

Open Administration for Schools 3.50

Administrator Documentation

Les Richardson

December, 2008

Contents

Installation	2
OverView for Newbies	2
File Locations	2
Program Requirements	3
Installation Overview	5
Step 1: Install Linux	5
Step 2: Copy the Files	5
Step 3: Configure the Webserver	8
Step 4: Set up MySQL	11
Step 5: Converting Student Data	13
Step 6: Configuration	13
Step 7: Getting Started on the Browser Side...	22
Install Appendix A - Setting up the PostgreSQL Database	23
Install Appendix B - Setting Up Webname Resolution	24
How to Upgrade Open Admin for Schools	26
Versions	26
Updating – OA 2.50	27
Updating – OA 2.40	29
Model OpenAdmin Installation - Single Site with SSL	32
Customize OA - Adding New Fields to Students and Staff	42
Overview	42
Extending Student Demographics	42
Security	45
Web Security	45
Maintenance	46
Multiple Servers	46

Installation

Version Information

- OA 3.50 – December 2008
- OA 3.25 – August 2008
- OA 2.50 – December, 2007 - new SSL single site instructions based on Debian/Ubuntu by Andy Figueroa. Other directions updated to match.
- OA 2.30/2.40 – July, 2007 – updated
- OA 2.25 – May, 2007 – updated
(older entries removed)

Please read this over before trying to do too much!

OverView for Newbies

Each school that is running Open Admin would normally have three virtual websites...one for the school office, one for teachers, and one for parents/students. The html and cgi scripts for all sites for each school are located in a common folder, normally under the school name. A new version capable of running on a single SSL protected website is now available starting with the 2.50 release. Version 3.25 adds authentication for teacher site directly from staff table.

The administration or office site has it's files in **admin** for the HTML files and **cgi** for the matching cgi scripts. Similarly the teacher site directories are called **tadmin** and **tcgi**. So also, **phtml** and **pcgi** for the parent site.

If you are limited to having **one** site only (as some Australian schools are), it can also be configured to run on this single site (although this will require more work until the new single site release is available).

File Locations

Everything is arranged like this (/opt/openadmin can be replaced with another file path on your server):

/opt/openadmin/stpeter/admin – for St Peter School main admin site
/opt/openadmin/stpeter/cgi
/opt/openadmin/stpeter/tadmin – for St Peter School teacher site
/opt/openadmin/stpeter/tcgi
/opt/openadmin/stpeter/padmin – for St Peter School parent site
/opt/openadmin/stpeter/pcgi

/opt/openadmin/bruno/admin – for similarly for Bruno School...
/opt/openadmin/bruno/cgi
/opt/openadmin/bruno/tadmin
/opt/openadmin/bruno/tcgi
/opt/openadmin/bruno/padmin
/opt/openadmin/bruno/pcgi

There is also a `global.conf` configuration file listing all of the schools in a division, so that withdrawn students from one school can be easily enrolled in another (ie. feeder schools). Even single schools have to add their database name to this file. This is one of the installation problems people have had. The location of this directory is set in the `admin.conf` file and would normally be your `schools etc` folder.

If you are running the IEP (Special Education) site, you can install those files in:

/opt/openadmin/iep/admin – for the special needs website.
/opt/openadmin/iep/cgi

Program Requirements

One of the design goals of OA was to make it relatively easy to install and lightweight to run both in hardware and bandwidth demands so that even satellite connections with long latency will work. This has already been demonstrated in many remote schools.

- A *webserver* supporting virtual hosting and/or SSL (for encrypted communication over the Internet). This is normally the freely available Open Source Apache web server software.
- A *Perl interpreter*; a recent version would probably be good. The 5.8.10 release is current.
- *MySQL or PostgreSQL* - freely available SQL databases. For MySQL, a 5.x version should be used. One should be conservative in the use of *bleeding edge* software for a database. The current Open Admin 2.xx release supports MySQL 5.x (with new reserved words). Use the appropriate *blank.sql* database from the `utility/sql` folder of the download.
For PostgreSQL, a new stable version should be used. However, Pg has not been used for a couple of years in normal testing.
- A standard *TeX* installation including `pdflatex` (normally installed with Linux distributions). The new TeX Live installation should/could be used from Tug.org (TeX User Group). This will run on all Unix-like Operating systems (including Mac OSX) and also on MS Windows servers.
- Accessory utility software. The student image management functions will require an ImageMagick installation (for the `convert` program to convert image sizes.
- LibXML2 (from www.xmlsoft.org) - an XML library in C. This is required to parse the FET scheduler teacher xml file to import your timetable into OA. If not using the FET scheduler (and not a Saskatchewan school using the SDS system), you do not need this

xml parser, currently. This will likely change as new reporting functions require xml markup.

The following perl modules will also be required:

- CGI
- DBI (and the DBD drivers to connect to MySQL or Pg)
- Text::CSV_XS - for conversion to/from CSV format.
- XML::LibXML - a perl module that uses LibXML2 library (if using XML).
- Date::Business - for date functions such as attendance reports.
- Time::JulianDay - other date calculations
- FreezeThaw (install this **before** CGI::Session since it is required by CGI::Session).
- CGI::Session - session management for the teacher site.
- Data::Password - used for checking staff passwords for security.
- Crypt::GeneratePassword - used to generate staff passwords when adding new staff.
- Number::Format - a rounding function used in place of sprintf.

In order to install the perl modules, the easiest approach is to use the CPAN method. The CPAN (Comprehensive Perl Archive Network) is a worldwide network of servers from which you can *automatically* download and install the modules. The first time this is used you configure the server locations you want to use, etc. The defaults suggested work fine.

One syntax is as follows:

```
perl -MCPAN -e 'install PERLMODULE'
```

where PERLMODULE is the name of the module to install... such as FreezeThaw. Case appears to be an issue. If you can't find the right name go to the CPAN search site (<http://search.cpan.org>) and have a look around.

If you are using the software in Saskatchewan and would like to communicate directly to the provincial SDS system, then additional software and perl modules are required since we create a secure HTTPS connection to the Sask Learning server and send and receive transactions in an XML format.

You will also need:

1. OpenSSL (from www.openssl.org) - SSL library required for https.
2. Perl Modules:

-
- (a) `Bundle::LWP` (which includes `LWP::UserAgent`, `HTTP::Headers`, and `HTTP::Request::Common`) - used to connect via https.
 - (b) `Crypt::SSLeay` (for SSL connections)
 - (c) `XML::Writer` - used to write the output XML.
 - (d) `XML::Writer::String` - holds xml output in a string.
 - (e) `Data::UUID`

Installation Overview

The installation process consists of the following steps:

1. Install Linux
2. Copy the files.
3. Configure the Webserver.
4. Add the Data to the database.
5. Import Student Data.
6. Edit the Configuration.
7. Get Started (on the Browser Side)

Step 1: Install Linux

Download the latest Ubuntu Server or Debian Installation (or Red Hat or your Linux distribution of choice). Make sure that you add Perl, MySql (or PosgreSQL), and TeX (if available). The current TeXLive installation may be a better choice, since other versions have limited support.

If creating an Open Admin only server, then a minimal, lightweight installation would be a good choice. The idea is to create a server appliance that needs only minimal, if any, management. As well, only a minimal hardware platform will be required if only a few schools are running on it.

Step 2: Copy the Files

1. Download the latest Open Admin version from <http://richtech.ca/openadmin> into `/tmp` (or some other commonly used file location). If you download to your workstation desktop you can copy to your server using a secure copy program like `scp`, or simply copy it to a flash drive and copy onto the server in this way.

