# INSTALL PLAYSMS AND SMSTOOLS ON DEBIAN 10/11/12 and Ubunru 20/22/24

install other sources to install php 7.2

**apt -y install gnupg2 apt-transport-https ca-certificates software-properties-common**

**wget -O /etc/apt/trusted.gpg.d/php.gpg https://packages.sury.org/php/apt.gpg**

**echo "deb https://packages.sury.org/php/ $(lsb_release -sc) main" > /etc/apt/sources.list.d/php.list**

**apt update && apt install -y apache2 mariadb-server php7.2 php7.2-opcache php7.2-cli php7.2-mysqli php7.2-mysql php7.2-gd php7.2-mbstring php7.2-xml php7.2-curl php7.2-zip php7.2-fpm libapache2-mod-php7.2**

**update-alternatives --set php /usr/bin/php7.2**

**sudo a2enmod mpm_prefork && sudo a2enmod php7.2**

**update-alternatives --set php /usr/bin/php7.2**

**sudo service apache2 restart**

**********

while if you are on ubuntu 18.04 only:

**apt-get install apache2 mariadb-server php php-cli php-mysql php-gd php-curl php-mbstring php-xml php-zip**

**********

## Install playSMS 1.4.6 and smstools 3 (compiling it)

# 1. Prepare Ubuntu

## 1.1. Add Normal User

In DO you need to login as root first. But it is recommended to not login as root all the time, so we create a new normal Linux user.

As root create a new normal Linux user and set a strong password for it:

**adduser playsms**

**Add user playsms to sudo group:**

**usermod -a -G sudo playsms**

## 1.2. Copy authorized_keys

This is additional and optional steps you need to do if you're login SSH as root using private key instead of password.

You need to copy root's authorized_keys to playsms:

**sudo mkdir -p /home/playsms/.ssh**

**sudo cp /root/.ssh/authorized_keys /home/playsms/.ssh/**

**sudo chown -R playsms.playsms /home/playsms**

**After this you can login SSH as user playsms using the same private key as root.**

**(you could  have errors on Debian releases cos they don't have public certs, like Ubuntu)**

## 1.3. Enable Ubuntu Firewall

Allow SSH first:

**sudo ufw allow ssh**

Enable UFW, activate it and make it starts on boot:

**sudo ufw enable**

Reload UFW:

**sudo ufw reload**

As of now only SSH allowed by server, later we will allow http and https. Don't forget to ufw reload after changing UFW rules.

# 1.4. Install mc , zip and unzip

Yes. Install mc and unzip :) I'm using nano as console text editor, and you might be checking files/folders frequently, for that I think mc helps. But you can always choose not to install it and stick with nano or vi.

You need to install unzip, composer will need it and playSMS will need composer.

Install mc , zip and unzip:

**sudo apt update**

**sudo apt install mc zip unzip**

# 1.5. Upgrade Server

Update and upgrade:

**sudo apt update**

**sudo apt upgrade**

Most likely after upgrade Ubuntu asks for server reboot, reboot it then:

**sudo shutdown -r now**

Re-login SSH using user playsms instead of root. Pass this point you need to login to the server as normal user playsms, and you will use sudo when you need to

**execute commands as root**.

# 2. Install MySQL Server

We will use MariaDB as MySQL server.

If you have not logout out from root you need to logout now and re-login as normal user playsms.

Install MySQL server MariaDB:

**sudo apt install mariadb-server**

**Starts MariaDB and enable it:**

**sudo systemctl start mariadb.service**

**sudo systemctl enable mariadb.service**

**Test your MySQL root access:**

**sudo mysql**

You should now logged in to your MySQL server as MySQL user root. Type quit and **<Enter>** to exit MySQL console.

Note that you cannot login to MariaDB as MySQL user root if you are not Linux user root. Use sudo to access MySQL server as MySQL user root, you won't be asked for password.

We will not use MySQL user root in playSMS but we will create a new MySQL user just for playSMS database later.

