the appropriate lines to be added to the zone file for your domain if a BIND DNS server is used. Make sure to replace 1.2.3.4 with the **actual IP address** of your Cisco gateway and .mytelecom.net with **your domain name**:

```
callback      IN  A      1.2.3.4
callback      IN  MX     10    callback.mytelecom.net.
```

Application configuration

The last step is to make the callback trigger aware of your environment (for instance, how your Cisco gateway can be reached). Edit the `web_callback.cgi` file using any text editor (if you followed the examples above, this file should now be in the `/usr/local/web_callback/cgi-bin` directory).

1. Check that Perl is installed on your server and the Perl interpreter is available as `/usr/bin/perl`. To check this, type
`/usr/bin/perl -v`
If Perl is installed in a different location, change the first line of the `web_callback.cgi` script so that will look like:
`#!/<full-path-to-your-perl> -w`
2. In the line
`my $host = 'your_domain.com';`
replace `your_domain.com` with the domain name for your Cisco gateway, as follows:
`my $host = 'callback.mytelecom.net';`
3. Check that your `sendmail` binary is installed in `/usr/sbin`. If not, in the line
`my $sendmail = '/usr/sbin/sendmail';`
replace `/usr/sbin/sendmail` with the actual path to the file.
4. Save the file.

Configure your Cisco gateway

Please refer to the *Installation and configuration* section.

Test it

Open the callback web form in your browser, fill in the required parameters, and press Submit. You should see a web page with a “Please wait...” message and, in a few seconds, receive a call. If this does not happen, please consult the *Troubleshooting* section below.

Troubleshooting

- If you press the Submit button, but either see a Perl code text or receive a web server error, then your web server is misconfigured. Please check the Apache configuration file and see the Apache error log for more details.
- If the log files of the web server say something about a Perl error, then you probably made a syntax error when changing the `web_callback.cgi` file. Restore this file from the backup and repeat the steps described in the application configuration. Do not change anything outside the single quotes.
- If you see the “Please wait” screen, but nothing happens, perhaps the request email was not delivered to your Cisco gateway. Check `/var/log/maillog` and see if the message was transferred to your gateway. If not, check your DNS records and `sendmail` configuration.

- If the mail logs indicate that the email message was successfully delivered to your Cisco gateway, then the web callback trigger has completed its job successfully. All further troubleshooting should now be done on the Cisco gateway.

3. PortaOne callback engines

Cisco gateway-based callback engine

This type of callback uses the Cisco gateway's VoIP capabilities, which are controlled by a TCL script.

Pros and cons

Compared to similar software and hardware solutions, this has the following main advantages:

- Using a Cisco TCL, you can develop your own custom IVR for interaction with your customers,
- Reliability of the Cisco gateway,
- No interconnection problems, since most VoIP carriers around the world use Cisco or Cisco-compatible equipment.

However, there are also several disadvantages:

- The total number of simultaneous calls the system can handle is defined by the number of Cisco voice ports available.
- If your service model assumes that both calls in the callback must be made using VoIP, then the total number of such calls a single Cisco gateway can support will be equal to half the number of the available voice ports.

Requirements

Cisco VoIP gateway, IOS 12.3.x (or later).

There are two limitations on the Cisco gateway side:

- Cisco VoIP gateway cannot bridge two VoIP calls together,
- Cisco gateway cannot play prompts in the VoIP leg if a codec different from g711 is used.

Thus, in order to provide normal callback service, the first call leg must be established via the PSTN interface. If all your carriers are VoIP-based, another solution is available: You may use PSTN loopback on your gateway, connecting two E1/T1 ports with a cross-over cable; for more details about this, see *Appendix E - Setting-up a back-to-back T1/E1*. Then one of the calls (usually the first one) will be routed to the local PSTN port, arrive at another PSTN port, and be forwarded for termination via IP.

Installation and configuration

Place the script on the TFTP server

After you have obtained the archive with the application, extract the files from it. Place the script on your TFTP server, making sure that it is accessible via TFTP.

Note for PortaBilling100 users: a TFTP server is already pre-configured (but not enabled, for security purposes) on your PortaBilling slave server. You may enable it yourself in `/etc/inetd.conf` or contact the PortaOne support team for assistance. For all others, the examples below assume that your TFTP root directory is `/tftpboot`.