2. Untar the download:

```
tar xvzf openadmin-3.xx.tgz
```

This will create `/tmp/openadmin-3.xx` with:

- `/tmp/openadmin-3.xx/school` - containing the main files required for each school site
- `/tmp/openadmin-3.xx/utility` - contains files needed for installation. These first 2 are the main directories needed for installation.
- `/tmp/openadmin-3.xx/iep` - contains the IEP special education website application. This is the IEP site. It contains a `cgi` folder for the perl scripts and a `www` folder for the html files.
- `/tmp/openadmin-3.xx/docs` - contains a Readme file telling where *real* docs are found. Also contains the Gnu GPL license file.
- `/tmp/openadmin-3.xx/sis` - Contains a central office application to see all school enrollments, search for students in all schools, ethnic enrollments, etc.

3. Create a directory to hold the files for a school (or several) if it doesn't already exist.

```
mkdir /opt/openadmin
```

We will assume that you are using `/opt/openadmin`. Just substitute another filepath.(ie. `/opt`, `/Library/Webserver/` if OSX, etc.)

4. Copy the directory containing the main OA school site files into this new directory with a short single word for the school:

```
cp -r /tmp/openadmin-3.xx/school /opt/openadmin/stpeter
```

where `stpeter` is replaced with your school name. (if it's not St. Peter!) You now have the main school files in place to run the 3 main St. Peter websites (main, teacher, parent).

5. Copy the school folder as many times as required to get one school folder for each school.

(ie. `cp -r /tmp/openadmin-3.xx/school /opt/openadmin/stmary`)

This assumes a multi-school installation. Just do this once if a single school.

6. Check/change the ownership and permissions on the files/folders. The files in the downloaded tar distribution are set to `root` ownership and members of `root` group.

Change the ownership of the copied files/directories to that of the user and group that the webserver runs as. In Debian/Ubuntu this is `www-data` for both user and group. For Slackware this is `webuser` and `webgroup`. For Red Hat this is `apache` for both. Make sure that you are in the `/opt/openadmin` directory so that you change the ownership of everything below this level in the filesystem.

```
If www-data:www-data then chown -R www-data:www-data *
```

```
If apache:apache then: chown -R apache:apache *
```

If this seems too loose in terms of security (since you may have users on this system), ownership may be left at `root/root` for scripts and just make sure that the running scripts

can create required LaTeX files in the folder where the script runs that is creating it. If not set correctly, you'll see entries in the error log about the script not being able to create tex files and no pdf files (such as student roster reports and report cards) will be created. Get into the habit of checking the webserver logs for each of the sites. Again, the location of these log files varies depending on the linux distribution that you are using. On Debian/Ubuntu servers, it is located in `/var/log/apache2` while Red Hat stores them in `/var/log/httpd`.

A running script must also be able to copy generated PDF files into the *download* folder in *admin* or *tadmin* of the main or teacher site. Scripts must also be able to change the *term* file, *studentnumber*, *receiptnumber* files in the school level *etc* folder as well. You will have to change the permissions on these files and directories if you don't reset the ownership to that of the User Id (UID) that the webserver runs as (as outlined above).

7. Copy your School Logo file (if you have one) into the school directory `admin/images`. For example, `/opt/openadmin/stpeter/admin/images` folder and name it *logo.gif*. It should already be in GIF format! You should replace/erase the existing files (*logo.gif* and *logotn.gif*).

The size of your *logo.gif* should be approximately 150-200 pixels high or so (and a reasonable width). Excessive width (more than 300 pixels or so) may cause layout problems on smaller screens. Pick a size that is pleasing when displayed on the main page. This should give a good starting point. A smaller version of the school logo (about 100 pixels high) should be saved and renamed to *logotn.gif* (Logo Thumbnail) This is the logo visible on the attendance, discipline, report card pages, etc. To conserve bandwidth, these should be as small as possible. The other 2 small image files *tabactive.gif*, *tabmain.gif* (total 2k in size) implement the menu system on the admin site.

8. Edit the configuration file called `admin.conf` located in the school `etc` directory. For example, (`/opt/openadmin/stpeter/etc`) This is read by all of the scripts (`cgi`, `tcgi`, `pcgi`) and is required to access the database, put in the right navigational links, etc.

Three important entries of the *admin.conf* configuration file are for user, database, and password. The user and password would be set when installing the database. More on this below. The fourth entry, *dbtype* selects whether you are using MySQL(`mysql`) or PostgreSQL (`Pg`). The rest of the entries in this file should be obvious from the comments around them. More on this below, also.

There is also a file called *studentnumber* located in the school `etc` folder that stores **the next available student number**. This should be set once students have been imported into this system. It stores the **next** available student number. For small standalone schools, a 3 digit starting number would be a good choice (ie. 100). This is the default. Other files in this folder... *term*, *receiptnumber* must also be writeable by the webserver user.

If you are installing multiple copies of openadmin for division schools, then **blocks of numbers** should be set aside for each school. There is currently **no** checking for overlap of numbers, so give each lots of room. This will give a unique student number to each student in the division, thus enabling easy data transfers and enabling use of the shared IEP system (which require unique student numbers for all students in that system and still requiring access back to their home school database).

Make sure that the *download* folders in *admin* and *tadmin* for each school are writeable by the webserver user (the user the webserver runs as) so that scripts, which run with the same UID, can place created files there for downloading (mentioned above). A background script running daily (ie. a cron job) should be set to empty out these folders.

Step 3: Configure the Webserver

These are directions to set up the Apache webserver. If using other webserver software, you're on your own. Look for the virtual website documentation for that server. (It should have *name based* virtual site capability rather than IP based virtual sites, although either method could be used.)

There are several basic variations on configuration depending on your version of Linux:

1. **Debian / Ubuntu** – In these distros the Apache configuration files are located in `/etc/apache2`.

In the utility folder of your download, there is a file called `httpd-debian.conf`. Edit this file and search and replace the 'SCHOOLNAME' text into the folder name of your school (ie. `stpeter`). As well, change the `ServerName` entries to the domain names for your websites. This should provide a good starting point for your school sites.

Copy and Rename this file into `/etc/apache2/sites-available`. Then create a symbolic link in the `/etc/apache/sites-enabled` folder to point to this file. Alternatively, just copy the file directly into `sites-enabled`.

2. **Red Hat and Friends** – In these distros the Apache configuration files are located in `/etc/httpd`.

In the utility folder of your download, there is a file called `httpd-redhat.conf`. Edit this file and search and replace the 'SCHOOLNAME' text into the folder name of your school (ie. `stpeter`). As well, change the *ServerName* entries to the domain names for your websites. This should provide a good starting point for your school sites.

Copy this file into `/etc/httpd/conf` and rename it with a school name (ie. `stpeter.conf`). Then edit the `httpd.conf` file in the same folder. At the bottom of the file (where Virtual Sites are normally configured), add a line: `Include conf/stpeter.conf` where `stpeter.conf` is the name of the file renamed above. The text `NameVirtualHost *:80` should also be added above the `Include` (if not present somewhere above the `Include` line). This turns on virtual hosting.

3. **Other Distros** – More General Instructions

The webserver settings are located in a file called **httpd.conf**. This file is normally located in the server's, `etc/apache2` or `/etc/httpd` directories.