## Start Apache2 and enable it:

**sudo systemctl start apache2.service**

**sudo systemctl enable apache2.service**

**a2enconf php7.2-fpm**

**systemctl restart php7.2-fpm**

## Allow HTTP and HTTPS:

**sudo ufw allow http**

**sudo ufw allow https**

**sudo ufw reload**

## Let’s test the PHP:

**cd /var/www/html**

**sudo nano test.php**

**<?php**

**echo "Hello World";**

Save test.php and browse the file, you should *Hello World* displayed.

Remove `test.php` after testing:

**sudo rm -f /var/www/html/test.php**

# 4. Supports HTTPS

HTTPS supports will be added to our web server by requesting, installing and configuring SSL certificate from Let’s Encrypt on Apache2. Let’s Encrypt provides a free SSL certificate for everyone.

## 4.1. Setup VirtualHost

This step is required for getting free SSL certificate for our HTTPS service from Let’s Encrypt.

In this example I will be using dm143.playsms.org domain as my entry in VirtualHost setup. I also have set the DNS to point dm143.playsms.org to my CentOS server’s public IP. Of course you will need your own domain/subdomain and point to your own CentOS server’s public IP.

The example VirtualHost configuration will make Apache serve PHP file for domain playsms from our regular user (user playsms) Home Directory (**/home/playsms/public_html** to be exact).

## Prepare user’s Home Directory:

**cd /home/playsms**

**mkdir -p public_html log**

**sudo chmod 775 /home/playsms public_html log**

**sudo chown playsms.playsms -R /home/playsms**

**sudo chown www-data.playsms -R /home/playsms/log**

**ls -l /home/playsms**

## Create VirtualHost configuration file for domain playsms:

**sudo nano /etc/apache2/sites-enabled/000-default.conf**

```
<VirtualHost *:80>
    ServerName playsmm
    DocumentRoot /home/playsms/public_html
    ErrorLog /home/playsms/log/httpd-error.log
    CustomLog /home/playsms/log/httpd-accesss.log combined
    <Directory /home/playsms/public_html>
        AllowOverride FileInfo AuthConfig Limit Indexes
        Options MultiViews Indexes SymLinksIfOwnerMatch IncludesNoExec
        Require method GET POST OPTIONS
        php_admin_value engine On
    </Directory>
</VirtualHost>
```

Enable it:

**sudo systemctl reload apache2.service**

# Switch user as user playsms

## and test VirtualHost by create a PHP file in

**/home/playsms/public_html.**

**nano /home/playsms/public_html/test.php**

```
<?php
echo "<b>Welcome !!</b>";
```

Save the file and browse this file at your domain, in this example

**Browse to: http://ip_of_your_playsms_machine/test.php**

You know your VirtualHost is working when you see Welcome !! on your browser.
Remove test.php after testing:

**rm -f /home/playsms/public_html/test.php**

## ********** OPTIONAL IF WANT HPPTS **********

## 4.2. Install certbot (not needed…)

We will get the SSL certificate from Let's Encrypt and use certbot to install it on the server.
Install certbot:

**sudo apt install python3-certbot-apache**

## 4.3. Setup SSL Certificate

**Run certbot for Apache:**

**sudo certbot --apache**

Answer questions correctly. You will need to input your email address, choose A to Agree with the ToS and last choose Redirect (selection no. 2) to completely remove HTTP and just serve HTTPS by redirecting all HTTP requests to HTTPS.

Example of successful SSL certificate request and installation:

Visit ssllabs.com/ssltest and submit your domain to test your HTTPS configuration.

**********

# 5. Install playSMS (needed!!!)

Now that we have a working web server with PHP and HTTPS supports, and MySQL server, we can then install playSMS 1.4.6.

From now on you must execute commands as normal Linux user. In this article playSMS will be installed under user playsms as mentioned before.

## 5.1. Prepare Directories

Here are some important directories that need to be ready before playSMS installation:

public_html and log is already exists and prepared, they are created previously on section 4.1 as part as VirtualHost configuration. So now we need to create the rest and set proper permission.

