

# Freedom Fone



FREEDOM ★ FONE  
IT'S FOR YOU

## Installation Guide

### Part 2: Middle ware and Application Layer

December 2009  
Author: IT46  
Version: 1.1

## Table of Contents

1. Data flow from FreeSWITCH to CakePHP .....	1
2. Assumptions .....	2
3. The dispatcher .....	2
3.1 Database setup .....	2
3.2 Configuration file .....	3
3.3 Libraries .....	4
3.4 Tests available .....	4
3.5 Run dispatcher .....	4
3.6 Test the dispatcher .....	5
4. Installation of CakePHP in production mode .....	5
4.1 Install packages .....	5
4.2 Download and unpack CakePHP .....	5
4.3 Permissions .....	6
4.4 Rewrite rules .....	6
4.5 Configure CakePHP .....	7
4.5.1 Change salt value .....	7
4.6 Database setup .....	7
4.6.1 Create database for CakePHP .....	7
4.6.2 Access to spooler .....	8
5. FreedomFone configuration in Cake .....	9
5.1 Configuration files .....	9
5.2 Cronjob .....	10

## 1. Data flow from FreeSWITCH to CakePHP

Let us start with a brief walk-through of the architecture from FreeSWITCH, via the dispatcher, the spooler and finally CakePHP.

Incoming SMSs or recorded audio messages are forwarded to FreeSWITCH, where events are triggered by the event manager.

The dispatcher connects and authenticates to FreeSWITCH, and subscribes to a set of Freedom Fone specific events.

When an event arrives to the dispatcher, it is converted to XML format. An XSL template is applied to the XML file, in order to filter out the data needed for the application, and drop the rest of the data.

A set of rules are applied to the XML output in order to match the event with a certain application. Finally, the event data is parsed to an SQL query and inserted into an application specific table in the spooler database.

The CakePHP application fetches data from the spooler using the spooler API (*spooler\_ff.php*). This can be done manually by using the refresh method, or by using a crontab. The event data is finally inserted in the freedomfone database, which is a part of the CakePHP application.