The virtual websites are normally at the bottom of this file. External config files are the easiest way to configure OA, and can be pulled in with *include* statements in the `httpd.conf`. It would look like:

```
NameVirtualHost *:80
Include conf/myhttpd-school.conf
Include conf/another-school.conf
```

The *NameVirtualHost* directive turns on name based virtual hosting so that web browsers can send a site name in an http header and the webserver will provide a site based on that name.

The *conf* folder in the Include statement is added since the include file is located relative to DocumentRoot, described below). Examples of this external file is found in the utility directory and are called **httpd-xxxx.conf**).

Here is an example of an httpd-*.conf include file for St Peter School. It assumes that the NameVirtualHost *:80 has been placed in the main httpd.conf file before the Include statement that loads this school configuration file. It also includes teacher authentication from staff table rather than file.

```
# Start of Admin Site
<VirtualHost *:80>
    ServerAdmin root@localhost
    DocumentRoot /opt/openadmin/stpeter/admin
    ServerName stpeter-admin.mysd.ca

    # Make .shtml files be parsed
    AddType text/html .shtml
    AddHandler server-parsed .shtml
    DirectoryIndex index.shtml

    ErrorLog /var/log/httpd/spa-error-log
    CustomLog /var/log/httpd/spa-access-log common
    ScriptAlias /cgi-bin/ "/opt/openadmin/stpeter/cgi/"

<Directory /opt/openadmin/stpeter/admin>

    Options +Includes # turns on Server Side Includes (SSI)
        # to parse date (and user in teacher)

    AuthType Basic
    AuthName Admin
    AuthUserFile /etc/httpd/auth_users/stpeteradmin
    <Limit GET POST>
        require valid-user
    </Limit>
</Directory>

<Directory /opt/openadmin/stpeter/cgi>
    AuthType Basic
    AuthName Admin
    AuthUserFile /etc/httpd/auth_users/stpeteradmin
    <Limit GET POST>
        require valid-user
    </Limit>
</Directory>
```

```
</VirtualHost>
```

Some key points being:

DocumentRoot: The path to your html files for this virtual site.

ServerName: the name the virtual site should be identified as.

ScriptAlias: where your perl (.pl) scripts are located for this site.

The server side includes are turned on to process the main admin .shtml file so that it will correctly display the current date. This is done above with the "Options +Includes". This is also true of the teacher site to allow the Apache user to be seen.

You can test this configuration with the Apache controller.

```
apachectl configtest
```

You may have to include it's full path, typically /usr/sbin/apachectl. It should respond with an OK. If not, fix the problem and try again. If all is well, then you may restart the web server. One method is:

```
apachectl graceful
```

Setup Access Control

There are 2 methods to control access to these sites:

1. **File Access Control** - files contain userid and password information that control access to particular site.

The password file(s) that control access to the virtual site(s) are created using the *htpasswd* program. Currently this is only the main admin site since the teacher site can be configured to use the values in the staff table for authentication / authorization.

A folder named *auth_users* must be created in /etc/httpd or /etc/apache2 (if it doesn't already exist) and the *htpasswd* program used to create entries in the *auth_users/stpeteradmin* file. Only these users will have access to the admin site. Normally only a few passwords are used for these sites. Read the man page for *htpasswd* program to see how to do this.

The teacher site normally uses database authentication, but a file based authentication like the main admin site may be used also.

2. **Mysql Access Control** - user information is read from the staff table. The file called *httpd-teacherauth.conf* in the utility folder contains the directory commands that may be copied in place of the teacher site directory commands in your *school.conf* (*stpeter.conf*) file. This will allow teachers to log into the teacher site using their own userid and password. The *mod_auth_mysql* module will also have to be installed and working in your Apache installation.

Setup Log Files

Next are the log files which record site activity. The log files are very important to signal any problems or errors. In the event of *white screens* with nothing on them (Secretaries notice

things like this... I call them WSOD... White Screens of Death),

Look at the Log Files!

Normally we create a `/logs` symbolic link to point to where the log files are actually located (ie. `/var/log/apache2/`). This simplifies the query for problems:

```
tail /logs/stpetera-error.log - view main site error log.
```

Also note, that if doing updates, you can use symbolic links in Linux filesystems (Windows NTFS filesystems have limited symbolic link support, as far as I know. This would allow you to install an updated version of OA in a different folder and then use a symbolic link to point to this new folder, rather than another older folder. If you have installation problems (and your site is very active, as it normally is with school admin), you can quickly switch back to older code (as long as the table structures haven't changed with the newer version).

These directions assume that Apache has its configuration installed in `/etc/httpd` or `etc/apache2`. These are the normal locations for most Unixlike operating systems.

Step 4: Set up MySQL

See the Installation Appendix for the PostgreSQL instructions if using that database engine.

I will assume that you have a basic installation of MySQL and that the MySQL engine is running. (If you're not sure, do a `ps aux` and you should see several `mysql` processes running). Normal installation of Linux should give you a working copy of this. You may also download and install versions from www.mysql.com as well.

1. Create a database for the application:

```
mysqladmin create -pROOTPASSWORD myschool
```

where *rootpassword* is the root password for the MySQL server (which is the not same thing as the root password for the entire computer) and *myschool* is the name of the database to be created. This is the name that will be placed in the `/cgi/admin.conf` file with `$dbase = .`

2. Now create a non-root password for this school by using the `mysql grant` command. First go into the `mysql` client by typing:

```
mysql -prootpassword -u root myschool
```

Then type the following to create an entry so that you may use your own user and password to access the database:

```
grant all on myschool.* to myuser@localhost identified by 'mypassword';
```

where *myschool* is the name of the database, and *myuser* and *mypassword* are the user and password to get access to the program. They do not have to exist on the system as a real user, but are merely used by the scripts. These are the `$user` and `$password` variables in the `admin.conf` file. You can now use this userid and password to access your database.

A new mysql identity (*rouser* - read only user) should be created that has only read (sql select) capability for that school's database. This is particularly important for use with the publicly accessible parent page/site that allows students and parents to view attendance, report and gradebook marks. Other scripts will use this as well.

```
grant select on myschool.* to rouser@localhost identified by 'ropassword';
```

where *rouser* and *ropassword* are the values also entered in the admin.conf configuration file for \$rouser and \$ropassword.

A utility called `mysql_setpermission` can also be used to set access control, like the grant command.

Another mysql identity (which I call *global*) with read capabilities for all of the school databases should be created, if using the SIS site (central office). This will allow a *global* user to read all of the school databases, but not change them. This is required for the central office application and some global view scripts.

3. Create the database tables

Run the following command using the file in the utility/sql folder of the download called *blank.mysql.xxx*. It contains the SQL statements necessary to create the entire database (where xxx is the latest version).

```
mysql -pmypassword -u myuser myschool <blank.mysql.xxx
```

If all went well, and no errors were shown, you should now have tables in your database, ready to accept records. You can verify this by doing a *show tables* command when in the mysql client.

The **meta** table (data dictionary) is initially blank. The meta.sql file in the utility folder contains starting student and staff table defaults for this. Run this just like you did for the blank.mysql file above and it will put in defaults, etc. required by the templates used with these 2 tables for forms, etc. There are 2 versions, one for Canada and one for the US. Choose one.

You now add the *studentall* table. It is a union of both current students and withdrawn students. If using MySQL 5.x, then use the *studentall.sql* file in the utility/sql folder. If using MySQL 4.x still, then use the *studentall.mysql4* file. Do this in the same way as you did previously with the blank database file.

If you add more fields to the student and/or staff tables, you can update this table by running the Meta Update script located on the Start/End of Year page of the admin site (See also the section on customizing OA).