- Copy `app_web_callback.tcl` and the required voice prompts to the server which will run the TFTP server (using secure copy – scp) in your user's home directory
- Login into the server using ssh. All of the following commands must be executed as a super-user, or via the `sudo` command.
- Make sure you have a `tcl` directory on your TFTP server – do `mkdir /tftpboot/tcl`
- Move the file into the TFTP directory
`mv app_web_callback.tcl /tftpboot/tcl`
- Make sure a directory for voice prompts exists
`mkdir /tftpboot/prompts`
`mkdir /tftpboot/prompts/en`
- Move the voice prompts from
`mv prompts/en/* /tftpboot/prompts/en/`
- Make sure the file is accessible via TFTP
`$tftp`
`tftp> connect <your-server-IP>`
`tftp> get /tcl/app_web_callback.tcl`
Received 16589 bytes in 6.8 seconds
`tftp>`

Define an application

Enter configuration mode on your Cisco gateway (`conf term`). The application is created using the following command:

```
call application voice web_callback tftp://<SERVER>/tcl/app_web_callback.tcl
```

where `<SERVER>` is either the IP or domain name of your TFTP server.

Add the configuration parameters for the application, for instance:

```
call application voice web_callback ext-mode enabled
```

See *Appendix C – Web callback application config parameters* for a list of all available parameters. There are several typical configurations:

```
# Extended mode example
# application definition
call application voice web_callback tftp://<SERVER>/tcl/app_web_callback.tcl
# enable extended mode
call application voice ani_callback ext-mode enabled
# play the specified message to the caller before connecting the second call
call application voice ani_callback on-route-message
tftp://<SERVER>/prompts/en/en_please_wait_male.au
```

```
# Hosted mode example
# application definition
call application voice web_callback2 tftp://<SERVER>/tcl/app_web_callback.tcl
# enable extended mode
call application voice web_callback2 handoff-to my_debit_app
```

Note the “`my_debit_app`” configuration parameter. This means that after the call is established, control will be transferred to the “`my_debit_app`” application. Make sure

you have defined it in the gateway configuration! See the next section for more details.

Configure the hand-off application

This is a separate application required for hosted mode. The functionality of this application is very similar to that of the default debit card application, with a few important implementation differences:

- The application should correctly process the `ev_handoff` event. A normal call application starts with the `ev_setup_indication` event, but since here the call is already established, it will start with `ev_handoff` when the call is handed off by the callback script.
- A normal debit card application assumes that the prompts should be played on the incoming call, and therefore uses the `leg_incoming` constant. In the callback application both call legs are outgoing, and so this should be taken into account.

In order to configure the hand-off application, put the application itself in the `/tcl` directory on your TFTP server, and all the required voice prompts in the `/prompts` directory. Do not forget the language subdirectory, so that prompts for English will be located in `/prompts/en`. The application is created using the following command

```
call application voice my_debit_app tftp://<SERVER>/tcl/app_debit_handoff.tcl
```

where `<SERVER>` is either the IP or domain name of your TFTP server. Assuming your application uses the same parameter settings as the Cisco default debit card application, the following parameters will be required: `pin-len`, `language`, `set-location`. Your application config might look similar to the one below:

```
call application voice my_debit_app tftp://<SERVER>/tcl/app_debit_handoff.tcl
call application voice my_debit_app pin-len 0
call application voice my_debit_app language 1 en
call application voice my_debit_app set-location en 0 tftp://
<SERVER>/prompts/en/
```

Associate the callback application with the multimedia over IP dial-peer

Since the call is triggered not by an incoming call, but by an email received from the web callback trigger, a special dial-peer is required.

- Allow incoming emails to be received by entering the following configuration commands:

```
fax interface-type fax-mail
mta receive aliases <cisco-gw-hostname>
mta receive maximum-recipients 10
```

`<cisco-gw-hostname>` is the hostname you have assigned to your Cisco gateway, and to which email will be delivered (callback.mytelecom.net, in our example).

