Table of contents

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

Appendix A – SMS callback commands.

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.
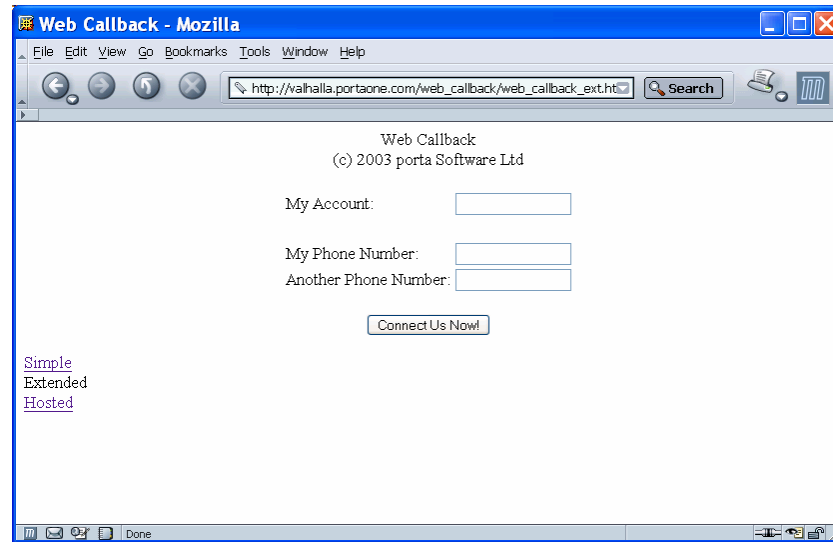
16

Types of web callback

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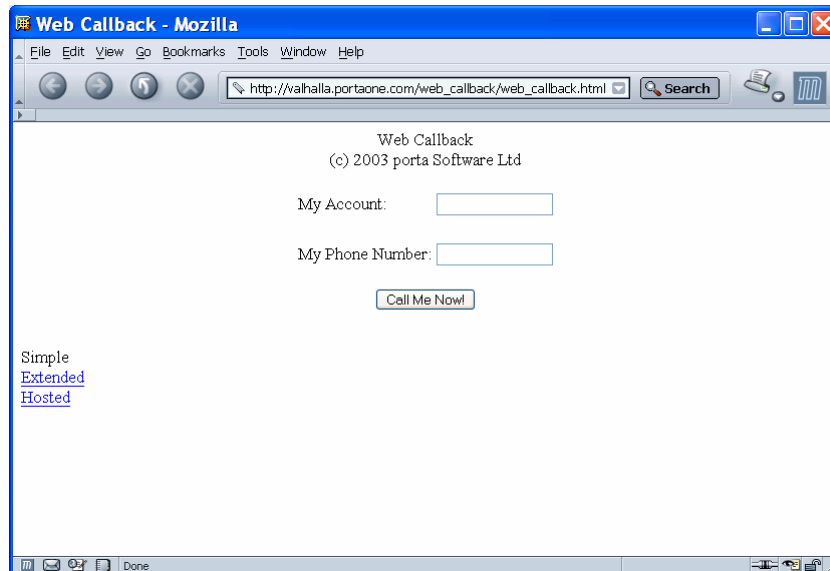
- The user provides his account ID (PIN) and the phone number.
- The second number is pre-defined in the application configuration.
- The system checks with billing that the account provided is valid and that it is allowed to establish a call to the given number.
- When the user answers, the second call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