If you are using the Special Needs application, it has a different database design. It is installed in the same fashion as outlined above, using the file called *iepdataXXX.sql* located in the utility/sql folder. It is inserted into an *iep* database (or whatever you would like to name it) created with mysqladmin (as above).

The **central office application** does not require any database, since it queries all of the schools (and iep if it exists) databases.

Your database setup it now complete.

Step 5: Converting Student Data

The student information can now (OA 3.50+) be uploaded from a CSV spreadsheet file using a new import script on the Export page of the main admin site. This is the easiest approach.

The following information outlines how to do this using another utility script (`studentupload.pl`) from the command line. It is more complex, but is more powerful since it can also munge the data and rearrange information within fields, etc.

The import script, `studentupload.pl`, must be edited so that what the script expects matches the input file format. This kind of file format can be exported from various spreadsheet and database programs. If starting from scratch, it might be easiest to quickly type in required student information into a spreadsheet, and then import that (either with `studentupload.pl` or via the web import script on the Export page.

Run it (assuming your data file is `mydata.csv`) with:

```
./studentupload.pl mydata.csv
```

This should correctly put the students into the student table in the school database. In the event of errors, simply empty the student table and try again. You can do this with:

```
mysql -pmypassword -u myuser myschool
```

Once in the MySQL client, enter the SQL command to empty the student table (without deleting it):

```
delete from student;
```

and try, try again. It will take a couple of times at least to get the data in a format that you like (or will accept). You can also edit the CSV file in a spreadsheet before importing it here to quickly get the data in the form that you would like.

Next, set the `etc/studentnumber` file to the student number for the next student enrolled (as mentioned above). This will ensure that once the secretar(y/ies) start adding students, that student numbers are assigned starting with the next available number.

Step 6: Configuration

OA is configured by editing configuration files located in the *etc* folder of each school.

The main configuration file is called **admin.conf** and is read by all scripts from main admin (cgi), teacher (tcgi), and parent (pcgi). Setting correct values is very important since it affects all program operations.

