Installation and Configuration of Pandora FMS and SMS Gateway

Installation and Configuration of Pandora FMS and SMS Gateway

About the GSM device

A special GSM device is used to send SMS via a serial port (USB). You can use either another similar GSM model, or a cell phone with USB or serial connection. The device used here is an MTX 65 v3. This device can be purchased commercially for approximately $100 USD from various web sites:

- http://matrix.es
- http://www.youtube.com/watch?v=OxcKAarS2M0

As you can see in YouTube, it is a very small and GNU/Linux compatible device, which has different optional components, such as a GSM antenna that is very useful, for example, if the data center is underground.

Installing the Device

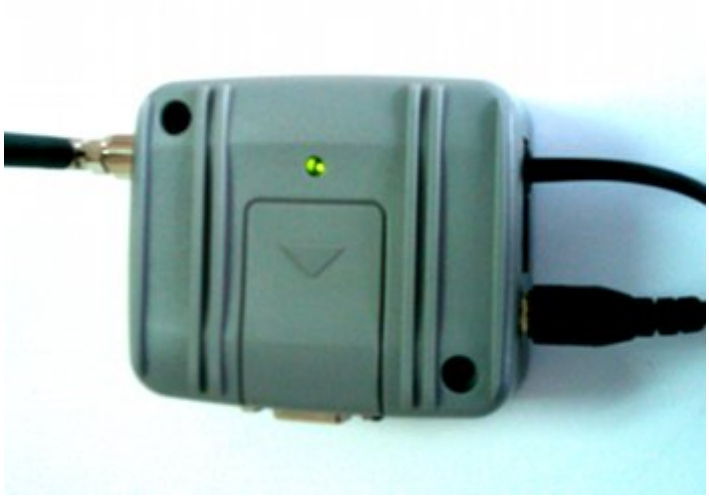
The first step is to install the hardware device. This device is composed of several parts:

- Standard USB cable, with small connector and an end.
- Power supply. In this sample is European 220 volts, if you live in America (except Chile) please be sure that power supply supports 110 or 120 volts.
- SIM card.
- Pandora FMS SMS gateway device.

Open the Pandora FMS SMS gateway device.

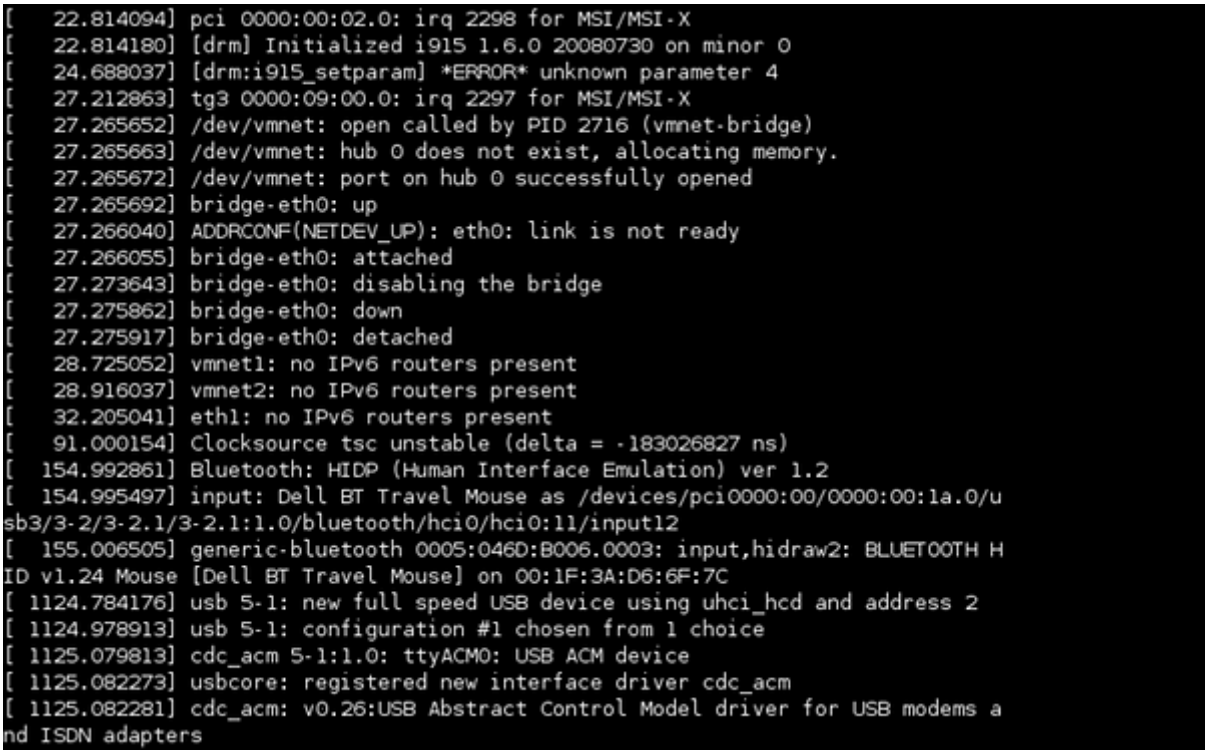
Put the SMS card inside.