- Create a multimedia over IP (MMOIP) dial-peer:

```
dial-peer voice 1111 mmoip
description *** Callback Trigger
application web_callback
incoming called-number A.T
information-type fax
```

NOTE: You may use a difference dial-peer number than 1111, as long as it does not overlap with any other existing VoIP or POTS dial-peers.

- Create a POTS dial-peer. Unfortunately, a valid POTS dial-peer matching the MMOIP dial-peer is required, and the incoming called-number must be the same on both. This is why, in the example above, we chose the strange pattern A.T – to make sure it will not interfere with any real PSTN numbers handled by your gateway.

```
dial-peer voice 1112 pots
description *** Mandatory for matching mmoip dial-peer
incoming called-number A.T
no digit-strip
voice-port 0:0
```

NOTE: Replace the voice port ID with a port specification applicable to your gateway. You may use a difference dial-peer number than 1112, as long as it does not overlap with any other existing VoIP or POTS dial-peers.

Configure the required dial-peers to establish an outgoing call

This part of the configuration is totally dependent on your network infrastructure, so we cannot provide any specific examples. Please consult the Cisco manuals for detailed information and see the special requirements for configuration below.

Save the configuration

Exit config mode and type `wr i mem` to save your configuration into non-volatile memory.

Troubleshooting

When troubleshooting, be sure to check that:

- The email is delivered to your gateway – check the mta debug
- The correct incoming dial-peer is matched – check the CCAPI debug
- The correct application (web callback) is launched and there are no errors in the application – check the IVR debug
- The incoming call is authorized – check IVR debug and your billing server logs
- The outgoing call is established successfully – check the CCAPI debug
- The call is handed off to the main application – check the IVR debug
- Correct functioning of the main application – check the IVR debug

To activate the ISDN debug, enter
`debug isdn q931`

```
in exec mode.  
To activate the CCAPI debug, enter  
debug ccapi inout  
in exec mode.  
To activate the IVR debug, enter  
debug voip ivr all  
no debug voip ivr dynamic  
in exec mode.
```

PortaSwitch-based callback engine

PortaSwitch callback engine (PCE) is a specialized SIP Back-to-Back User Agent (B2BUA) which can establish an audio session between two independent SIP endpoints at the request of a third party.

- PCE includes a special application which accepts incoming requests for a callback call from the callback triggers.
- When a new request is received (say for phone numbers 123 and 456), a request to establish the first call leg (123) is sent to the PortaSIP server. Along with the destination number, this request includes the username and password of the account which is trying to establish the call.
- PortaSIP authorizes this call (in a manner similar to the authorization of an outgoing call from a SIP phone). If authorization is successful, it proceeds with the call.
- PortaSIP asks billing for the optimal routing for destination number 123, and establishes the outgoing call to the first destination number.
- Immediately after the first call is connected (i.e. the user picks up the phone), it sends a request to PortaSIP to establish the second call (to 456).
- PortaSIP authorizes this call as well, but in a special way: the billing engine is informed that there are two simultaneous calls, i.e. to 123 and 456. Thus PortaBilling calculates the maximum allowed call duration such that the sum charges for both calls according to the appropriate rates are not greater than the current balance.
- If authorization is successful, PortaSIP obtains the routing for the second destination number and establishes the outgoing call.
- PCE is able to relay progress tones received from the gateway for the second call to the gateway holding the (already established) first call, so the callback originator will be able to hear ringback tones.
- When the user of the second phone number (456) answers, PortaSIP starts a timer, and the call will be disconnected after the maximum allowed time. Information that the call is connected to PCE is sent.

- PCE sends a special re-INVITE request, informing the two gateways (the one terminating the first call and the one terminating the second) that they should send RTP traffic directly between each other.
- The users of phones 123 and 456 can now start their conversation.
- When one of the users hangs up, PCE sends a request to PortaSIP to disconnect the other call as well. The same happens if the second call is disconnected because the time has run out; PCE will request that the first call be disconnected as well.
- PortaSIP sends accounting information to PortaBilling, so that the account is charged for two calls (to 123 and 456). Two CDRs appear in the account's database, and its balance is modified accordingly.