## Then create directories:

**cd /home/playsms**

**mkdir -p bin etc lib src**

**sudo chmod 775 bin etc lib src**

Prepare log files too, this need to be done so that both web server Apache2 and playSMS daemon have write access to playSMS log files:

**cd /home/playsms**

**sudo touch log/audit.log log/playsms.log**

**sudo chmod 664 log/audit.log log/playsms.log**

**sudo chown www-data.playsms -R log**

**ls -l log**

# 5.2. Check PHP Modules

Required PHP modules should already be installed if you follow this article from the start, it is on section 3. But before proceeding with playSMS installation you need to make sure that required PHP modules are installed:

**php -m**

Make sure you see at least curl, gd, mbstring, mysqli and xml on the list. If they are not on the list then please install them, see section 3.

# 5.3. Prepare Database

Create MySQL database that will be used by playSMS:

sudo mysqladmin create playsms

Login as MySQL user root and create a new MySQL user for above database:

**sudo mysql**

**CREATE USER 'playsms'@'localhost' IDENTIFIED BY 'YourOwnPassword';**

**GRANT ALL PRIVILEGES ON playsms.* TO 'playsms'@'localhost';**

**FLUSH PRIVILEGES;**

**exit**

Do not copy-paste above SQL commands directly to MySQL console, you must use your own strong password, change the strongpasswordhere with your own strong password.

Execute (by root user)

**mysql_secure_installation**

As of this section you will have a MySQL database named playsms and MySQL normal user playsms with your own strong password which only have access to database playsms.

# 5.4. Get playSMS Source Code

playSMS source code available on Github, you will need git to get them.

**cd /home/playsms/src**

**Get playSMS version 1.4.6:**

## git clone -b 1.4.6 --depth=1 https://github.com/pappicio/playsms

## 5.5. Prepare install.conf

Go to playSMS source code directory, copy install.conf.dist to install.conf and then edit it.
Go to playSMS source code directory:

**cd /home/playsms/src/playsms**

copy and edit install.conf:

**cp install.conf.dist install.conf**

**nano install.conf**

These are values I set on install.conf:

remember…(CREATE USER 'playsms'@'localhost' IDENTIFIED BY 'YourOwnPassword';)

```
# INSTALL DATA
# ============
DBUSER="playsms"
DBPASS="YourOwnPassword"
DBNAME="playsms"
DBHOST="localhost"
DBPORT="3306"
WEBSERVERUSER="www-data"
WEBSERVERGROUP="www-data"
PATHSRC="/home/playsms/src/playSMS"
PATHWEB="/home/playsms/public_html"
PATHLIB="/home/playsms/lib"
PATHBIN="/home/playsms/bin"
PATHLOG="/home/playsms/log"
PATHCONF="/home/playsms/etc"
# END OF INSTALL DATA
# ===================
```

Values need to reflect your server configuration. If you follow this article from the start then above values should be correct, with exception your true database password (DBPASS) of course.

Save install.conf and ready to run install script.

# 5.6. Run playSMS Install Script

playSMS install script will download composer and download packages from repo.packagist.org. After that the script will copy necessary files from playSMS source code to public_html and bin.

Since theres requirement to be able to download from external site (repo.packagist.org), you have to make sure that external site is working and reachable.

But you can just start the install script, because you'll know if something not right, for example the script fail to download packages. When that happens you can fix the problem first, like fix your networking setup and perhaps firewall, or simply wait (theres a chance the external site down too), and then go back to re-run the install script.

Just to make sure that networking stuff is right, please see section 1.6.

OK, let's start the installation:

**cd /home/playsms/src/playsms**

**./install-playsms.sh**

Verify installation:

Press Y (you will be asked twice, answer Y both) and proceed the installation.

Successful installation will show that all playSMS daemon is running:

## give some permissions on playsms folders:

**cd /home/playsms**

**mkdir -p public_html log**

**sudo chmod 775 /home/playsms public_html log**

**sudo chown playsms:playsms -R /home/playsms**

**sudo chown www-data:www-data -R /home/playsms/public_html**

**sudo chown www-data:playsms -R /home/playsms/log**

Browse your playSMS, don’t worry if the login page looks broken, it’s because we haven’t configure playSMS to enable HTTPS, we will do that after this. For now, check if you can see playSMS login page.

# 5.7. Adjust config.php

Edit playSMS config.php and adjust some value, or just one part, the HTTPS support.

