# call-back mini HOWTO

# Table of Contents

# call-back mini HOWTO

**by Pawel Skonecki, stona@kft.umcs.lublin.pl**

v2.1a, 2001-10-06

---

*This document describes how to set up call-back by using the Linux system and modem. I would like to thank Anna for her patience.*

---

# 1. Introduction

## 1.1 OPINION

I will be waiting for all opinions about this document. I have tried to gather information as complete as possible. Tell me when your find any mistakes. I'll be grateful to people who will send me any suggestions or corrections. Their contributions will make this document better. I don't mind answering your questions but I'd rather you read the whole article first.

## 1.2 PUBLISHING

This document can by published under the conditions of Linux Documentation Project. Get in touch with the author if you can't get this license. This document is free.

# 2. Procedure

## 2.1 PART I: Net at home ?

Most of us use the Internet in a place of work. However we offen need the net at home or outside the place of work. It may be possible that the work from home is cheaper then from a company building. I think that the best solution is to install call-back software on the Linux server. Call-back makes it possible to re-call index number at the cost of the company. I'll try to present how it works. An entitled person who calls the modem is varied for the first time in Linux server. Then on the user's side the "hang up modem" is switched on. At the same time Linux calls the user. User is verified again. We have connection and the server is charged. The user pays only for the initiation of the connection. The double verification and extra options in the call-back program unable the unsuitable persons to charge our bill. We can restrict the access to the connection only to corporation network or the Internet. Call-back is very flexible. Below, I'll try to present the configuration of a call-back server on Linux system and I'll show you how to set up your computer for re-calling the connection. I don't describe configuration of ISDN call-back because I don't use ISDN in my connection to the Internet. If you set call-back on ISDN send me your configuration. I had some problem when I changed my kernel from 2.2.x to 2.4.x. I will describe new option for a new kernel. Remember that if you change your kernel for high version you will have to change pppd, too. I don't create a new section for description of new possibility in kernels 2.4.x but I write a new configurations in old section. I would like to apologise to people who asked me about options for new kernels. I didn't have time to write a new version of HOW-TO. I have changed my job and place of liveing. Sorry.

## 2.2 PART II: The first steps with modem.

The administrators prefer different modem but while buying a modem we should remember certain guidelines:

- If you have USR WinModems see Linmodem-HOWTO.
- The externale modem is moust flexible you have more place inste your computer.
- The internal modem with ISA slot is better then the are with PCI slot (you can use your PCI slot for something different )
- Don't use Plug&Play modem see Plag-and-Play-HOWTO.

When we have the suitable modem we have to set it up in our system. We have to check on which com our modem is. Then we have to make a symbolic link to this hardware and /dev/modem. For example, if we have the modem for the 2nd com we write:

```
ln -s /dev/cua1 /dev/modem
```

We check it

```
lrwxrwxrwx 1 root uucp 9 Sep 19 19:10 /dev/modem -> /dev/cua1
```

If we have the modem on different com we have to remember that

```
/dev/cua0 is com1

/dev/cua1 is com2

/dev/cua2 is com3

/dev/cua3 is com4
```

For new kernels:

```
/dev/ttyS0 is com1

/dev/ttyS1 is com2

/dev/ttyS2 is com3

/dev/ttyS3 is com4
```

Now, we check our configuration using the program minicom.

## 2.3 PART III Call Linux

The first step to make the call-back on Linux accessible is to set up a suitable parameters in kernel. Then we check whether our kernel serves the protocol ppp. If you don't have ppp in your kernel or in module you will have to compile your kernel and add ppp. You will find more information in Kernel-HOWTO.In the kernels 2.4.x series you have to mark follow options:

CONFIG_PPP=m # CONFIG_PPP_MULTILINK is not set CONFIG_PPP_ASYNC=m CONFIG_PPP_SYNC_TTY=m CONFIG_PPP_DEFLATE=m CONFIG_PPP_BSDCOMP=m

After compilation you have to add some lines for /etc/modules.conf

alias /dev/ppp ppp_generic alias char-major-108 ppp_generic alias tty-ldisc-3 ppp_async alias tty-ldisc-14 ppp_synctty alias ppp-compress-21 bsd_comp alias ppp-compress-24 ppp_deflate alias ppp-compress-26 ppp_deflate

You can't forget that you need new pppd demon for kernels 2.4.x (for me it was ppp-2.4.0 this ppp has some mistake you have to get latest from ftp.samba.org in /pub/ppp ).