Plug to network in the power input, plug the USB cable in the SMS Gateway device.



Connect the other end (USB cable) to the Pandora FMS server using a standard USB port.

When you connect the device to the server, wait a few seconds and run dmesg command from the command line, you should see something like this screenshot..



This means your device has been recognized by the kernel and it is ready to accept commands on a device, such as `/dev/ttyACM0`.

If you came up to here, the hardware setup is done. If not, please review all steps and make sure that:

- The device is connected and the wire is blinking in a green color.
- The device is connected to the USB port, by both sides of the wire, one side to the SMS device, and other side to Pandora FMS server host.
- The device has a SIM card inside, and it is placed properly.

Configure SMSTools to Use the New Device

This device is managed by a software package called SMSTools. You can install smstools using the package provided by your GNU/Linux distribution selected or use RPM package (Red Hat Package Manager o RPM Package Manager) provided by Artica PFMS (only for RPM distributions: Fedora, Mandriva, Mageia, PCLinuxOS, among others).

RPM based system

Using the RPM provided by Artica PFMS on Red Hat is very simple. Just install it with the following command:

```
# rpm -i smstools*.rpm
```

Configure SMStools

Edit the base configuration file:

```
# vi /etc/smsd.conf
```

Enter this contents. If your dmesg output is not `ttyACM0`, *use the* `tty` *device detected by your system*.

```
# Example smsd.conf. Read the manual for a description

devices = GSM1
logfile = /var/log/smsd.log
loglevel = 10

[GSM1]
device = /dev/ttyACM0
incoming = no
pin = 2920
```

Use the PIN assigned to your SIM, in this example, PIN is `2920`.

Then, start manually smstools:

```
# /usr/bin/smstools start
```

Send a test SMS. Beware: Phone numbers must have full (int.) preffix. In this sample, +34 is Spanish preffix, and my phone number is 627934648:

```
$ sendsms 34627934648 "Pandora FMS rocks"
```

Wait a minute and watch your logs to check that everything is correct. You should receive the SMS in a few seconds. Depending on the network, the first SMS can timeout every 10-20 seconds, after that, wait. The next SMS should be almost immediate. SMSTools uses a queue to send messages, so you can send as many as you want, *and they will be out as soon as your mobile network could manage*.