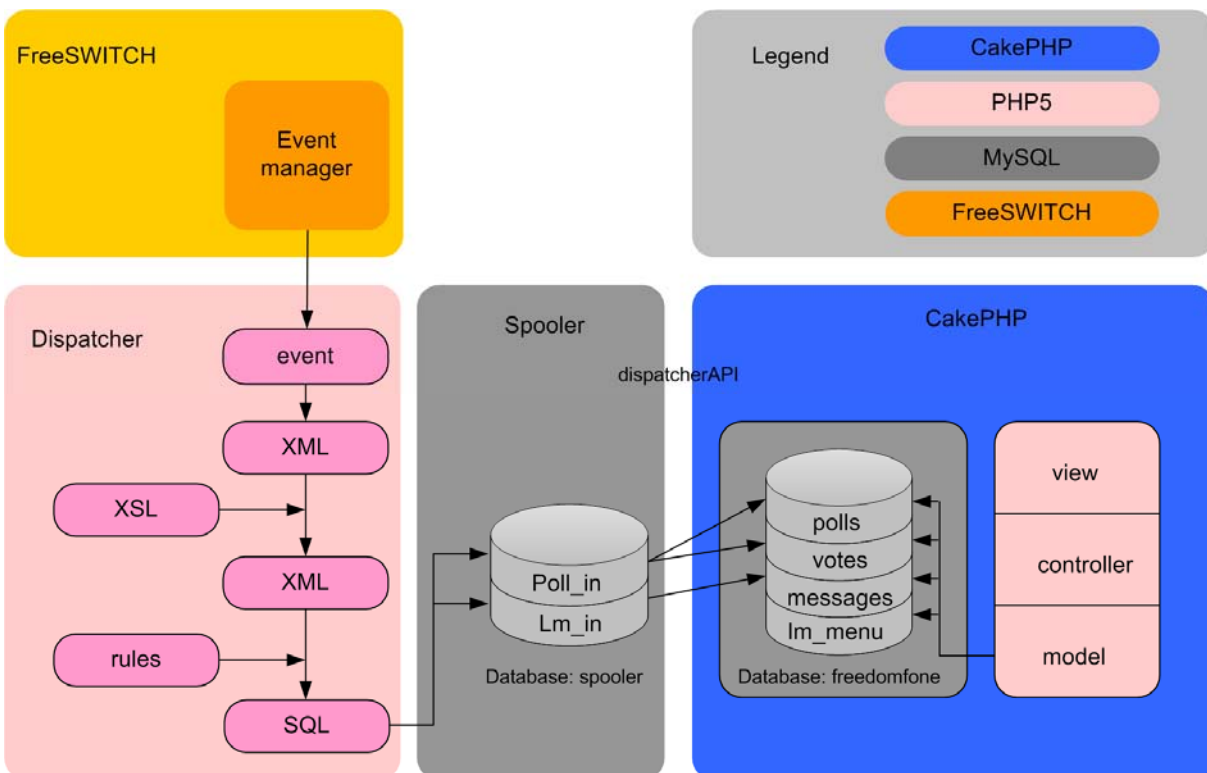


Image 1: Information flow from FreeSWITCH to the CakePHP applications.

## 2. Assumptions

This installation guide assumes that you have checked out the Freedom Fone SVN. We will assume that you have your local SVN repository under `/usr/local/src/pictus/`.

This installation guide will focus on the following two folders:

```
#!/usr/local/src/pictus/cake/app
#!/usr/local/src/pictus/dispatcher
```

## 3. The dispatcher

### 3.1 Database setup

The dispatcher exchanges information with the CakePHP applications through a set of database tables. For this purpose we will create a database called *spooler*

```
#mysqladmin -u root -p create spooler
```

Log in to the database spooler as root

```
#mysql -u root -p spooler
```

Set permissions for the user “dispatcher”. Replace `<password>` for your password.

```
mysql>GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, DROP, INDEX, ALTER,
CREATE TEMPORARY TABLES, LOCK TABLES ON spooler.* TO
'dispatcher'@'localhost' IDENTIFIED BY '<password>';
```

Reload the grant table

```
mysql>flush privileges;
```

The *spooler database* contains one (or more) application specific tables. A table can either contain incoming data (from FreeSWITCH to CakePHP), or outgoing data (from CakePHP to FreeSWITCH).

Dump the database schema (`/dispatcher/install`) into the *spooler* database.

```
#mysql -u dispatcher -p spooler < dbSchema_Spooler.sql
```

At the time of writing (September 2009), the spooler contains two tables:

**poll\_in:** contains incoming SMS data to the Poll application

**lm\_in:** contains incoming data for audio messages to the Leave a Message application.

### 3.2 Configuration file

The dispatcher configuration file can be found in the *config* directory. The settings that you should

modify, are marked with an asterisk (\*).

**Parameter: ParentXML**

Description: Parent tag used in the XML object created by the dispatcher.

Value: ff-event (do not change)

**Parameter: DefaultXSL**

Description: The default xsl filter to be used for incoming event.

Value: default.xsl (do not change)

**Parameter: DirXSL**

Description: Directory for xsl filters

Value: templates/ (do not change)

**Parameter: SocketParam**

Description: Socket parameters for connecting to FreeSWITCH

Values: (do not change)

```
host = 127.0.0.1
port = 8021
pass = ClueCon
timeout=3
stream_timeout = 0.5
```

**Parameter: DispatcherDB (\*)**

Description: Database settings for the spooler database. These values must reflect the settings you just did for the spooler database.

Values:

```
host= localhost
user= dispatcher
password= <your password>
database= spooler
```

### 3.3 Libraries

The libs directory contains php classes used by the dispatcher.

### **fs\_sock.php**

A class for FreeSWITCH event socket. The dispatcher uses this class to connect to FreeSWITCH, listen to events, and fetch data.

### **dispatcher\_ff.php**

Class for CakePHP to connect to the spooler and fetch incoming data or insert outgoing data.

## **3.4 Tests available**

The *tests* directory contains a set up of scripts that facilitates testing of the system

### **autoSend.php**

Connects to FreeSwitch and generates 1 SMS event every second. This script is useful for testing the poll application. To run the script:

```
#php5 autoSend.php <code> <message>
```

For example, to vote No is the “Election” poll, run

```
#php5 autoSend.php Election No
```

## **3.5 Run dispatcher**

The dispatcher can be found at

```
#cd /usr/local/src/pictus/dispatcher
```

To run the dispatcher, run the following command:

```
#php5 ./dispatcher.php
```

## **3.6 Test the dispatcher**

Run the autoSend.php script to auto-generate incoming SMS in FreeSWITCH.