OK. We have a good kernel. Now, we have to set up the software to our system. The call-back program is a part of mgetty-sendfax and ppp. You will find it all in your distribution. Because the call-back system has double verification we create a user who will be running ppp on the side of server. In /etc/passwd you have new user and you have to change their shall.

```
pppuser:klkIOM89mn65H:230:PPP Dialin:/home/pppuser:/etc/ppp/ppplogin
```

I changed the above line for kernel 2.4.x this line in /etc/passwd (I use shadow and you don't see password)

```
pppuser:x:6778:44:PPP Dialin:/etc/ppp/:/usr/sbin/pppd
```

I don't use a special script for the running of pppd but I run it directly while login pppuser.

Then change the password. We have to add information abut password in the file `/etc/ppp/pap-secrets` (more in man pppd)

```
pppuser * password_for_pppuser *
```

In the 2.4.x kernels you have to write in `/etc/ppp/pap-secrets`

```
*                *        ""                        *
```

This user doesn't have a usual shell but a file `/etc/ppp/ppplogin`. We have to make it ourselves. For example `vi /etc/ppp/ppplogin` and we type:

```
#!/bin/sh

exec /usr/sbin/pppd -detach 192.168.1.1:192.168.1.2
```

where the address 192.168.1.1 is the address of server with modem and the address 192.168.1.2 is the address which we assigned to our modem. We set up executable options for this file. Because we will use the ppp demon we have to set up the options for this demon. We edit file `/etc/ppp/options`:

```
proxyarp

lock

crtscts

modem
```

If it's 2.4.x kernel you write in `/etc/ppp/options`

```
  -detach
  asyncmap 0
  modem
  crtscts
  proxyarp
  lock
  require-pap
  refuse-chap
  ms-dns 192.168.1.1
  usepeerdns
```

The last 3rd option is very important. You use only PAP authentication ,require-pap. Don't use chap authentication, refuse-chap. You can use ms-dns, If you have M$ Windows system clients you can send them information about DNS server. If you wont to send IP of DNS server for Linux/UNIX machine you use usepeerdns option, more in man pppd.

Proxyarp is the most important from the above options, because you can go to Internet by the modem in the server. The remaining options are used to control your modem. You can only work on the server if you remove proxyarp option. You have to see PPP-HOWTO and man pppd for more information. We will set up our modem now. Our server must be ready to receive a connection after start. We edit file `/etc/inittab` and we add it's to modem on the 2en com.

```
s1:2345:respawn:/sbin/mgetty ttyS1 -D /dev/ttyS1 vt100
```

or

```
s1:2345:respawn:/sbin/mgetty ttyS1 -s 115200 -D /dev/ttyS1
```

For the 1st com line looks as follows:

```
s0:2345:respawn:/sbin/mgetty ttyS1 -D /dev/ttyS1 vt100
```

or

```
s0:2345:respawn:/sbin/mgetty ttyS0 -s 115200 -D /dev/ttyS0
```

We make `init q`. If we don't have information about any mistakes in logs we go to the next step. We come back to directory `/etc/ppp` and create `options.ttyS1` (for modem com1 `options.ttyS0`)

```
IP_local: IP_remote
```

for our net it will be

```
192.168.1.1:192.168.1.2
```

We have done a lot work so far. Now, we check the file `/etc/mgetty+sendfax/login.config`. The most important line is:

```
/AutoPPP/ - a_ppp /usr/sbin/pppd auth -chap +pap login detach 7 debug
```

If you have 2.4.x kernel you have write in this file:

```
 /AutoPPP/ -     a_ppp   /usr/sbin/pppd file /etc/ppp/options
```

The remaining lines can be marked `#`.

We have to set up suid for ppp demon, because pppuser has to run pppd and make interface work. I insert have to insert opinion of Bill Staehle "NOTE: Some distributions think they know better than you, and delete this permission when their 'admin' tool (yast in SuSE, linuxconf in many others) is used. Read the documentation of the "tool" to see how to stop this crap."

```
chmod u+s /usr/sbin/pppd
```

and its effect is:

```
-rwsr-xr-x 1 root root 106892 Jan 11 1999 /usr/sbin/pppd
```

I think that it is a good idea is to add it to cron becouse I had a problem after restart of my server pppd changed preference. It's normal situation in. I add opinion of Bill Staehle "COMMENT: This is what I was referring to. In Red Hat, look at /usr/lib/linuxconf/redhat/perm/ppp (delete or rename the file)."