To see the logs:

```
# cat /var/log/smsd.log
2009-11-12 11:30:12,2, smsd: Smsd v2.2.20 started.
2009-11-12 11:30:12,6, smsd: outgoing file checker has started.
2009-11-12 11:30:12,6, GSM1: Modem handler 0 has started.
2009-11-12 11:30:13,6, smsd: Moved file /var/spool/sms/outgoing/send_mNZxHa to
/var/spool/sms/checked
2009-11-12 11:30:13,6, smsd: I have to send 1 short message for
/var/spool/sms/checked/send_iUegPD
```

```
2009-11-12 11:30:13,6, GSM1: Sending SMS from  to 627934648
2009-11-12 11:30:13,6, GSM1: Checking if modem is ready
2009-11-12 11:30:13,7, GSM1: -> AT
2009-11-12 11:30:13,7, GSM1: Command is sent, waiting for the answer
2009-11-12 11:30:14,7, GSM1: <- AT
OK
2009-11-12 11:30:14,6, GSM1: Checking if modem needs PIN
2009-11-12 11:30:14,7, GSM1: -> AT+CPIN?
2009-11-12 11:30:14,7, GSM1: Command is sent, waiting for the answer
2009-11-12 11:30:14,7, GSM1: <- AT+CPIN?
+CPIN: SIM PIN
OK
2009-11-12 11:30:14,5, GSM1: Modem needs PIN, entering PIN...
2009-11-12 11:30:14,7, GSM1: -> AT+CPIN="2920"
2009-11-12 11:30:14,7, GSM1: Command is sent, waiting for the answer
2009-11-12 11:30:15,7, GSM1: <- AT+CPIN="2920"
OK
2009-11-12 11:30:15,7, GSM1: -> AT+CPIN?
2009-11-12 11:30:15,7, GSM1: Command is sent, waiting for the answer
2009-11-12 11:30:15,7, GSM1: <- AT+CPIN?
+CPIN: READY
OK
2009-11-12 11:30:15,6, GSM1: PIN Ready
2009-11-12 11:30:15,6, GSM1: Checking if Modem is registered to the network
2009-11-12 11:30:15,7, GSM1: -> AT+CREG?
2009-11-12 11:30:15,7, GSM1: Command is sent, waiting for the answer
2009-11-12 11:30:16,7, GSM1: <- AT+CREG?
+CREG: 0,2
OK
2009-11-12 11:30:16,5, GSM1: Modem is not registered, waiting 10 sec. before
retrying

2009-11-12 11:30:26,7, GSM1: -> AT+CREG?
2009-11-12 11:30:26,7, GSM1: Command is sent, waiting for the answer
2009-11-12 11:30:26,7, GSM1: <- AT+CREG?
+CREG: 0,5
OK
2009-11-12 11:30:26,6, GSM1: Modem is registered to a roaming partner network
2009-11-12 11:30:26,6, GSM1: Selecting PDU mode
2009-11-12 11:30:26,7, GSM1: -> AT+CMGF=0
2009-11-12 11:30:26,7, GSM1: Command is sent, waiting for the answer
2009-11-12 11:30:26,7, GSM1: <- AT+CMGF=0
OK
2009-11-12 11:30:26,7, GSM1: -> AT+CMGS=94
2009-11-12 11:30:26,7, GSM1: Command is sent, waiting for the answer
2009-11-12 11:30:27,7, GSM1: <- AT+CMGS=94>

2009-11-12 11:30:27,7, GSM1: ->
0011000991269743 46F900F1FF5CC8373BCC0295E7F437A83C07D5DDA076D93D0FABCBA069730A22
97417079BD2C0EBB406779789C0ECF41F0B71C44AF83C66FB7391D76EBC32C503B3C46BFE9651608
1E7693DFF230C8D89C82E4EFF17A0E
2009-11-12 11:30:27,7, GSM1: Command is sent, waiting for the answer
```

```
2009-11-12 11:30:31,7, GSM1: <-
0011000991269743 46F900F1FF5CC8373BCC0295E7F437A83C07D5DDA076D93D0FABCBA069730A22
97417079BD2C0EBB406779789C0ECF41F0B71C44AF83C66FB7391D76EBC32C503B3C46BFE9651608
1E7693DFF230C8D89C82E4EFF17A0E
+CMGS: 0
OK
2009-11-12 11:30:31,5, GSM1: SMS sent, To: 627934648
2009-11-12 11:30:31,6, smsd: Deleted file /var/spool/sms/checked/send_iUegPD
2009-11-12 11:30:32,6, smsd: I have to send 1 short message for
/var/spool/sms/checked/send_mNZxHa
2009-11-12 11:30:32,6, GSM1: Sending SMS from  to 34627934648
2009-11-12 11:30:32,7, GSM1: -> AT+CMGS=29
2009-11-12 11:30:32,7, GSM1: Command is sent, waiting for the answer
2009-11-12 11:30:33,7, GSM1: <- AT+CMGS=29>

2009-11-12 11:30:33,7, GSM1: ->
0011000B914326974346F900F1FF11D0B09BFC968741C6E614247F8FD773
2009-11-12 11:30:33,7, GSM1: Command is sent, waiting for the answer
2009-11-12 11:30:36,7, GSM1: <-
0011000B914326974346F900F1FF11D0B09BFC968741C6E614247F8FD773
+CMGS: 1
OK
2009-11-12 11:30:36,5, GSM1: SMS sent, To: 34627934648
2009-11-12 11:30:36,6, smsd: Deleted file /var/spool/sms/checked/send_mNZxHa
```