Pros and cons

Compared to similar software and hardware solutions, PCE has the following principal advantages:

- No specialized hardware necessary. PCE can be used on a normal PC-based server, typically the same server on which PortaSIP is installed (it can also be installed on a separate server).
- PCE works with any RFC3261-compliant SIP softswitch / proxy / endpoint.
- Direct media path between endpoints. Once the session is established, RTP audio flows directly between the endpoints, thus providing minimal delay, loss and jitter. For instance, if your callback server is installed in New York, and someone tries to make a callback call between London and Singapore, the RTP stream will go directly between the London gateway of termination partner A to the Singapore gateway of termination partner B. Thus the traffic travels the shortest possible route, and virtually no bandwidth will be used on the New York server side.
- High performance. Since PCE does not relay or process RTP audio, it consumes very little resources (CPU time and network bandwidth) during call establishment and tear-down, and no resources during the call.
- Virtually no limitation on the number of simultaneous calls. Since there is no limitation on your server's bandwidth and performance, you can connect as many calls as your termination partners are able to handle.

However, this architecture also implies certain limitations:

- Remote gateways must support SIP re-INVITE messages (specifically, the ability to change RTP parameters via re-INVITE). You can find information about re-INVITE support among different vendors in *Appendix D – Vendor support for re-INVITE messages*, or contact the vendor directly.

- Because PCE does not interfere with RTP (voice stream), it cannot insert any sounds (IVR prompts, music) into the call or collect input from a user using DTMF; it is only possible to bridge two calls together. Thus you cannot implement services such as “call the user and ask him to enter the destination number” or “the user may press ## to disconnect the current call and enter a new destination number”.

Installation and configuration

Please contact support@portaone.com for assistance with installation.

4. FAQ

Can I use ANI callback, when my users from country A call my access number in country B?

Yes, you can set up this service, however, there is a high chance that the customer's ANI information will not be passed correctly to your gateway – so the system will be unable to determine the customer's callback number. In this case you can offer your customers DNIS callback.

Can I use another billing package with your callback modules?

Yes, you can use Cisco gateway-based ANI or web callback with the third-party billing package. Cisco gateway will communicate with the billing using RADIUS protocol for call authentication and authorization and the billing will charge the calls based on the accounting from the gateway.

Please note that there is no guarantee that your billing will be able to process these calls correctly. The billing may fail to support the special features in combination with Porta Callback scripts, such as different tariffs, based on the access number.

Can I use your SMS callback on CDMA-based wireless networks?

Yes. The wireless protocol used to send SMS messages is irrelevant to PortaOne callback as long as the message is delivered to the server. If you use one of the hosted SMS providers they will deal with the protocol details on their side, and all messages will be delivered to PortaOne callback server in a unified format. If you plan to use a modem directly connected to PortaBilling server, it is also possible, since there are CDMA modems. Just note that in this case the phone number must be pre-programmed in the modem – so it is more difficult to obtain such modem than take a GSM modem and plug in an ordinary SIM card.

I need to connect several GSM modems to my server, but there is only one serial port – what should I do?

You should increase the number of ports in the server. You can install an extra PCI card, which can support 4 or 8 extra serial ports. Here is a list of manufacturers of such cards:

- Moxa www.moxa.com

Which Cisco gateway can I use for callback?

You can use any Cisco VoIP gateway, which supports TCL scripting and is adequately equipped with voice ports and DSPs. The most popular gateway models used for callback are AS5300 or AS5350. For your convenience we provide the model IDs:

Model ID	Description
AS535-2E1-60-AC-V	AS5350, 2 E1s and 60 voice channels
AS535-2T1-48-AC-V	AS5350, 2 T1s and 48 voice channels
AS535-4E1-120-AC-V	AS5350, 4 E1s and 120 voice channels
AS535-4T1-96-AC-V	AS5350, 4 T1s and 96 voice channels
AS535-8E1-210-AC-V	AS5350, 8 E1s and 210 voice channels
AS535-8T1-192-AC-V	AS5350, 8 T1s and 192 voice channels
AS530-2T1-48-AC-V	AS5300, 2 T1s and 48 voice channels

When choosing a gateway, make sure it has the required amount of flash and RAM to run a required IOS version (currently we recommend the latest available release from 12.3 branch)

When using a Cisco gateway-based callback, can I terminate both calls over IP to my termination carrier?