Run the dispatcher, and see how the SMSs are received, processed and inserted into the spooler (poll\_in).

# **4. Installation of CakePHP in production mode**

## **4.1 Install packages**

Freedom Fone v.1 requires the following packages: apache2, php5, php5-xsl, mysql-server, php5-

## mysql and lame

```
#apt-get install apache2 php5 php5-xsl  
#apt-get install mysql-server php5-mysql  
#apt-get install lame
```

Restart the database server and the web server.

```
#!/etc/init.d/mysql restart  
#!/etc/init.d/apache2 restart
```

## 4.2 Download and unpack CakePHP

Download the latest stable release of CakePHP (cakephp.org).

Unpack it and save it outside of your SVN folder.

```
#wget http://cakeforge.org/frs/download.php/734/cake_1.2.5.tar.gz  
#tar -zxvf cake_1.2.5.tar.gz
```

Copy all Cake files, except the *app* directory, to your SVN

```
#cd cake_1.2.5  
#cp -Rf cake /usr/local/src/pictus/cake/  
#cp -Rf vendors /usr/local/src/pictus/cake/  
#cp -Rf index.php /usr/local/src/pictus/cake/  
#cp -Rf .htaccess /usr/local/src/pictus/cake/
```

Make a symbolic link from you SVN to the webroot directory. If you have any date stored under /var/www, make sure to move that elsewhere.

```
#cd /var  
#rm -Rf www  
#ln -s /usr/local/src/pictus/cake/ www
```

## 4.3 Permissions

Make sure that /app/tmp folder is writable by the webserver

```
chown -Rf www-data.www-data /var/www/app/tmp/
```

## 4.4 Rewrite rules

Freedom Fone v.1 requires the *rewrite module* of Apache to be enabled. Enable the module:

```
#a2enmod rewrite
```

Check that the module is enabled:

```
vi /etc/apache2/mods-enabled/rewrite.load
```

```
LoadModule rewrite_module /usr/lib/apache2/modules/mod_rewrite.so
```

Open the Apache config file and change the value of AllowOverride from none to all.

```
# vi /etc/apache2/sites-enabled/000-default

<Directory /var/www/>
    Options Indexes FollowSymLinks MultiViews
    AllowOverride all
</Directory>
```

Make sure that these *three directories* contain a .htaccess file with the following data:

**Directory: Cake**

```
<IfModule mod_rewrite.c>
    RewriteEngine on
    RewriteRule ^$ app/webroot/ [L]
    RewriteRule (.*) app/webroot/$1 [L]
</IfModule>
```

**Directory: cake/app**

```
<IfModule mod_rewrite.c>
    RewriteEngine on
    RewriteRule ^$ webroot/ [L]
    RewriteRule (.*) webroot/$1 [L]
</IfModule>
```

### Directory: cake/app/webdir

```
<IfModule mod_rewrite.c>
RewriteEngine On
RewriteCond %{REQUEST_FILENAME} !-d
RewriteCond %{REQUEST_FILENAME} !-f
RewriteRule ^(.*)$ index.php?url=$1 [QSA,L]
</IfModule>
```

Restart the webserver

```
/etc/init.d/apache2 restart
```

## 4.5 Configure CakePHP

### 4.5.1 Change salt value

Change the value of 'Security.salt' in *app/config/core.php* to a salt value specific to your application (any sequence of numbers and letters, around 40 characters long)

Save and exit

## 4.6 Database setup

### 4.6.1 Create database for CakePHP

Create a database called “freedomfone”. This is the database that CakePHP will use to store all its application specific data.

```
#mysqladmin -u root -p create freedomfone
```

Log in to it

```
#mysql -u root -p freedomfone
```

Set permissions for the user “freedomfone”. Replace <password> for your password.

```
mysql>GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, DROP, INDEX, ALTER,
CREATE TEMPORARY TABLES, LOCK TABLES ON freedomfone.* TO
'freedomfone'@'localhost' IDENTIFIED BY '<password>';
```

Reload the grant table

```
mysql>flush privileges;
```

Exit mysql

```
mysql>exit;
```

Dump the database schema (located under /app/install) into your freedomfone database.

```
#mysql -u freedomfone -p freedomfone < freedomfone_db_schema.sql
```

## 4.6.2 Access to spooler

CakePHP needs to access the *spooler* to push or pull data to/from FreeSWITCH. Each application accesses the spooler through an application specific table and user.