New options are now placed at the top of the file, making it easier to edit the file when doing upgrades from previous OA versions.

```
# -- New in 3.50 ----  
# password settings used by Crypt::GeneratePassword, also Data::Password  
$password_minlen = 5;
```

```
$password_maxlen = 6;
$password_lang = 'en';
$password_signs = 0;
$password_caps = 1;
$password_minfreq = .001;
$password_avgfreq = .001;

#--- New in 2.50 -----
# For translation purposes, this has to be in configuration...
# Files now use this to display Months, Days of Week, etc.
@month = ('','January','February','March','April','May','June','July',
  'August','September','October','November','December');
@s_month = ('','Jan','Feb','Mar','Apr','May','Jun','Jul','Aug','Sep','Oct','No

@dow = ('','Sunday','Monday','Tuesday','Wednesday','Thursday','Friday','Saturday')
@s_dow = ('','Sun','Mon','Tue','Wed','Thu','Fri','Sat');

# Location of special ed config file in your filesystem
# Actual filesystem location of the file ON YOUR SERVER.
$iepdir = '/opt/openadmin/iep';

# Paper Size: Letter, Legal, A4 (entered as: letterpaper, legalpaper, a4paper)
$defaultpapersize = 'letterpaper';

# ---- End of New Section for 2.50 -----

# Basic Constants for the database
$user = 'USER';
$password = 'PASSWORD';
$dbase = 'DATABASE';
$dbtype = 'mysql'; # mysql or Pg (postgresql; watch case!)

# SQL Syntax settings for mysql or Pg.
$sql{default} = 'NULL'; # Change to DEFAULT for Pg.
$sql{like} = 'like'; # Change to ILIKE for Pg; case insensitive searches.

# Language Settings
$langcode = 'en';
$language = 'English';

# Full Text Name of School and Address
$schoolname = 'YOURSCHOOL';
$schooladdr1 = 'Addr1';
$schooladdr2 = 'Addr2';
#$schooladdr3 = 'Addr3';
```

```
$schoolcity = 'City';
$schoolprov = 'Province';
$schoolpcode = 'Pcode';
$schoolphone = '(306) 123-4567';
$schoolfax = '(306) 123-4568';
$schoolemail = 'contact@YOURSCHOOL.ca';

# CGI areas - normal file paths; cgi is main admin, tchcgidir is teacher
# These are the pathnames ON YOUR SERVER! (not part of the URL)
$cgidir = '/opt/openadmin/YOURSCHOOL/cgi';
$tchcgidir = '/opt/openadmin/YOURSCHOOL/tcgi';
$parcgidir = '/opt/openadmin/YOURSCHOOL/pcgi';

# CGI URL's - for self referencing scripts
# part of Apache configuration, not server filesystem.
$cgiurl = '/cgi-bin';
$tchcgiurl = '/cgi-bin';

# Picture Directories and URL's
# dir is actual filesystem pathname, url is apache config.
$picdir = '/opt/openadmin/YOURSCHOOL/admin/pic-big';
$picdirurl = '/pic-big';

# Picture area for thumbnails
$ndir = '/opt/openadmin/YOURSCHOOL/admin/pic-sm';
$ndirurl = '/pic-sm';

# Full path to download folders; used by scripts to copy files there.
# Actual pathname ON YOUR SERVER!
$downloaddir = '/opt/openadmin/YOURSCHOOL/admin/download';
$tchdownloaddir = '/opt/openadmin/YOURSCHOOL/tadmin/download';

# Web location of download (put into URL)
# Apache configured values. Part of the URL.
$webdownloaddir = '/download';
$tchwebdownloaddir = '/download';

# CSS File locations and name
$css = '/admin.css';
$tchcss = '/tadmin.css';
$parcss = '/padmin.css';

# Location of global config file to integrate school databases.
# Actual filesystem location of the file ON YOUR SERVER.
$globdir = '/opt/openadmin/global';
```

```
# Path for PDFLatex Executable
$pdflatex = "/usr/bin/pdflatex";

# Administrator Contact Info; used by scripts for error contacts.
$adminname = "ADMIN";
$adminemail = "ADMIN\@YOURBOX";

# Return Address - address of school, or secretary/administrator
# Currently used by Sask SDS code for enrollment resolution.
$returnaddress = "SECRETARY\@YOURSCHOOL";

# Appended to top of generated HTML.
$doctype = "<!DOCTYPE HTML PUBLIC \"-//W3C//DTD HTML 4.01 Transitional //EN\"
\"http://www.w3.org/TR/html4/loose.dtd\">";

# Character Type
$chartype = '<meta http-equiv="Content-Type" content="text/html; charset=ISO-8

# CSS Files - Webspaces Links
$g_maincss = "<link type=\"text/css\" rel=\"stylesheet\" href=\"admin.css\">";
$g_tchcss = "<link type=\"text/css\" rel=\"stylesheet\" href=\"tadmin.css\">";

# Top Level HTML Pages - URL Links
$homepage = "/index.shtml";
$discpage = "/discipline.html";
$reppage = "/repcard.html";
$attpage = "/attendance.html";
$exppage = "/export.html";
$eoypage = "/eoy.html";
$schpage = "/schedule.html";
$feespage = "/fees.html";

$tchpage = "/index.shtml";

# Global Color Settings: used by gradebook groups
$g_color[0] = '#8FA';
$g_color[1] = '#FF8';
$g_color[2] = '#AAF';
$g_color[3] = '#F8F';
$g_color[4] = '#FFA';
$g_color[4] = '#FAF';
$g_color[5] = '#AFF';
$g_color[6] = '#FAA';
$g_color[7] = '#AFA';
$g_color[8] = '#AAF';
$g_color[9] = '#AAA';
```

```
# Attendance Settings for Periods per Day (ppd) for each grade.
# Put your grades in the curly braces and
# the number of periods in the quotes.
$g_ppd{PK} = "2";
$g_ppd{K} = "2";
$g_ppd{1} = "2";
$g_ppd{2} = "2";
$g_ppd{3} = "2";
$g_ppd{4} = "2";
$g_ppd{5} = "2";
$g_ppd{6} = "2";
$g_ppd{7} = "2";
$g_ppd{8} = "2";
$g_ppd{9} = "2";
$g_ppd{10} = "2";
$g_ppd{11} = "2";
$g_ppd{12} = "2";

# The school year;
$schoolyear = "2008-2009";
$schoolstart = "2008-08-28";
$schoolend = "2009-06-27";

# School Terms
$g_termstart{1} = "2008-08-28";
$g_termend{1} = "2008-11-16";
$g_termstart{2} = "2008-11-19";
$g_termend{2} = "2009-03-07";
$g_termstart{3} = "2009-03-10";
$g_termend{3} = "2009-06-27";

# Days Per Cycle; only needed for Scheduling.
$daysPerCycle = 6; # assume a 6 day cycle.

# Dates to reset the 'dayInCycle' to throughout the year. If there are
# no reset values here, then the schoolstart value is assumed to be Day
# 1, and days loop through the cycle from there, omitting any dates set
# for the school year with a 'N' in the DayInCycle field (on Start/End
# of year page) under Date Management.

#$resetDate{'2008-08-31'} = 1;

# These are ONLY needed if your school is a K-12 school and
# has two (or more) different school term dates in same school.
# (ie. 3 terms for K-6, and 4 terms for 7-12). All programs support
# the structure above this ($g_termstart{1} = 'date') in any case.
```

```
# You may use other values rather than 'e' and 'h' as long as the
# $g_termtype{Grade} = 'MyValue' matches
# $g_termstart{MyValue}[TermNumber] = 'TermStartDate'

$multiTrack = 0; # 1=yes, 0=no

# Scripts supporting this date structure are: (repcard card
# script... soon, then SDS scripts, etc.)

# $g_termstart{e}[1] = "2008-08-27";
# $g_termend{e}[1] = "2008-11-19";
# $g_termstart{e}[2] = "2008-11-22";
# $g_termend{e}[2] = "2009-03-28";
# $g_termstart{e}[3] = "2009-03-31";
# $g_termend{e}[3] = "2009-06-28";

# $g_termstart{h}[1] = "2008-08-27";
# $g_termend{h}[1] = "2008-11-19";
# $g_termstart{h}[2] = "2008-11-22";
# $g_termend{h}[2] = "2009-01-28";
# $g_termstart{h}[3] = "2009-01-31";
# $g_termend{h}[3] = "2009-01-28";
# $g_termstart{h}[4] = "2009-02-01";
# $g_termend{h}[4] = "2009-07-28";

# $g_termtype{K} = 'e';
# $g_termtype{1} = 'e';
# $g_termtype{2} = 'e';
# $g_termtype{3} = 'e';
# $g_termtype{4} = 'e';
# $g_termtype{5} = 'e';
# $g_termtype{6} = 'e';

# $g_termtype{7} = 'h';
# $g_termtype{8} = 'h';
# $g_termtype{9} = 'h';
# $g_termtype{10} = 'h';
# $g_termtype{11} = 'h';
# $g_termtype{12} = 'h';

# Saskatchewan Specific constants
#-----
$schoolnumber = "123456";
# An english stream program in Saskatchewan; used by export/sds program.
# Note that each student record also has a program code. This is just a
# default so no need to add all records, unless special ed, etc.
$program = "10";
```

```
# Used by Saskatchewan SDS code to disable functions for non-native schools.
# Disables reserve residency, band affiliation, and treaty number xml updates.
$nativeschool = '0';

# Main controls for Sask Learning
$url = 'https://www.sasked.gov.sk.ca/spls/sdsxml/pkgb_xml_interface.sl_message
$sds_userid = 'USERID';
$sds_password = 'PASSWORD';

$maxfilesize = 30000; # Max Filesize for transfers.
# SL has 32k limit for xml file transfers. Must be too big for oracle...
# 30k setting above it relatively safe limit. Don't go higher unless
# you know what you're doing.

# SDS XML Schema Declarations
$xmlns = "http://www.sasked.gov.sk.ca/xsd/sl/1.x/SLMessage.xsd";
$xmlnxsxi = "http://www.w3.org/2001/XMLSchema-instance";
$xsischemaLocation="http://www.sasked.gov.sk.ca/xsd/sl/1.x/SLMessage.xsd
http://www.sasked.gov.sk.ca/xsd/sl/1.x/SLMessage.xsd";
#----- End of Sask Specific Constants -----

# --- Attendance Settings - currently used in teacher attendance entry.

my $g_maxterms = 12; # maximum # of terms to search for.
my $g_attendview = 16; # Number of previous attendance records to see;
my $g_teachview = 12; # Number of teacher attendance records to see.
my $g_allowedit = 1; # Comment out to disable teacher editing of att records.

# Reason Categories.... the Unexcused ones, and Late/Absent Groups...
# currently used by attendance profiles...
$absentUnexcused = 'Absent Unexcused';
$lateUnexcused = 'Late Unexcused';
$lateString = 'Late';
$absentString = 'Absent';

# Reasons for Attendance
# Note: "Absent" and "Late" MUST be used to be correctly tagged by reports.
# Case is significant. We have several classes: Absent Unexcused; Absent
# for a variety of reasons. (Absent Illness, Absent Moved, etc.). If
# excused but for another reason it's Absent Excused. Similar for Lates.

$attend[0]="Absent Unexcused";
$attend[1]="Absent Excused";
$attend[2]="Absent Illness";
$attend[3]="Absent Moved";
$attend[4]="Absent Suspension";
$attend[5]="Late Unexcused";
```

```
$attend[6]="Late Excused";

# Array to list reasons to ignore for perfect attendance; cgi/rptattperf.pl
@ignore_reason = (5, 6);

# Trigger Points Match Reasons Above.
# Values assigned to each type of missing attendance reason.

$points[0] = "3";
$points[1] = "0";
$points[2] = "0";
$points[3] = "0";
$points[4] = "0";
$points[5] = "1";
$points[6] = "0";

# Points => Discipline Event;
# These numbers of points trigger a discipline event.
%trigger = (
6 => 'Absence (Two)',
12 => 'Absence (Four)',
18 => 'Absence (Six)',
24 => 'Absence (Eight)',
30 => 'Absence (Ten)'
);

# A formletter file (in /cgi/forms) that matches trigger events above
# Generated by the Attendance Scan script (/cgi/attscan.pl)
%formletter = (
6 => 'discipline.tex',
12 => 'discipline.tex',
18 => 'discipline.tex',
24 => 'discipline.tex',
30 => 'withdraw.tex'
);

# Ethnic values now stored in meta table as defaults.
# Religion Values now stored in meta table as defaults. - $g_religion

# Staff Member Positions; all globals vars start with g_

# Allows admin functions in Discipline system.
@g_posadmin = ('Principal', 'Vice-Principal');

# New functions in upcoming PTI system
$g_posped = 'Special Education Teacher';
```

```
$g_position[0] = "Classroom Teacher";
$g_position[1] = "Teacher Assistant";
$g_position[2] = "Special Education Teacher";
$g_position[3] = "Principal";
$g_position[4] = "Vice-Principal";
$g_position[5] = "Secretary";
$g_position[6] = "Librarian";
$g_position[7] = "Community School Room";
$g_position[8] = "Custodian";

# Needed by perl scripts to loading this config file.
1;
```

Other Configuration Files in the etc directory:

1. `repcard.conf` - a configuration file to control report card printing. Please read this file over carefully so that your report card prints correctly. This has lots of comments in it to explain the different features, which include: column spacing, show averages, show homeroom teachers, show school logo, number of terms to show, term descriptors, per grade evaluation keys, student placement text for year end report cards. Still missing: show per subject attendance.
2. `image.conf` - a configuration file to control the student/staff images. It mainly contains limits on filesize uploads and resolution of created student images.
3. `schedule.conf` - a scheduling system configuration file. Currently it only contains the period start and end times during the day
4. `transcript.conf` - a configuration file for the transcript system.
5. `gbook.conf` - gradebook configuration file.