Yes, but only if you use PSTN loopback, so that one of the calls passes through PSTN interface on the gateway.

5. Appendixes

Appendix A – SMS callback commands

Command syntax

- Most of the commands have the syntax **<code> <value>**, where **<code>** is an abbreviated parameter code such as **R** or **DN** and **<value>** is the required parameter. Note that **<code>** and **<value>** must be separated by a space; if more than one **<value>** is to be provided, the individual elements must be separated by a space as well.
- The callback trigger is not case-sensitive, so **DN 420212345** is equal to **dn 420212345**
- Phone numbers should be specified in the E.164 format (country code, followed by area code, followed by the phone number). You should not put 00 or + in front of the phone number; if you do so, the callback trigger will strip them off automatically.

In the description below the following terminology is used:

- **Source phone number** means the phone number which was used to send the SMS message.
- **Registered phone number** means the phone number subscribed to the service, so that there is an account in the billing with such an ID.

Registration of a new account

Format of the SMS message: **R** or **R NNNNNNN** (note the space after R)

Action: The system will take the source number and attempt to create a new account with such an ID (parameters such as product, billing model, account prefix, etc. will be taken from the config file). The new account is created with a zero balance; but if the voucher code NNNNNNN is provided, the new account will immediately be recharged using this voucher.

Following registration, the user will receive an SMS message confirming his subscription and providing him with basic information on use of the service.

If an account with such an ID already exists, and no voucher ID is provided, this command will be ignored; otherwise it is executed as an account recharge.

Examples:

```
R  
R 234873640324
```

Initiate callback call – type A

Connect a registered phone number with the provided destination number.

Format of the SMS message: **DNNNNNNN** or **DN NNNNNNN**

Action: Establish a call to a registered phone number. When the call is connected, establish a call to destination number NNNNNNN and bridge the two calls together. This is the most commonly used service, so it allows easy use of an SMS message: the user just includes the phone number.

Examples:

6581433653

DN 420212345678

Initiate callback call – type B

Connect a provided source phone number with a provided destination number. This service is typically used when SMS messages can be sent from the user's phone (the one which is registered in the system) but he does not wish to receive incoming calls to it. For example, he may be abroad, and so an incoming GSM call would be very expensive for him. Instead, he can instruct the system to call him on a local access number.

Format of the SMS message: **SN XXXXXXXX DN NNNNNNN**

Action: Establish a call to phone number XXXXXXXX. When the call is connected, establish a call to destination number NNNNNNN and bridge the two calls together.

Examples:

SN 6581433653 DN 420212345678

Initiate callback call – type C

It may happen that the user is not able to send SMS messages from his mobile phone (phone not working, prepaid balance exceeded, etc.). In this case, he may use another phone to send an SMS message. In this case, of course, in addition to providing his original phone number he also must provide a valid password, so that no misuse of this service can occur.

Format of SMS message: **P AAAA YYYY DN NNNNNNN SN XXXXXXXX** or **P AAAA YYYY DN NNNNNNN**

Action: Authenticate by account AAAA with password YYYY. If source number XXXXXXXX is not provided, then assume that the first call must be established to the registered number. Establish the first call. When the call is connected, establish a call to destination number NNNNNNN and bridge the two calls together.

Examples:

P 6192345678 1ak45 SN 6581433653 DN 420212345678

P 6192345678 1ak45 DN 420212345678

Check current balance

Format of SMS message: **CB**

Action: If the phone number from which the SMS was sent is registered, send back current balance information by SMS.

Examples:

CB

Change current password

Format of the SMS message: **CP XXXXX YYYYY**

Action: If the old password XXXXX provided is correct, change the password for the account with the registered phone number to YYYYY.

Examples:

CP 1ak45 andrew1

Appendix B – ANI callback application config parameters

Specified as VSA parameters, in the following form:

call application voice <app-name> <parameter> <value>

Authenticate by

Syntax: authenticate-by <param>

Specifies what is to be used as a username for authentication. Possible options are:

- **ani** - Calling-Station-Id (ANI)
- **fixed** - Fixed string, provided as "user-name" parameter
- **dnis** - Called-Station-Id (DNIS), i.e. access number which the customer dialed. When dnis authentication is used, the system calls back to the redirect number associated with this account in the billing.