Finally, some tasks to ensure the operation for the future:

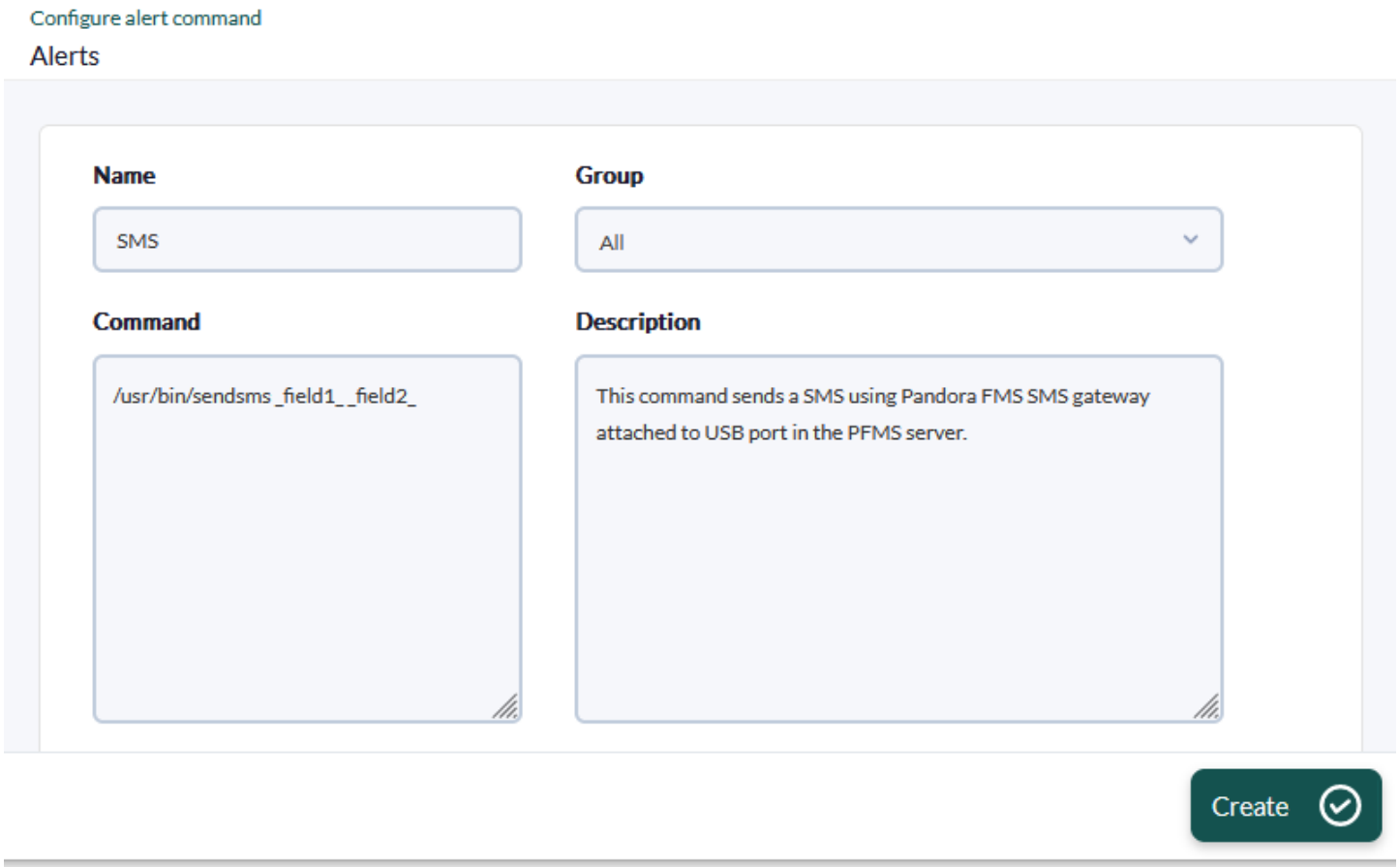
1. Set 1 to loglevel in `/etc/smsd.conf` to avoid a very big, non-necessary log file.