Grant permissions to user **poll\_in** to access the table **poll\_in** in the database **spooler** using <password>.

```
mysql>GRANT SELECT, INSERT, UPDATE, DELETE ON spooler.poll_in TO  
'poll_in'@'localhost' IDENTIFIED BY '<password>';
```

Do the same for the user/table **lm\_in**

```
mysql>GRANT SELECT, INSERT, UPDATE, DELETE ON spooler.lm_in TO  
'lm_in'@'localhost' IDENTIFIED BY '<password>';
```

All database tables and permissions are now completed. Next step is to make sure that CakePHP knows how to connect to the spooler.

## 5. Freedom Fone configuration in Cake

### 5.1 Configuration files

Freedom Fone v.1 uses two files for configuration; core.php and config.php. Core.php includes CakePHP specific settings, while config.php contains Freedom Fone specific settings.

Let's open the config.php file and have a look at the settings.

```
#vi /usr/local/src/pictus/cake/app/config/config.conf
```

You will find the following parameters:

```
poll_in  
lm_in  
lm_settings  
lm_default
```

Start by adjusting the database parameters (the password) for the spooler tables (poll\_in, lm\_in).

```
$config['poll_in']= array(  
    'host'      => 'localhost',  
    'user'      => 'poll_in',  
    'password' => '<password>',  
    'database' => 'spooler',  
);  
  
$config['lm_in']= array(  
    'host'      => 'localhost',  
    'user'      => 'lm_in',  
    'password' => '<password>',  
    'database' => 'spooler',  
);
```

*LM\_SETTINGS* defines path and folder names for uploaded messages and the IVR menu for the Leave-a-message application. Adjust the host variable to reflect your setup.

It is **not** recommended to change the other *LM\_SETTINGS*. If you do so, the corresponding change must be done in FreeSWITCH.

```
$config['LM_SETTINGS'] = array(
    'host'           => 'http://freedomfone.org',
    'path'           => 'freedomfone/leave_message/',
    'dir_messages'  => 'messages/',
    'dir_menu'      => 'audio_menu/',
    'dir_conf'      => 'conf/'
);
```

*LM\_DEFAULT* defines the default fallback messages of the Leave-a-Message IVR. If no alternative text message has been provided by the user, these are the fallback messages that will be synthesized by FreeSWITCH.

```
$config['LM_DEFAULT']= array(
    'lmWelcomeMessage'=> 'Welcome to Freedom Fone Leave a Message Service!',
    'lmInformMessage' => 'Record your message after the beep. To Finish, Press #',
    'lmInvalidMessage'=> 'Invalid option. Please try again.',
    'lmLongMessage'   => 'Your message is too long, to the point, please!',
    'lmSelectMessage' => 'To Play..press *. To Delete.. press 0. To Save..press 1',
    'lmDeleteMessage' => 'Your message has been deleted!',
    'lmsaveMessage'   => 'Thank you!',
    'lmGoodbyeMessage'=> 'Goodbye'
);
```

## 5.2 Cronjob

Every CakePHP application needs to pull/push data from the spooler on a regular basis. This is done through a “refresh” method in each controller.

```
#http://freedomfone.org/polls/refresh
#http://freedomfone.org/messages/refresh
```

These methods are called by a cronjob every  $x^1$  minute. They can also be executed manually by requesting the pages as above<sup>2</sup> (a blank page will be provided as result).

You can add a cronjob per application that calls the refresh method, e.g:

```
#crontab -e

*/1 * * * * /usr/bin/wget -O - -q -t 1 http://localhost/polls/refresh
*/1 * * * * /usr/bin/wget -O - -q -t 1 http://localhost/messages/refresh
```

---

1 The frequency of the refresh method is to be defined in the cronjob.  
2 Replace freedomfone.org with your own domain name.