The special education (IEP (Individual Education Program)) application has a similar script with common settings.

There is also a global configuration file (nominally located in the school etc folder and called `global.conf`) that stores information about all of the schools on the local server (normally, that would be all participating schools in the division). This information is used to enable global reporting. This is used by central office for reporting as well as used by the schools to enable school transfers of demographic information. Once a secretary or admin in a school finds the student number of a withdrawn student, they can simply put that number into the normal *entry/withdraw* input box on the main page to register that student and have all demographic information transferred.

I've located the global config file in the school etc folder, and set the `admin.conf` configuration file to point to it's location there. This is because most installations are for single schools, not large division installs.

If you are doing a multiple school installation then move this file to a *global* folder at top level (ie. /opt/openadmin/global). Add all databases to it. Then edit all school admin.conf files so that their \$globaldir setting points to this folder.

The *Look and Feel* of all school sites can be changed by simply changing the stylesheet files (.css) located in *admin*, *tadmin*, and *padmin* folders. Normally I just change the color scheme slightly to match colors in the school logos (ie. school colors). I do this by searching and replacing the #049 (RGB) with a color triplet matching the logo.

The scripts that generate webpage reports, etc. also use the same stylesheet so that this one file can control the overall look of the application. Currently, this has only been done to a very limited degree to make sure browser compatibility doesn't become an issue. As browser support continues, this will evolve.

The HTML pages are basically *separate* from the scripting, so that the look of all school sites can be changed extensively if desired by simply changing the html files and the CSS files. The only issue will be to maintain those elements in the CSS files used by the scripts to control report rendering, etc.

As well, within the stylesheets are settings to it hide the religion fields in the student table and also the high school functions in the SDS update section of the export page.

Step 7: Getting Started on the Browser Side...

From this point on, secretaries may take over, if desired. The following functions are on the Start/End of Year page (There is more about this in the User Documentation under *Getting Started*).

1. **Add your teachers and other staff members.** For a teacher in a split situation (teaching 2 grades in the same classroom), add two copies of the same teacher differing only in the grade number, but with the same homeroom value. For jobsharing teachers, add each one in, with the *most important* one entered first. (You get to decide who that is...)

In a middle years or high school (or with any subject specialists who only teach certain subjects such as music or foreign languages), the homeroom and grade fields are normally left blank. The teacher is grouped with students by their subject enrollment in his/her class.

2. **Enter school dates** that school is not in session, Monday to Friday. This is located on the Start/End of Year page also.

This includes any school holidays within the year and teacher inservices, etc. (The system automatically assumes (using the Date::Business module) that school is not in session on Saturday/Sunday). This should now correctly do attendance calculations, etc. The DayInCycle setting is used to mark those days that actually *count* in schools using different types of cycles (ie. 6 day cycles) for classes, even though the students may not be in class (ie. due to teacher professional development).

You are now ready to enter attendance, discipline, etc.

Install Appendix A - Setting up the PostgreSQL Database

(Courtesy of David Bandel)

Note (July, 2007): This section hasn't been updated in a while, so some options may have changed, etc. You're on your own with this since I have no test versions running pgsq any more.

Ensure you have installed the PostgreSQL server and associated tools (including plpgsql). Most distributions include PostgreSQL, but if not you can probably find binaries at the PostgreSQL site (<http://www.postgresql.org/>).

Make sure Postgres is running and starts on bootup. If running, you should find a postmaster daemon (`ps ax — grep postmaster`). If the postmaster process is not running, start it. If this is a new install, you may get an error. If so, you probably haven't initialized the database properly. See the Postgres instructions for running `initdb`.

These instructions assume you are running PostgreSQL as the user `postgres`, and that user is also the db administrator. If you don't understand the preceding sentence, please read the PostgreSQL documentation that came with your binaries or can be found on the PostgreSQL site.

1. First, if you haven't installed the `plpgsql` language, do so:

```
createlang -U postgres plpgsql template1
```

2. Then, if you want better security, create a user for `openadmin` with a password:

```
createuser -U postgres -P openadmin
```

Allow the user to create new databases, but decline creating new users. The `-U` allows you to specify the user, `-P` prompts for a password. If you don't want your user to have a password (bad idea), omit the `-P`. It's also a good idea to add a password to your `postgres` admin user.

Note: if you are refused a connection to your database in 1 or 2 above, check your `pg_hba.conf` file and change the lines:

```
local all postgres ident sameuser
local all all ident sameuser
host all all 127.0.0.1 255.255.255.255 ident sameuser
```

to:

```
local all postgres trust
local all all trust
host all all 127.0.0.1 255.255.255.255 trust
```

and restart your PostgreSQL server.

3. Create the database (let's assume you chose `openadmin` as your username above):

```
createdb -U openadmin myschool
```

-
4. Now create the database tables:

```
psql -U openadmin myschool ; blank.pg.sql
```

You will find the PostgreSQL schema in the utilities directory under the name blank.pg.sql.

5. Configure each school to use PostgreSQL. In `etc/admin.conf` under each school directory, ensure your `dbtype` line looks like this:

```
$dbtype = 'Pg';
```

The P is capitalized, the Pg is surrounded by single quotes, and the assignment ends with a semicolon. This paragraph is a change to the instructions as presented in Configuration, below.

6. Ensure you run a backup script to backup your databases regularly. A Pg backup script is available for download.

Modify your crontab thusly for a full nightly backup at 2 a.m.:

```
0 2 * * * /root/pg_backup.sh bva
```

Remember to configure your backup script appropriately.

Install Appendix B - Setting Up Webname Resolution

You can set these website locations in individual computers in their `hosts` files so that they can access these sites without any DNS (Domain Name) records being installed on your DNS server (that maps webnames to IP addresses). This can be quite useful in initial testing before any DNS updates have been done. On Linux workstations these are located in `/etc/hosts`. On Windows workstations (XP) these are found in:

```
c:\windows\system32\drivers\etc
```

Look on the wikipedia website (wikipedia.org) under *Hosts file*.

The entries in the hosts files look like this for St. Peter:

```
spa.mysd.ca 142.165.5.25 # admin site  
spt.mysd.ca 142.165.5.25 # teacher site  
spp.mysd.ca 142.165.5.25 # parent site
```

Using hosts files will help *hide* the virtual sites to some degree, since there will not be DNS records for them. However, they have to individually maintained. Also, in terms of security, avoid placing links to admin sites on school web pages. (security by obscurity). The next step up in security is the use of SSL to give encrypted connections to/from websites.