2. Make sure that smsd is set to start automatically when the system restarts (this means a link to `/etc/init.d/sms` to `/etc/rc2.d/S90sms` or `/etc/rc.d/rc2.d/S90sms`). If you installed it from a package, it may already exist in your system, just check it.

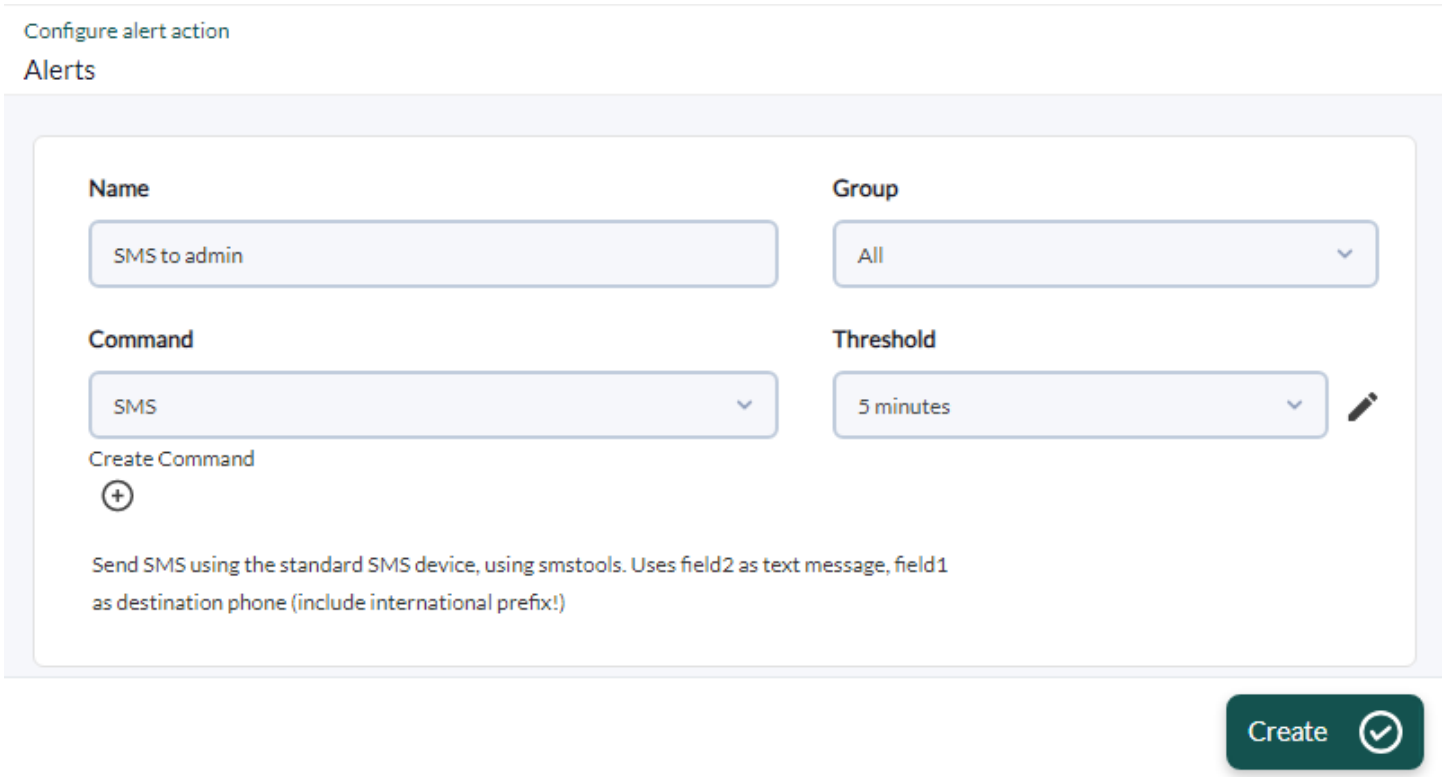
Configure Pandora FMS Alert

This steps reproduce the basic steps to create SMS alerts in Pandora FMS 3.x. For more information see “Pandora FMS Alert System”.

Create the command:



Create the action:



Associate the action to a module using a previous alert template. In this case, alert template will be fired when the module status would be CRITICAL.

Gateway to Send SMS using a generic hardware and Gnokii

It is recommended to keep your systems updated with the latest version of Pandora FMS. This information is kept for historical purposes.

This method for sending SMS was the one proposed in Pandora FMS versions 1.x and 2.x. in version 3.x smstools is used. The method based on smstools is better and it is not recommended to use Gnokii due to ist complexity and possibilities of failure. However it is referenced here to provide alternative methods to smtools.

This section describes how to build a SMS sending gateway based in a sending queue. This way, it is possible to implement a SMS sending server, connected with a mobile and sending the SMS through the software of the Gnokii project, and different remote servers can send its messages in order the SMS sending server could process them. This allow that different Pandora FMS servers (or another machines that want to use the *gateway*) could send messages in a centralized way, *without having to have a mobile for each server*.

In the first place, you should create an `sms` user in the machine where you want to install it in the SMS sending gateway. After this, create the directories `home/sms` y `/home/sms/incoming`. If you want to use the SMS sending *gateway* from another machines, you will need to make accessible the directory */home/sms/incoming* for other servers through any file sending system or file systems partition: NFS, SMB, SSH (scp), TCP or Tentacle.

