Freedom Fone File Permissions

Alberto Escudero-Pascual <aep@it46.se> April 2013

Table of Contents

Introduction....1
Software Components....2
  Freeswitch....2
  iWatch....2
  Dispatcher....2
  Database backend....3
  Cron deamon....3
  GUI....4
Diagram and summary of interactions....4
Editing files via the web interface and reloading services....5

Introduction

Freedom Fone uses several software components running under different permissions. In order to integrate new sofware with Freedom Fone is important to understand the permission structure. This document summarizes the overall architecture for branch 3.0 and its internal permissions.

Development Branch 3.0 corresponds to 2.S software releases also known as “Manguensis - Security Editions”.

Software Components

Freeswitch

Freeswitch runs the voice telephony services backend. Freeswitch is responsible of attending voice calls coming via SIP or gsmopen channel, executing Javascript applications as the leave-a-message service, executing voice menus using native freeswitch XML and processing SMS arriving to the system via gsmopen.

Freeswitch also serves and receives events from/to external applications via the Event Socket Layer (ESL)

Freeswitch runs as user *freeswitch*

iWatch

Iwatch monitor the filesystem for incoming audio files uploaded via the web interface or files that are deliver by Freeswith leave a message application.
Files are convereted from WAV to MP3 and viceversa. Iwatch is reponsible of changing file permissions to files are available to the web interface with user *www-data*

iWatch runs as user *root* and changes permissions of files to be *www-data readable*

Dispatcher

The dispatcher is a service/daemon that is listening to Freeswitch events (ESL) and pushing them into the spooler_in database. The Dispatcher is a PHP5 client (CLI) script that runs under user *www-data* so its logs are available to the web. It is possible to run the dispatcher as different user.

Dispatcher runs as user *www-data* but run as different user if there is no need to make its logs available to the web.

The dispatcher needs credentials to access the spooler_in database and push/pull events from Freeswitch event socket layer. Credentials are stored at dispatcher_in/config.php

Database backend

A MySQL database that stores that stores all freedomfone databases

Freedom Fone runs three major databases

- **spooler_in**: this database stores the events delivered by Freedom Fone dispatcher that is listening to Freeswitch event socket layer. The Spooler is separated from the main freedomfone database to allow different applications to access the same event or subsets of it.
  Freedom fone applications poll data from the spooler allowing concurrency over the same data and permission separation between applications. This
- **freedomfone** this database stores the data uses by the applications and the graphical interface
- **smsd (optional):** this database is used by Gammu-smsd to store SMS arriving to Huawei and Mobigater operating as SMS senders and receivers.
  Gammu-Smsd and Freedom Fone interoperate by means of a cronjob that pools incoming SMS from the smsd database and create events into Freeswitch that are attended by the dispatcher. (See gammu-smsd/daemon/gammu_daemon.php)
  When an incoming SMS arrives to a device attended by Gammu-smsd, the SMS is stored in the smsd table. The cronjob pools the SMS and creates a external event into Freeswitch (jsrun /opt/freeswitch/scripts/freedomfone/sms/main.js) that is attended by the dispatcher in a similar way that SMS are attended internally when arriving by a Freeswitch gsmopen device.

Mysql runs as *mysqld,* each database has its own access grants created at:

- /gui/app/sql/freedomfone_db_createdb_user.sql
- /dispatcher_in/sql/spooler_db_createdb_user.sql
- (smsd) gammu-smsd/doc/freedomfone_gammu_smsd_v1.odt

All credentials for application access to the databases are available at:

- gui/app/config/config.php

Cron deamon

The cron service is reponsible of refreshing the GUI with new incoming data, processing incoming SMSs that are stored as e-mails in the Office Route (pop3 polling) and processing incoming SMSs that are stored in the smsd table of Gammu-smsd
The cron daemon can also perform sweeping tasks (deleting files and records).

The cron daemon runs as *root* and requires access to the credentials to Office Route POP3,

smsd database. For sweeping the cron daemon requires access to all databases.

GUI

The graphical interface runs as the apache web server user and needs to be able to read and write audio files, access logs, poll data from the spooler_in database and push data into the different applications databases.