The DNS System

The webserver is setup for several *virtual* OA websites. You must setup your DNS records (on your DNS server) to match. These DNS names are set to point to the **same** server.

Let's assume that you are setting up the webserver and DNS for the Tiny School Division and they have a registered domain name for *mysd.ca* and a server located at 142.165.5.25 called *gandalf*.

You would setup a DNS record to point to the server:

```
gandalf IN A 142.165.5.25
```

and also several *aliases* that also point to the same server:

```
spa IN CNAME  gandalf  - St Peter School Office Administration
spt IN CNAME  gandalf  - St Peter School Teacher Admin
spp IN CNAME  gandalf  - St Peter School Parent Site
```

```
bruna IN  CNAME  gandalf  - Bruno School Office Administration
brunt IN  CNAME  gandalf  - Bruno School Teacher Admin
brunp IN  CNAME  gandalf  - Bruno School Parent Site
```

```
iep      IN  CNAME  gandalf  - Special Education Student Site
central  IN  CNAME  gandalf  - Central Office Site
```

Even though you are going to the same computer, the virtual hosting capabilities of the Apache web server will show you different websites (based on the name that you're looking for which is sent in the HTTP header).

The St Peter secretaries and administrators go to *spa.mysd.ca* to enter attendance, etc. while teachers at St Peter would go to *spt.mysd.ca* to print their classlists, etc. Parents could go to *spp.mysd.ca* to see their children's attendance, reportcards and ongoing grades from the gradebook (where allowed). This would be similar for other schools running on this same server.

Central Office staff and special education teachers would go to *iep.mysd.ca* to view special needs student programs. There is also a single central office application to show combined school enrollments, ethnic enrollments and other functions.

All these addresses are going to the **same** server. This is the basic idea behind this administration system. The entire school division can run from one medium performance server that can be housed and maintained at some central location. The parents site is not protected by a common password (since the scripts require individual passwords from parents), while the other two (per school) are protected by passwords stored in a file (or database)for the main admin site and in the staff table for the teacher site.

How to Upgrade Open Admin for Schools

Versions

Installation for OA 3.25/50 - August/December 2008

There have not been any large changes in this release. (This is an abbreviated version of the 2.50 instructions, below) The usual process is, as before, :

1. Move the old version out of the way and insert the new version.

- Move/Rename the old version to, for example, *stpeterold*.
- Create a new directory of the same name and copy the new folders of OA 3.50 **school** into this *stpeter* folder.
- Change the ownership of the new *stpeter* folder so that it matches that of the UID/GID that the web server is running as.

2. Copy the configuration files from the old etc folder into the new etc folder. Copy the *admin.conf*. Copy the *studentnumber* file (which contains the next available student number), the *term* file (which contains the current term) and the *receiptnumber* file (which contains the next receipt number for the fees system). Copy the *admin.conf* (main configuration file) and *repcard.conf* file (report card system) (and any other .conf files that have been changed in previous install. (Look at file dates)

3. Copy the previous 2 school logo files (logo.gif and logotn.gif) in admin/images into their new location. Do the same for the admin.css file which contains the color information (in /admin).

4. Edit the (old) admin.conf and the admin.conf.350. Copy the changes at the top of the new 3.50 file and make any other necessary changes in file locations, etc. Do the same with the repcard.conf and the new repcard.conf.350 and update changes at the top of the file.

5. Backup your data using your backup tool of choice. Mysqldump works well too.

6. Update the database table structure using the provided update scripts (in the utility/sql/325to350 folder of the download). The sql files add the new PK report card (altrc file) and the new translation tables (xlat file).

Now update the meta data by emptying the meta table (in console do `delete from meta`). Then add in new meta data from the metadata.sql file (use either CDN or US version). You could also just use the metadata update script on the end of year page to do the same thing.

You will also have to update the studentall table if using a version 4.x mysql installation. Drop this table `drop table studentall` in console and then use the provided studentall file in utility/sql to add in the new one. Use the mysql4 version.

7. If you would like to use the *iep* (special education) application, please read the installation notes. It is a separate website with it's own html and cgi folders. It would normally be installed into /opt/openadmin/iep.

-
8. **TaDa.** Sit down and gloat. Gloating is **Important**. Tell your Spouse or Significant Other how wonderfully talented you are, etc.

Installation for OA 2.50 - December, 2007

This version includes update to the transcript system (multiple page single student transcripts), report card system, fees system (new report, other fixes). There are also a large number of script rewrites to support alternate languages.

Updating – OA 2.50

When updating, do the following:

1. **Move the old version out of the way and insert the new version.**

Assume the new version is in `/tmp/openadmin-2.50/school`. (as a result of untarring in the `/tmp` folder: `tar xvzf openadmin-2.50.tgz`)

I would suggest that you use the excellent file management tool, Midnight Commander, **mc**, (a clone of Norton Commander from the *olden days*) to carry out the following directions (although you could use the command line also). These instructions assume that your installation is located in `/opt/openadmin` and that your present working directory is `/opt/openadmin` also.

- Move/Rename the old version (`mv stpeter stpeter.old`). You can then use any files from there if you forgot they were customized for your school. If you already have the directory `stpeter.old` from a previous installation, either delete it or move it into a backup area.
- Create a new directory of the same name (`mkdir myschool`) and copy the new folders of OA 2.50 into this folder. (`cp -r /tmp/openadmin-2.50/school/* stpeter`)
- Change the ownership of the new `stpeter` folder so that it matches that of the UID/GID that the web server is running as. This is necessary so that the webserver user id (which all users run as when talking via the server) have permissions to create temporary tex files for pdf creation and can also rewrite the studentnumber and term files in etc. If they can't you won't be able to create pdf reports or have new student numbers show up (ie. enrol students).

`chown -R webuser:webgroup stpeter` where *webuser* and *webgroup* are the UID and GID respectively that the webserver is running as. This is *www-data* in debian for both user and group.

You now have your old installation in `stpeter.old` and the new release taking it's place in `/opt/openadmin/stpeter`. The next steps will be the replacement and editing of the configuration files and customized features (colors, logo).

2. Copy 3 configuration files from the old etc folder. Copy the studentnumber file (which contains the next available student number):
(`cp stpeter.old/etc/studentnumber stpeter/etc/`).

Do the same for the *term* file (which contains the current term) and the *receiptnumber* file (which contains the next receipt number for the fees system). They will replace the newer ones in the *stpeter/etc* folder.

3. Copy the previous 2 school logo files (*logo.gif* and *logotn.gif*). (`cp stpeter.old/admin/images stpeter/admin/images`). Do the same with the thumbnail file (*logotn.gif*).
4. Update the color scheme in your current *admin.css* by searching for #049 and replace it with the color code of your choice. Normally the color code is chosen to match the school logo colors. This is a simple way to customize each of your schools.

The *cgi*, *tcgi*, *tadmin* have been updated and scripts changed so that there should be no need to change anything in any of these folders when updating. (unless you have a customized version of a script). For this, I would suggest making a folder called *custom* in the folder where this script is located, and then create a symbolic link to where the customized version is (in the *custom* folder). This will make it easier to locate those changed scripts.

All configuration data will continue to be placed in the */etc* directory for each school. This greatly simplifies updating the software since there is only one location for changing configuration.