The SMS sending gateway mechanism is very easy: for each file that is at the directory `/home/sms/incoming`, an SMS will be processed, deleted and sent, with the file content. This file should have an specific format, which is detailed here:

```
Phonenumber|SMSText
```

Gateway Implementation with Gnokii

It is recommended to keep your systems updated with the latest version of Pandora FMS. This information is kept for historical purposes.

You must create four *scripts*:

SMS: Script that sends the SMS using Gnokii through an USB data cable.This script is only in the system where the sending gateway is (the system that has the data cable connected to a GSM

mobile).

SMS_GATEWAY: Script that process in a periodical way the entry directory (`/home/sms/incoming`), processing files that are waiting to be send. This script is only in the system that is used as sending gateway.

SMS_GATEWAY_LAUNCHER: launcher *script* for the SMS_GATEWAY script (start and stop daemon). This script is only in the system that does the sending gateway.

COPY_SMS: copies an SMS using the scp command from a client system to a gateway system. Uses the TELEPHONE as first parameter, and the second as text to send (using "" for specifying each parameter).The script trust in the SSH automatic autentication and in the `sms` user for the transfer. In the local system you can remplace the scp for the cp command or use a system like Tentacle to transfer the file.

sms

It is recommended to keep your systems updated with the latest version of Pandora FMS. This information is kept for historical purposes.

This is the *script* that sends SMS using Gnokii. You should have Gnokii well configured (using the file `/etc/gnokii.conf` or similar). Probably should be the user root to could launch the script, or establish the `SETUIDO` in the Gnokii binary.

```
#!/bin/bash
texto=$1
number=$2
if [ $# != 2 ]; then
echo "I need more parameters"
exit 1;
fi
/bin/echo $1 | /usr/local/bin/gnokii --sendsms $2
```

sms_gateway

It is recommended to keep your systems updated with the latest version of Pandora FMS. This information is kept for historical purposes.