**nano /home/playsms/public_html/config.php**

Inside config.php:

Search for **logstate** and set it to 3

Search for **ishttps** and set it to true. (if prefer https and not only http)

# Optional but good to set!

**Optimize PHP-FPM**

**sed -i "s/;request_terminate_timeout = 0/request_terminate_timeout = 300/" /etc/php/7.2/fpm/pool.d/www.conf**

**sed -i "s/max_execution_time = 30/max_execution_time = 60/" /etc/php/7.2/fpm/php.ini**

**sed -i "s/upload_max_filesize = 2M/upload_max_filesize = 20M/" /etc/php/7.2/fpm/php.ini**

**sed -i "s/post_max_size = 8M/post_max_size = 20M/" /etc/php/7.2/fpm/php.ini**

**sed -i "s/memory_limit = 128M/memory_limit = 512M/" /etc/php/7.2/fpm/php.ini**

**systemctl enable php7.2-fpm**

**systemctl restart php7.2-fpm**

**systemctl reload apache2**

# Daemon result red color on web page: go in this file and modify:

**/home/playsms/public_html/plugin/feature/playsmslog/config.php**

From:

**$plugin_config['playsmslog']['playsmsd']['bin'] = '/home/playsms/bin/playsmsd';**

**$plugin_config['playsmslog']['playsmsd']['conf'] = '/home/playsms/etc/playsmsd.conf';**

to:

**$plugin_config['playsmslog']['playsmsd']['bin'] = '/home/playsms/bin/playsmsd';**

**$plugin_config['playsmslog']['playsmsd']['conf'] = '/home/playsms/etc/playsmsd.conf';**

# save, all green now!!!

## Edit also this file to remove (for me is +393xxx)

**/home/playsms/public_html/plugin/core/sendsms/fn.php**

After this line:

```
if (is_array($user)) {
		$prefix = ($user['replace_zero'] ? $user['replace_zero'] : $core_config['main']['default_replace_zero']);
		$local_length = (int) $user['local_length'];
		_log('before prefix manipulation:[' . $number . ']', 3, 'sendsms_manipulate_prefix');
```

## //add here//

```
	$number = str_replace('+', '', $number);
		if (substr($number, 0, 3) == '393') {
			_log('my own prefix manipulation not need:[' . $number . ']', 3, 'sendsms_manipulate_prefix');
		} else {
			_log('my own prefix manipulation needed changing it:[' . $number . ']', 3, 'sendsms_manipulate_prefix');
			$number = '39' . $number;
			_log('my own prefix manipulation needed:[' . $number . ']', 3, 'sendsms_manipulate_prefix');
		}
```

# Install Playams service:

on console write:

# su root

(enter root password)

**cat >> /etc/systemd/system/playsms.service << EOF**
**[Unit]**
**Description=playsms**
**After=mariadb.service**
**[Service]**
**Type=oneshot**
**RemainAfterExit=yes**
**ExecStart=/home/playsms/bin/playsmsd /home/playsms/etc/playsmsd.conf start**
**ExecStop=/home/playsms/bin/playsmsd /home/playsms/etc/playsmsd.conf stop**
**User=www-data**
**Group=www-data**
**[Install]**
**WantedBy=multi-user.target**
**EOF**

# Enable and execute playsms service:

**chmod 755 /home/playsms/bin/playsmsd**
**systemctl daemon-reload**
**systemctl enable playsms**
**systemctl restart playsms**
**systemctl status playsms**

# 5.8. Change Default Password

Go to your browser, browse the server and login as playSMS administrator, and change the default admin password immediately.

# Install smstools by compiled:

**sudo apt install smstools**

and then

**sudo update-rc.d -f smstools disable**

**sudo update-rc.d -f smstools  remove**

**go on my repository and download as zip the fix:**

**provide to set for all folders/subfolder/files, 0777 permission and as user/group: www-data, and at end execute:**

**sudo systemctl daemon-reload**

**sudo update-rc.d -f sms3 defaults**

# OR better:  Install smstools and comipile for debian 10/11/12 – Ubuntu 20/22/24

**apt-get install build-essential libusb-1.0 libusb-1.0-0-dev build-essential manpages-dev**

**sudo apt-get update & sudo apt-get install usb-modeswitch**

**cd /tmp**
**git clone --depth=1 https://github.com/pappicio/smstools3**
**cd smstools3**
**make**
**make install**
**systemctl restars sms3**