Default value: ani

Example: call application ani_callback authenticate-by ani

AAA method

Syntax: method-list-name <name>

Specifies which authentication methods should be used. This option may be useful if you wish to use one RADIUS server for callback authentication, and a different server for prepaid card authentication.

Default value: h323

Example: call application ani_callback method-list-name myrad

Include optional information in accounting requests

Syntax: update-accounting <on|off>

Whether script should attempt to update information in accounting requests; aaa accounting update is not available in older IOS, so you might need to turn this feature off if using one.

Default value: on

Example: call application ani_callback update-accounting off

Access number information

Syntax: original-cld <yes|no>

Include information about the access number (the number which the customer dialed to access your callback application) in authentication and accounting requests. This will allow you to use the PortaBilling100 accessibility feature to apply different tariffs based on access number.

Default value: no

Example: call application ani_callback original-cld yes

Default account password

Syntax: password <password>

Password to be used for authentication queries.

Default value: cisco

Example: call application ani_callback password my_secret

Skip password check

Syntax: skip-password yes|no

Instruct PortaBilling100 not to check password during authentication or authorization.

Default value: no (the password "cisco" is used in this case).

Example: call application ani_callback skip-password yes

Delay before calling back

Syntax: wait-time <seconds>

Timeout (in seconds) before the application will call the user's number.

Default value: 30

Example: call application ani_callback wait-time 10

Translate username

Syntax: translate <expr>

Apply translation rules to the username used for authentication. This allows a username to be changed depending on the situation without actually modifying the application source, simply by using CLI. For instance, on your gateway in Prague, Czech Republic, ANI numbers are delivered as 02123456. If you provide services in multiple countries, you will probably enter a Czech account in billing as 4202123456. Now when the gateway receives 02123456, it must add 42 to the beginning of the username, then use 4202123456 for authentication. Similarly, a gateway in London could append 44, and so on.

<expr> is a regexp in the format /pattern/replace-with/. Please note that the regular expression is in TCL regexp syntax; see more details at:

<http://aspn.activestate.com/ASPN/docs/ActiveTcl/tcl/TclCmd/regsub.htm>

and
http://aspn.activestate.com/ASPN/docs/ActiveTcl/tcl/TclCmd/re_syntax.htm.

A few useful examples:

- add 123 to the beginning of the number: `/^./123&/`
- remove 123 from the beginning of the number: `/^123//`
- replace 123 at the beginning of the number with 456: `/^123/456/`

Default value: empty (no translation)

Example: `call application ani_callback translate /^0/420/`

Translate number to be dialed

Syntax: `dial-translate <expr>`

Change the phone number used for callback. For instance, an ANI number is provided to your gateway as 0213456. However, according to the current configuration of your gateway a call to the Czech Republic must be dialed as 004202123456. You can accomplish this using translation rules. See above for more information about the regular expression syntax.

Default value: empty (no translation)

Example: `call application ani_callback dial-translate /^00/`

Fixed username for authentication

Syntax: `user-name <id>`

Specifies a fixed string to be used as the username for authentication.

Default value: empty

Example: `call application ani_callback user-name CB_account`

Caller info for outgoing call

Syntax: `display-info <string>`

If a Caller ID-enabled terminal is used, this string will be displayed as the Caller ID of the incoming call.

Default value: Porta Callback Agent 1.0

Example: `call application ani_callback display-info "PortaOne"`

IVR application

Syntax: `handoff-to <app-name>`

After the outgoing call is established, control will be transferred to this application. Note that the value of this parameter must be the name of an already-created application (using the `call application voice` command).

Default value: empty

Make sure you actually specify this parameter, otherwise the script will establish an outgoing call which will then be immediately disconnected.

Example: `call application ani_callback handoff-to my_debit`

Appendix C – Web callback application config parameters

Specified as VSA parameters, in the following form:
call application voice <app-name> <parameter> <value>

Extended mode

Syntax: ext-mode enabled | disabled

Use extended mode when both phone numbers are provided by the user.