Our server will work as router. We have to enable IP forwarding and we add this line for the file `/etc/rc.d/rc.local`:

```
echo "1" > /proc/sys/net/ipv4/ip_forward
```

If you are RedHat user you can change in `/etc/sysconfig/network` from `FORWARD_IPV4=false` to `FORWARD_IPV4=true`.

For verification we call to Linux. We use scripts for it. If we do this in MS Windows we mark options "`call out a terminal after connection`". We login as pppuser with its password. I hope that all is OK.

## 2.4 PART IV Linux calls us

We can already call our Linux. Now it's time Linux called us. It's not very diffucalt. We have to edit only two files. We create a file `/etc/mgetty+sendfax/callback.conf` and we leave it empty.

Then we have to ask our users for their phone number. It's time to write the numbers we have connected earlier. In order to do it we edit `/etc/mgetty+sendfax/login.conf` and add line:

```
call - - /usr/sbin/callback -S 123456
```

where call is a pseudo-user needed to initiate the connection. The line in the `/etc/mgetty+sendfax/login.conf` puts in motion the program calling the given number (in this case it's 123456). The same procedures can be applied to other users. I'll try to explain how it works. When we call a server. It asks us to give verification. We login as pseudo-user, in this case it calls. The script in our computer hangs up the modem. We wait and the connection is cut off. The program call-back starts working and recalls us. We verify ourselves again as pppuser with password. We combine the connection and interface ppp. That's all. The configuration of work-stations is very simple. When you have MS Windows, you have to install dial-up for your number. In the modem propriety we find " propriety--->extended---> extra options" where we write.

```
&c0s0=1
```

We close the window and call. We log in according to the description given above. If we want to use Linux, we must refer to the script. It's difficult to give only one good script for our Linux. A good configuration of ppp in the system is of primary importance. (You can call it as pppuser through the scripts first).The scripts below were whiten by A. Gozdz. I suggest putting everything to catalogue. It is only my suggestion & you don't have to start the scripts here. Detailed information cocernig writing scripts on Linux can found in PPP-HOWTO.

The configuration file of daemon ppp (an example for modem on com2) **THESE SCRIPTS WORK GOOD WITH LINUX RED HAT 6.x**

- /etc/ppp/options

```
lock

defaultroute

noipdefault

modem

115200

crtscts

debug

passive

asyncmap 0
```

- /etc/ppp/pppcallback

```
TIMEOUT 5

ABORT 'ERROR'

ABORT 'BUSY'

ABORT 'NO ANSWER'

ABORT 'NO DIALTONE'

ABORT '\nVOICE\r'

ABORT '\nRINGING\r\n\r\nRINGING\r'

'' AT&FH0 'OK-+++\c-OK' 'AT&C0S0=1'

TIMEOUT 40

OK ATDT5376443 CONNECT ''

ogin:-ogin: ppp-pseudo-user

'\nNO CARRIER\r' ''

TIMEOUT 180

'\nRING\r' AT&C1A
```

```
CONNECT ''

TIMEOUT 20

ogin:-ogin: pppuser

sword:-sword password_for_ppuser
```

- /usr/bin/ppp-call

```
#!/bin/bash

teksta="Connection failed"

tekstb="Probably, You will be connect"

# /sbin/setserial /dev/ttyS1 spd_vhi

killall -INT pppd 2>/dev/null

rm -f /var/lock/LCK* /var/run/ppp*.pid

(/usr/sbin/pppd -detach call ppp_call &) || \

(echo $teksta; ls marsss >/dev/null; exit 1)

echo $tekstb

exit 0
```

- You can run ppp-call, now. :)

If you have M$ Windows you can use this script for conection. I don't test it (I use terminal) you can ask some more Adrian Debkowski ( adrian@cr-media.pl).

```
proc main

delay 1

waitfor "ogin:"

transmit "call^M"

waitfor "RING"

transmit "ATA^M"

waitfor "CONNECT"

waitfor "ogin:"

transmit "pppuser^M"

waitfor "word:"

transmit "ppp^M"

endproc
```

## 2.5 PART V Summary

The Configuration of call-back is not complicated. The most important thing is a proper arrangement of ppp server on Linux. I don't know a better way of setting up an access - server. The configuration presented above is a result of numerous attempts and it can be done in a different way. That's way I suggest reading all the documents concerning the issue man pppd, NET-HOWTO, PPP-HOWTO,ISP-Setup-RedHat-HOWTO,Modem-HOWTO . A special thanks for Bill Staehle.