# 5.8. compiled smstools, now configure:

**mkdir -p /var/log/sms/stats**

**mkdir -p /var/spool/sms/ {checked,failed,incoming,outgoing,sent}**

**mkdir /var/spool/sms/modem1**

**chown www-data:www-data -R /var/spool/sms**

**chmod 777 -R /var/spool/sms**

**mv /etc/smsd.conf /etc/smsd.conf.dist**

## configure your own, my is:

```
devices = modem1
loglevel = 7
# logfiles
stats = /var/log/sms/stats
logfile = /var/log/sms/smsd.log
# Default queue directory = /var/spool/sms
outgoing = /var/spool/sms/outgoing
checked = /var/spool/sms/checked
failed = /var/spool/sms/failed
incoming = /var/spool/sms/incoming
sent = /var/spool/sms/sent
delaytime = 2
errorsleeptime = 10
blocktime = 180
autosplit = 3
# Queue configurations

[queues]
modem1 = /var/spool/sms/modem1

[modem1]
device = /dev/ttyUSB1
init = AT^CURC=0
###init2 = AT+CPMS="ME","ME","ME"
#pin  = 1234
report = yes
incoming = yes
queues = modem1
# mode = new
smsc = 393770001016
baudrate = 115200 ###19200
memory_start = 0
decode_unicode_text = yes
#cs_convert = yes
report_device_details = no
```

## ...and at end...

**sudo update-rc.d sms3 defaults**

**sudo reboot**

and all works!!!

## Block usb modem on same ttyUSBX for ever:

Get deviceid for the dongle

**sudo lsusb**

Get to know properties of the device while it is switched in:

**udevadm info -q all -p $(udevadm info -q path -n /dev/ttyUSB1)**

( if your system is old, try instead with this command: )
**( udevinfo -a -p $(udevinfo -q path -n /dev/ttyUSB0) )**
Find some property that can identify the device (uniquely), for instance “serial”
Create a file called

**/etc/udev/rules.d/10-usb-serial**

which contains the line:

**BUS=="usb", ATTR{serial}=="xxxx", NAME="ttyUSB1"**
Note the two equal signs for properties that are tested, and one for that which is assigned to.

## Or better:

### Persistent paths for dynamic device file

Intro.

When USB GSM modem plugged to a server Linux kernel assigned dynamic device file /dev/ttyUSB*, such as /dev/ttyUSB0 or /dev/ttyUSB1. For example, USB GSM modem with 2 ports will then be assigned to /dev/ttyUSB0 for port 1 and /dev/ttyUSB1 for port 2.

Problem starts when we unplug the GSM modem and re-plug back afterwards. Linux kernel will then assign different device file to it, was /dev/ttyUSB0 now /dev/ttyUSB2 and was /dev/ttyUSB1 now /dev/ttyUSB3.

# Let’s talk about the problem.

Put your attention to this SMSC configuration part of our Kannel:

**## SMSC gsm1**
**group = smsc**
**smsc = at**
**smsc-id = gsm1**
**modemtype = wavecom**
**device = /dev/ttyUSB0**
**log-file = /var/log/kannel/smsc-gsm1.log**
**log-level = 0**

**## SMSC gsm2**
**group = smsc**
**smsc = at**
**smsc-id = gsm2**
**modemtype = wavecom**
**device = /dev/ttyUSB1**
**log-file = /var/log/kannel/smsc-gsm2.log**
**log-level = 0**

Note that SMSC ID gsm1 is mapped to /dev/ttyUSB0, and SMSC ID gsm2 is mapped to /dev/ttyUSB1.

# Here’s how to get what you want.

With the help of udev configuration and a script we can dynamically map device file to a specific, and persistent, path, upon plugging the physical device.