Default value: disabled

Example: call application web_callback ext-mode enabled

Skip password check

Syntax: skip-password yes | no

Instruct PortaBilling100 not to check password during authentication or authorization.

Default value: no (the password "cisco" is used in this case).

Example: call application web_callback skip-password yes

Append a prefix to the first dialed number

Syntax: first-prefix <prefix>

To make sure that the first call goes out via a specific dial-peer (e.g. to PSTN loopback) the application may dial the number with a certain prefix. For instance, say you normally terminate calls to the Czech Republic (420) to remote gateway 1.2.3.4. However, the first call leg of the callback call must go via PSTN loopback (connecting ISDN ports 0 and 1), and only then to the remote gateway.

Thus the application will dial the number as 888#4202123456, and the dial-peer with destination pattern 888# will route this call to ISDN port 0. The call will arrive to ISDN port 1 in the 'normal' format (420123456) and will then be sent to the remote gateway via your normal VoIP dial-peers.

Default value: empty (no prefix appended to the number)

Example: call application web_callback first-prefix 8881#

Append a prefix to the second dialed number

Syntax: second-prefix <prefix>

You may also append a prefix to the second dialed number, so as to select different dial-peers.

Default value: empty (no prefix appended to the number)

Example: call application web_callback second-prefix 8882#

After the first call is established, transfer call control to another application (PSTN)

Syntax: pstn-handoff-to <application>

The TCL application to which the call is handed off if the first call was established via PSTN. It is possible to hand off to a different application, depending on whether the first call was established via a PSTN interface or IP. (For instance, an application on the PSTN leg will be able to play prompts, while one on VoIP probably will not, because of codec limitations).

Default value: empty

Example: call application web_callback pstn-handoff-to debit1

After the first call is established, transfer call control to another application (VoIP)

Syntax: voip-handoff-to <application>

The TCL application to which the call is handed off if the first call was established via IP. If you want to use the same application regardless of whether the call was established using PSTN or VoIP, specify two parameters with the same value.

Default value: empty

Example: call application web_callback voip-handoff-to debit2

Remove a certain prefix from the destination number provided

Syntax: strip-prefix <prefix>

Typically, to avoid collision between your callback dial-peer and dial-peers for normal incoming calls, the phone number will be sent to the gateway prefixed by a string that will make it fundamentally different from PSTN numbers. We suggest using an A character, which is a valid E.164 number element, but is not used in normal PSTN dialing.

When using such a prefixed number (e.g. 4202123456 will be transformed into A4202123456), the callback application must be instructed to remove a certain portion of the number (A in our case).

Default value: A (so the application will remove A from the beginning of the number). Thus even if you configure your CGI application not to use a prefix, this should not create any problems, and the number will stay unchanged.

Example: call application web_callback strip-prefix A

Message to be played to the user while waiting for the second call to be connected

Syntax: on-route-message <URL>

Pre-recorded voice prompt to be played to the first party while waiting to be connected. This will only work if:

- the call is established via PSTN, or
- the call is established via IP and the g711 codec is used

Default value: empty (no prompt is played)

Example: call application web_callback on-route-message
tftp://1.2.3.4/prompts/en/ en_please_wait_male.au

Default phone number for the second call

Syntax: connect-to <phone-number>

For simple mode, the phone number to which the second call should be connected. This may be used in extended mode as well; for instance, you may specify your helpdesk number here. If there is a problem and the customer has not provided the second phone number on the web form, he will be connected to your technical support line instead.

Default value: empty

Example: call application web_callback connect-to 18661234567

Appendix D – Vendor support for re-INVITE messages

Currently we have confirmed that the following gateways work correctly with PortaSwitch callback:

- Cisco gateways (IOS 12.3 or later),
- Quintum gateways (firmware P101-14-10 or later)
- Mera Networks SIP-HIT gateways