This is the gateway daemon script:

```
#!/bin/bash

INCOMING_DIR=/home/sms/incoming
HOME_DIR=/home/sms

while [ 1 ]
do

        for a in `ls $INCOMING_DIR`
        do
                if [ ! -z "$a" ]
                then
                        NUMBER=`cat $INCOMING_DIR/$a | cut -d "|" -f 1`
                        MESSAGE=`cat $INCOMING_DIR/$a | cut -d "|" -f 2`
                        TIMESTAMP=`date +"%Y/%m/%d %H:%M:%S"`
                        echo "$TIMESTAMP Sending to $NUMBER the message
$MESSAGE">> $HOME_DIR/sms_gateway.log
                        $HOME_DIR/sms "$MESSAGE" "$NUMBER"
                        echo "$TIMESTAMP Deleting $a">>
$HOME_DIR/sms_gateway.log
                        rm -Rf $INCOMING_DIR/$a
                        sleep 1
                fi
        done
        sleep 5
done
```

sms_gateway_launcher

It is recommended to keep your systems updated with the latest version of Pandora FMS. This information is kept for historical purposes.

This is the launching script form the `sms_gateway`:

```
#!/bin/bash

# SMS Gateway, startup script
# Sancho Lerena, <slerena@gmail.com>
# Linux Version (generic)

# Configurable path and filenames
SMS_GATEWAY_HOME=/home/sms
SMS_PID_DIR=/var/run
SMS_PID=/var/run/sms.pid

# Main script
```

```
if [ ! -d "$SMS_PID_DIR" ]
then
        echo "SMS Gateway cannot write it's PID file in $SMS_PID_DIR. Please
create directory or assign appropiate perms"
        exit
fi

if [ ! -f $SMS_GATEWAY_HOME/sms_gateway ]
then
        echo "SMS Gateway not found, please check setup and read manual"
        exit
fi

case "$1" in
  start)
        OLD_PATH="`pwd`"
        if [ -f $SMS_PID ]
        then
                CHECK_PID=`cat $SMS_PID`
                CHECK_PID_RESULT=`ps aux | grep -v grep | grep "$CHECK_PID" |
grep "sms_gateway" | wc -l`
                if [ $CHECK_PID_RESULT == 1 ]
                then
                        echo "SMS Gateway is currently running on this machine
with PID ($CHECK_PID). Aborting now..."
                        exit
                fi
        fi

        nohup $SMS_GATEWAY_HOME/sms_gateway> /dev/null 2> /dev/null & 2>
/dev/null> /dev/null
        sleep 1

        MYPID=`ps aux | grep "$SMS_GATEWAY_HOME/sms_gateway" | grep -v grep |
tail -1 | awk '{ print $2 }'`
        if [ ! -z "$MYPID" ]
        then
                echo $MYPID> $SMS_PID
                echo "SMS Gateway is now running with PID $MYPID"
        else
                echo "Cannot start SMS Gateway. Aborted."
        fi
        cd "$OLD_PATH"
        ;;
  stop)
        if [ -f $SMS_PID ]
        then
           echo "Stopping SMS Gateway"
           PID_2=`cat $SMS_PID`
           if [ ! -z "`ps -F -p $PID_2 | grep -v grep | grep 'sms_gateway' `" ]
           then
               kill `cat $SMS_PID` 2> /dev/null> /dev/null
```

```
            else
                echo "SMS Gateway is not executing with PID $PID_2, skip Killing
step"
            fi
            rm -f $SMS_PID
        else
          echo "SMS Gateway is not running, cannot stop it."
        fi
        ;;
  force-reload|restart)
        $0 stop
        $0 start
        ;;
  *)
        echo "Usage: sms_gateway {start|stop|restart}"
        exit 1
esac
```

copy_sms

It is recommended to keep your systems updated with the latest version of Pandora FMS. This information is kept for historical purposes.

This small script creates a SMS sending file in a client machine and copies it to the SMS gateway using scp:

```
#!/bin/bash

SERIAL=`date +"%j%M%s"`
SERIAL=`hostname`_$SERIAL

TEL=$1
TEXT=$2

echo $TEL\|$TEXT>> /tmp/$SERIAL
scp /tmp/$SERIAL sms@192.168.1.1:/home/sms/incoming
rm -Rf /tmp/$SERIAL1
```

Go back to Pandora FMS documentation index