**Create /etc/udev/rules.d/80-ttyusb-map.rules:**

**nano /etc/udev/rules.d/80-ttyusb-map.rules**

And fill it with this:

**ACTION=="add", KERNEL=="ttyUSB[0-9]*", PROGRAM="/etc/udev/rules.d/ttyusb-map.sh %p", SYMLINK+="gsm%c"**

Then create **/etc/udev/rules.d/ttyusb-map.sh:**

**touch /etc/udev/rules.d/ttyusb-map.sh**

**chmod 755 /etc/udev/rules.d/ttyusb-map.sh**

**nano /etc/udev/rules.d/ttyusb-map.sh**

And fill it with this:

```
#!/usr/bin/perl -w
@items = split("/", $ARGV[0]);
for ($i = 0; $i < @items; $i++) {
    if ($items[$i] =~ m/^usb[0-9]+$/) {
        print $items[$i + 1] . "\n";
        last;
    }
}
```

That is all.

Now try to plug GSM modem, and then plug it back. We should see that /dev/gsm1-1 symlink to /dev/ttyUSB0 and /dev/gsm2-1 symlink to /dev/ttyUSB1.

See example below:

```
[anton@srv ~]$ ls -l /dev/gsm*
lrwxrwxrwx 1 root root 7 Mei  4 15:40 /dev/gsm1-1 -> ttyUSB0
lrwxrwxrwx 1 root root 7 Mei  4 15:40 /dev/gsm2-1 -> ttyUSB1
```

Those symlinks can be different each time you plug and re-plug the GSM modem, or restart the server, but the name of those device files are persistent.

We can then use /dev/gsm1-1 as our map to physical USB port 1 and /dev/gsm2-1 as our map to physical USB port 2.

Your Kannel configuration would then be like this:

**## SMSC gsm1**
**group = smsc**
**smsc = at**
**smsc-id = gsm1**
**modemtype = wavecom**
**device = /dev/gsm1-1**
**log-file = /var/log/kannel/smsc-gsm1.log**
**log-level = 0**

**## SMSC gsm2**
**group = smsc**
**smsc = at**
**smsc-id = gsm2**
**modemtype = wavecom**
**device = /dev/gsm2-1**
**log-file = /var/log/kannel/smsc-gsm2.log**
**log-level = 0**

**Restart your Kannel and tail SMSC log files, see if Kannel works properly.**

**tail -f /var/log/kannel/smsc-gsm1.log**
**tail -f /var/log/kannel/smsc-gsm2.log**

## MANAGE ACL per menu limitati per gli users/subusers:

## per gli USERS:

inc=feature_phonebook,
inc=core_user&route=subuser_mgmnt,
inc=core_user&route=user_pref&op=user_pref,
inc=feature_queuelog&op=queuelog_list

### o meglio ancora:

**inc=feature_phonebook,**
**inc=core_user&route=subuser_mgmnt,**
**inc=core_user&route=user_pref&op=user_pref,**
**inc=feature_report&route=user_inbox&op=user_inbox,**
**inc=feature_queuelog&op=queuelog_list**

## per i subuser:

inc=core_sendsms,
inc=feature_report&route=user,
inc=feature_schedule,
inc=feature_msgtemplate,
inc=core_user&route=user_pref&op=user_pref,
inc=feature_queuelog

### o meglio ancora:

**inc=core_sendsms,**
**inc=feature_report&route=user,**
**inc=feature_schedule,**
**inc=core_user&route=user_pref&op=user_pref,**
**inc=feature_report&route=user_inbox&op=user_inbox,**
**inc=feature_queuelog**

**SUBUSERS CON PHONEBOOK:**

inc=feature_phonebook,
inc=core_sendsms,
inc=feature_report&route=user,
inc=feature_schedule,
inc=feature_msgtemplate,
inc=core_user&route=user_pref&op=user_pref,
inc=feature_queuelog

## o meglio ancora:

**inc=feature_phonebook,**
**inc=core_sendsms,**
**inc=feature_report&route=user,**
**inc=feature_schedule,**
**inc=core_user&route=user_pref&op=user_pref,**
**inc=feature_report&route=user_inbox&op=user_inbox,**
**inc=feature_queuelog**

# Playsms trick on php-html files/functions

### /plugin/core/user/fn.php

// commentando questa funzione si possono aggiungere quanti subuser (addetti all'INVIO SMS) si vuole, con lo stesso numero telefonico (magari quello del centralino COC).

//che è il minimo comun denominatore per vedere la rubrica che crea l'utente del compartimento!!!