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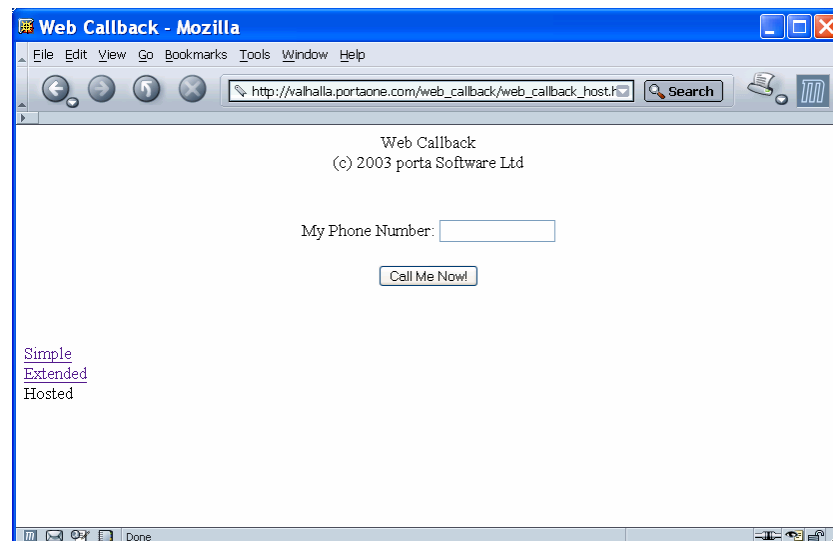
- The user provides his account ID (PIN) and two phone numbers.
- The system checks with billing that the account provided is valid and that it is allowed to establish a call to the first number.
- When the user answers, the message: "Wait until you are connected" is played to him, while the second number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

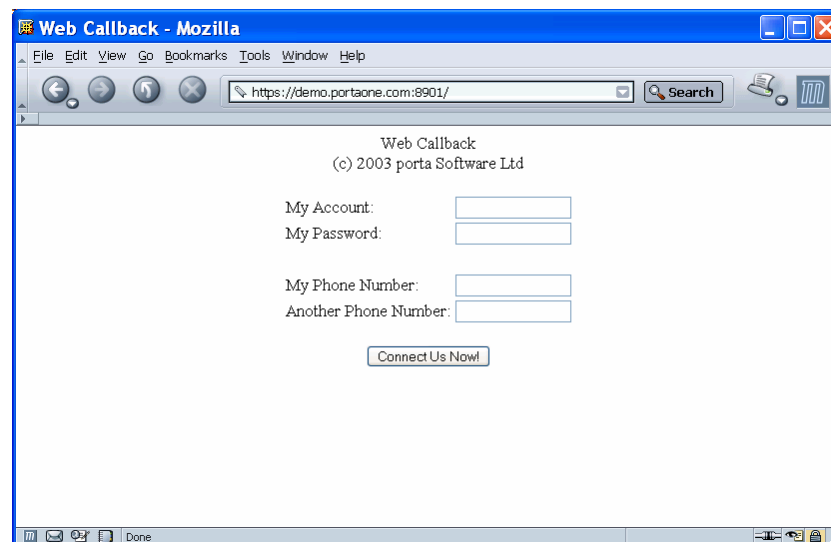### Hosted

- The 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

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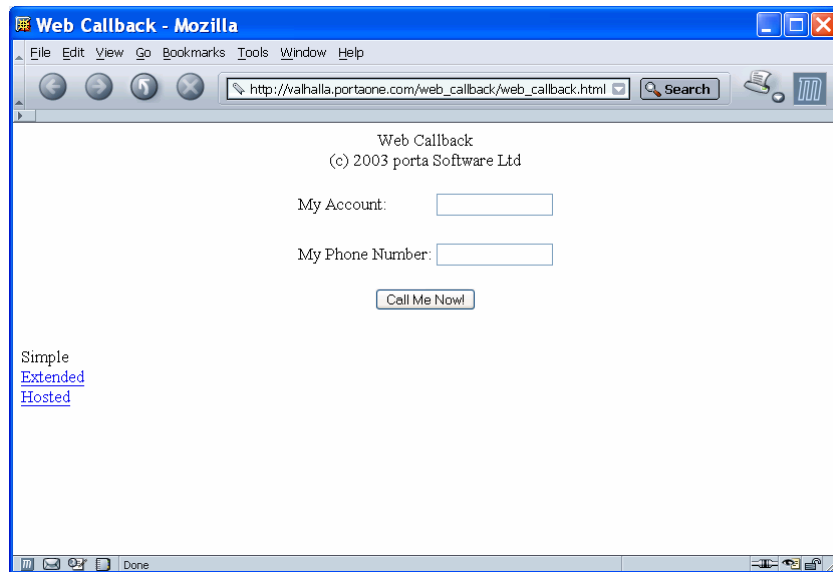
Note that there are several different web forms available for different types of callback:

- 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