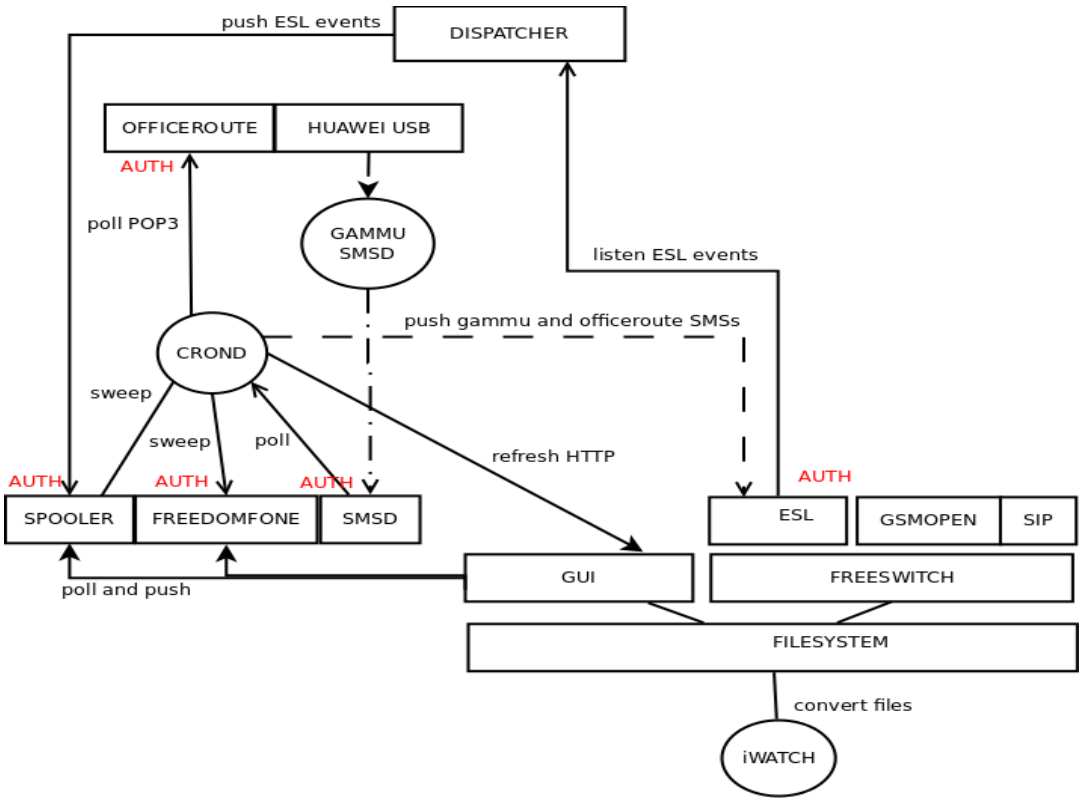
The GUI is also responsible of polling status information of OpenOffice by means of SNMP.

As in version 3.0/1966, the GUI talks directly to Freeswitch to obtain information as the status of Freeswitch and the gsmopen devices available. This will change in the future and GUI will have no access to Freeswitch ESL layer.

The GUI runs as www-data and requires access to Office Route SNMP, spooler_in and freedomfone databases.

Diagram and summary of interactions

The diagram shows the interactions between the network components and where authentication between components is needed.



Service	Run as	Access to	Needs
Dispatcher	www-data	Spooler Database Event Socket Layer	Auth to spooler_in Auth to ESL
Freeswitch	freeswitch	Filesystem /dev/ttyACM	Device Access to gsmopen devices
GUI	www-data	Audio files Logs Freedom Fone Database SNMP Access to Office Route Freeswitch	Files as www-data Logs as www-data Auth to freedomfone Auth to SNMP
Iwatch	root	Audio Files	Convert files to www-data
Cron Daemon	root	Office Route POP3 smsd Database (gammu support) Event Socket Layer freedomfone, spooler Database (sweeper) Refresh HTTP	Auth to Office Route POP3 Auth to smsd Auth to freedomfone, spooler_in Granted Access to refresh method.
Gammu	root	Smsd database (gammu support) /dev/ttyUSB# /dev/ttyACM#	Auth to smsd Device Access to Huawei/Mobigater

Editing files via the web interface and reloading services

All files written via the GUI will have www-data as owner. In case that there is a need to edit configuration files from the major components via the web, a permission wrapper is needed to change the file owner of the configuration file and stop and/or start the services.

The overall logic should works as follows:

A cron process running as (root) is responsible of detecting new configuration files pushed via the GUI.

The cron process is responsible of checking the integrity of the configuration file. Once the configuration files is validated, a permission change might be needed. This permission change needs to be performed by the cron process.

Similarly, another cron process can take care of restarting the services invoking a init.d script (System V) or a upstart script (Ubuntu).

The cron process needs to be able to inform the GUI of the result of the changes, this can be done by exposing the logs to the www-data user.

The cron process also needs to be able to revert to the old configuration file in case of failure,

Integrating Gammu Smsd with Freedom Fone GUI

To allow the Freedom Fone GUI to edit gammu configuration files and restart the service, the following steps are needed.

1. Write a new Cake PHP vendor and a model that reads configuration files
2. Provide information of the current configuration file
3. Write a new Cake PHP view that allows the user to edit the configuration file
4. Write a cron job that compares the current configuration file with the one pushed by the GUI, makes a backup of the old configuration file, fix permissions
5. Write a new Cake PHP view that allows the user to stop/start the service. This can be done by written in a file what service needs to be restarted and when,.
6. Write a cron job that checks if a new service needs to be restarted. Validate that the request is valid. Validate the dependencies, for example if freeswitch is stopped, the dispatcher also needs to be reloaded etc.