Appendix E - Setting-up a back-to-back T1/E1 connection

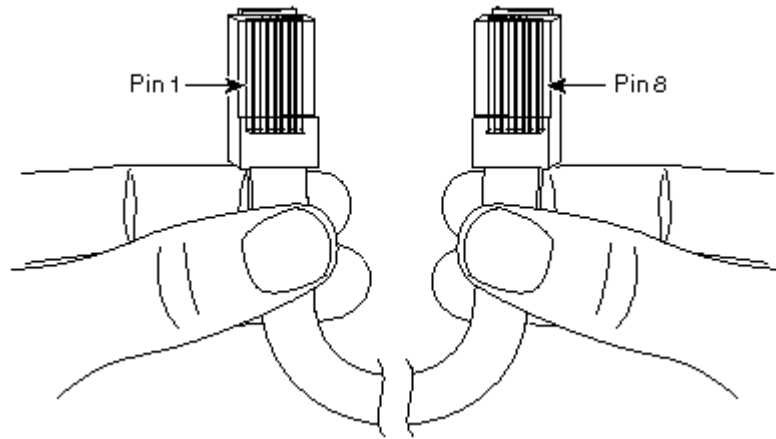
Hardware Setup

In order to use a Cisco voice gateway (such as AS5300/5350) for handling VoIP-VoIP calls, you need to physically loop one or more pairs of T1 or E1 voice ports on it so that these can be used for the PSTN→PSTN connection. To do this, construct one or more RJ-48C cross-over cables using the following table:

T1/E1 CSU/DSU Cross-Over Pinout

From RJ 48C Pin	To RJ 48C Pin
1	4
2	5
4	1
5	2

Make sure you count the RJ-48C pins as shown in the illustration below:



Alternatively, you can order readymade ones. You can find a number of vendors producing such cables by searching for “RJ-48C cross-over cable” on www.google.com.

Once the cable is ready, plug it into the designated pair of T1/E1 ports in your Cisco AS5300 gateway.

Software Configuration

You also have to configure the T1/E1 interfaces. The sample configuration below is for T1; adjust the time slots for E1:

```
isdn switch-type primary-5ess
!
controller T1 0
framing sf
clock source line primary
linecode ami
pri-group timeslots 1-24
!
controller T1 1
framing sf
clock source line secondary 1
linecode ami
pri-group timeslots 1-24
!
controller T1 2
framing sf
linecode ami
pri-group timeslots 1-24
!
controller T1 3
framing sf
linecode ami
pri-group timeslots 1-24
!
interface Serial0:23
no ip address
isdn switch-type primary-5ess
isdn protocol-emulate network
```

```

no cdp enable
!
interface Serial1:23
no ip address
isdn switch-type primary-5ess
no cdp enable
!
interface Serial2:23
no ip address
isdn switch-type primary-5ess
isdn protocol-emulate network
no cdp enable
!
interface Serial3:23
no ip address
isdn switch-type primary-5ess
no cdp enable

```

Appendix F – PortaSwitch callback configuration file

SMS provider section

Here you may choose how SMS messages are received

```

[SMS-Provider]
#
# SMS Provider's name.
# Currently supported providers are: Csoft and Modem
#
name=Modem
#
# Type of callback engine to be used for SMS callback
# Available options are SIP and GW, meaning use of
# PortaSwitch or Cisco GW, respectively
SessionInitiationMethod=SIP

```

SMS subscription section

You may provide different products and services via SMS callback. PortaSwitch callback allows you to associate different customers or products with different GSM access numbers. Thus users who register via your UK GSM number could get an “SMS-UK” product (charged in pounds), while users subscribing to a Czech GSM number would get a “Czech-SMS-Callback” product charged in Czech crowns. First of all, you must describe the different access numbers you have and assign a config section for each one of them, as follows:

```

[Customer-Links]
#access number=name of customer section
#access number - mobile number to which the messages were sent.
44812345678=UK-Main
44812345679=UK-ResellerA
420602123456=CZ-Main

```

The configuration section above defines three different access numbers. A detailed configuration of services provided on these numbers will be given in the respective configuration sections, for instance:

```
[UK-Maint]
# Customer who will own the new accounts. The value must
# match the name of the existing customer (or reseller)
Customer=MyCompany-UK
# The product to be assigned to new accounts
Product=SMS-UK
# The batch the accounts will be grouped under
Batch=sms-callback-uk
# Account type: Debit = -1 Credit = 1
Billing_model=-1
# The batch the accounts will be grouped under
Life_time=30
# Opening balance for the new accounts. Should be zero
# unless you are running a special promotion.
Balance=0
```