Quindi basta aggiungere un contatto nella rubrica tipo “f_a_k_e_n_a_p_o_l_i_u_s_e_r” con lo stesso numero telefonico (081000000) e tutti gli utenti addetti all’invio SMS che saranno creati dall’user di turno, avranno lo stesso numero telefonico personale (esempio: 081000000), cosi inserito “fakenapoliuser” in rubrica e nel gruppo NAPOLI, potranno vedere la rubruca!!!

E seguendo questo discorso logico (aggiiungendo un nuovo gruppo con nuovi contatti e nuovi subuser / utente con stesso numero, esempio f_a_k_e_p_g_n_a_p_o_l_i (081111111)) si puo creare quante ribriche si vuole e ognuno vedra solo il feuppo cui fa parte (con il fakenumebr, diciamo!)

```
                // check mobile, must check for duplication only when filled
//              if ($ret['status'] && $data['mobile']) {
//                      if (dba_isexists(_DB_PREF_ . '_tblUser', array(
//                              'flag_deleted' => 0,
//                              'mobile' => $data['mobile']
//                      ), 'AND')) {
//                              if ($data['mobile'] != $existing['mobile']) {
//                                      $ret['error_string'] = _('Account with this mobile already exists') . " (" . _('mobile') . ": " . $data['mobile'] . ")";
//                                      $ret['status'] = false;
//                              }
//                      }
//              }
        }
```

## Per dare tutti i gruppi imitati (ACL) anche ai subusers cambiare le seguenti righe di codice:

**in: /plugin/core/user/subuser_mgmnt.php**

//////$option_acl = _select('add_acl_id', array_flip(acl_getallbyuid($user_config['uid'])));
$option_acl = _select('add_acl_id', array_flip(acl_getall()));

e

in: /plugin/core/user/user_config.php

```
if ($user_edited['status'] == 4) {
                        $parent_id = user_getparentbyuid($user_edited['uid']);
                        if ($parent_id == $user_config['uid']) {
////////$c_option_acl = array_flip(acl_getallbyuid($user_config['uid']));
        $c_option_acl = array_flip(acl_getall());
```

## FACOLTATIVI (hidden by ACL)

per rendere readonly il telefono di un subuser (che ricordiamo deve essere fisso e uguale al fake-contatto in ribrica e nel gruppo, senno addio invio sms con ricerca….)

## qui: /plugin/core/user/user_prefs.php

**<tr>**
**<td>{{ Mobile }}</td>**
**<td><input type=text maxlength=20 name=up_mobile value="{{ mobile }}" readonly></td>**
**</tr>**

In pratica aggiungere “readonly”
per rendere readonly la “firma” del subuser qui:

# in: /plugin/core/user/template/user_config.html

**<tr>**

**<td>{{ Default message footer }}</td>**

**<td><input type=text maxlength=30 name=up_footer value="{{ footer }}" readonly> {{ HINT_MAX_ALPHANUMERIC }}**

**</td>**

**</tr>**

In pratica aggiungere “readonly”

Per rendere la “firma” readonly anche su invio sms qui:

**/plugin/core/semdsms/templates/sendsms.html**

**<label for="msg_footer">{{ Message footer }}</label>**

** **

**<input type="text" name="sms_footer" id="msg_footer"**

**style="width: 100%" value="{{ sms_footer }}" readonly>**

** **

**In pratica aggiungere “readonly”.**

# Install Adminer project homepage

Newer versions are offered here, such as 4.2.1.

**sudo mkdir /usr/share/adminer**

**sudo wget "http://www.adminer.org/latest.php" -O /usr/share/adminer/latest.php**

**sudo ln -s /usr/share/adminer/latest.php /usr/share/adminer/adminer.php**

**echo "Alias /adminer /usr/share/adminer/adminer.php" | sudo tee /etc/apache2/conf-available/adminer.conf**

**sudo a2enconf adminer.conf**

Once the installation completes, restart Apache.

**sudo service apache2 restart**

At this point, the setup is complete. You can access Adminer at either of the following addresses.

**http://[SERVER_IP]/adminer**

or:

**http://[SERVER_IP]/adminer.php**