5. Edit the new versions of your *admin.conf* (and other configuration files) to put back in the features you want. There are several approaches to this. One way is to open both the new and the old file and copy and paste larger sections from one to another. Another approach is to use *diff* to locate differences between the old version and the new version (which is why I provide the older original versions). Then just copy your old version into place and add in the new sections (if any) and remove old sections.

The differences between the *admin.conf* for 2.50 are now all placed at the top of the file, so that they are easily visible.

6. **Backup your data** using your backup tool of choice. *Mysqldump* works well too.
7. Update the database table structure using the provided update scripts (in the utility folder of the download) in */tmp/openadmin-2.40/utility/240to250*. The *sql* files update both the fees journal and add additional student table field.

If you have an earlier version of OA, you will have to follow the database updates through each version to get the data to the current version. Newer releases of OA have folders containing several *sql* files rather than just one large *sql* file to do the update. However, the method is the same regardless.

You run the update script using a command like:

```
mysql -ppassword database < fixfees_jrl.sql
```

where *password* is your database password and *database* is your school database and *fees.sql* is the *sql* update file.

8. You can now run the *metaupdate.pl* script located in the *cgi/meta* directory. It is normally run from the Start/End of Year web page of the admin site, but you can run it directly also. This will update the meta table with the new fields that have been added. The *metaedit* script can also be run once you have OA running to make changes to defaults, etc. This is located on the Eoy page.

-
9. If you would like to use the *iep* (special education) application, please read the installation notes. It is a separate website with it's own html and cgi folders. It would normally be installed into `/opt/openadmin/iep`.
 10. **TaDa**. Sit down and gloat. Gloating is **Important**. Tell your Spouse or Significant Other how wonderfully talented you are, etc.

Updating – OA 2.40

When updating, do the following:

1. **Move the old version out of the way and insert the new version.**

Assume the new version is in `/tmp/openadmin-2.40/school`. (as a result of untarring in the `/tmp` folder: `tar xvzf openadmin-2.40.tgz`)

I would suggest that you use the excellent file management tool, Midnight Commander, **mc**, (a clone of Norton Commander from the *olden days*) to carry out the following directions (although you could use the command line also). These instructions assume that your installation is located in `/opt/openadmin` and that your present working directory is `/opt/openadmin` also.

- Move/Rename the old version (`mv stpeter stpeter.old`). You can then use any files from there if you forgot they were customized for your school. If you already have the directory `stpeter.old` from a previous installation, either delete it or move it into a backup area.
- Create a new directory of the same name (`mkdir myschool`) and copy the new folders of OA 2.40 into this folder. (`cp -r /tmp/openadmin-2.40/school/* stpeter`)
- Change the ownership of the new *stpeter* folder so that it matches that of the UID/GID that the web server is running as. This is necessary so that the webserver user id (which all users run as when talking via the server) have permissions to create temporary tex files for pdf creation and can also rewrite the studentnumber and term files in etc. If they can't you won't be able to create pdf reports or have new student numbers show up (ie. enrol students).

`chown -R webuser:webgroup stpeter` where *webuser* and *webgroup* are the UID and GID respectively that the webserver is running as. This is *www-data* in debian for both user and group.

You now have your old installation in `stpeter.old` and the new release taking it's place in `/opt/openadmin/stpeter`. The next steps will be the replacement and editing of the configuration files and customized features (colors, logo).

2. Copy 2 configuration files from the old etc folder. Copy the studentnumber file (which contains the next available student number):

(`cp stpeter.old/etc/studentnumber stpeter/etc/`).

Do the same for the *term* file (which contains the current term). They will replace the newer ones in the `stpeter/etc` folder.

3. Copy the previous 2 school logo files (logo.gif and logotn.gif). (`cp stpeter.old/admin/images stpeter/admin/images`). Do the same with the thumbnail file (logotn.gif).

4. Update the color scheme in your current admin.css by searching for #049 and replace it with the color code of your choice. Normally the color code is chosen to match the school logo colors. This is a simple way to customize each of your schools.

The cgi, tcgi, tadmin have been updated and scripts changed so that there should be no need to change anything in any of these folders when updating. (unless you have a customized version of a script). For this, I would suggest making a folder called *custom* in the folder where this script is located, and then create a symbolic link to where the customized version is (in the custom folder). This will make it easier to locate those changed scripts.

All configuration data will continue to be placed in the /etc directory for each school. This greatly simplifies updating the software since there is only one location for changing configuration.

5. Edit the new versions of your admin.conf (and other configuration files) to put back in the features you want. There are several approaches to this. One way is to open both the new and the old file and copy and paste larger sections from one to another. Another approach is to use diff to locate differences between the old version and the new version (which is why I provide the older original versions). Then just copy your old version into place and add in the new sections (if any) and remove old sections. There are no differences between the admin.conf versions for 2.30 and 2.40, so the old version may be copied over directly.

6. **Backup your data** using your backup tool of choice. Mysqldump works well too.

7. Update the database table structure using the provided update scripts (in the utility folder of the download) in /tmp/openadmin-2.40/utility/2.30to2.40. The fees.sql file is the update.

If you have an earlier version of OA, you will have to follow the database updates through each version to get the data to the current version. Newer releases of OA have folders containing several sql files rather than just one large sql file to do the update. However, the method is the same regardless.

You run the update script using a command like:

```
mysql -ppassword database < fees.sql
```

where *password* is your database password and *database* is your school database and *fees.sql* is the sql update file.

8. You can now run the metaupdate.pl script located in the cgi/meta directory. It is normally run from the Start/End of Year web page of the admin site, but you can run it directly also. This will update the meta table with the new fields that have been added. The metaedit script can also be run once you have OA running to make changes to defaults, etc. This is located on the Eoy page.

-
9. If you would like to use the *iep* (special education) application, please read the installation notes. It is a separate website with it's own html and cgi folders. It would normally be installed into /opt/openadmin/iep.
 10. **TaDa.** Sit down and gloat. Gloating is **Important.** Tell your Spouse or Significant Other how wonderfully talented you are, etc.

Model OpenAdmin Installation - Single Site with SSL

This section written by *Andy Figueroa*

1. Operating System Installation

Download Ubuntu 7.10 server edition starting here:

<http://www.ubuntu.com/getubuntu/download>

Install the server, choosing the LAMP (Linux, Apache, MySQL, PHP) option when prompted, and also the openssh-server option (new in version 7.10) if you want to access your server remotely using ssh.

During the installation of Ubuntu server edition, you will be asked a few questions:

- You will be asked to set up a user: use a name and password you can remember and write it down.
- You will be asked to set up a MySQL `root` user password. Write it down.

When the installation is complete, boot into your new server.

Unless you want to work exclusively using `sudo`, set up a root password using:

`sudo passwd root` and you will be prompted to enter your password then you will be prompted to enter the root password twice. Write it down.

2. (a) Base System Configuration

Do all of the following logged in as root, using `su` to become root, or prefixing each command with `sudo`

Change the default DHCP network setting to a fixed IP address if appropriate for your installation (typically appropriate) by editing the file `/etc/network/interfaces` using IPs appropriate for your network. You may use any text editor. Text editors `nano` and `vi` are installed. A model file for static IP address follows:

```
#file /etc/network/interfaces

# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
auto eth0
#iface eth0 inet dhcp

iface eth0 inet static
    address 192.168.1.50
    netmask 255.255.255.0
    gateway 192.168.1.1
```

To avoid errors from `apache2` be sure to add your hostname and IP address to the `/etc/hosts`

```
# file /etc/hosts
127.0.0.1      localhost
127.0.1.1      oaserver2.myschool.dnsalias.net    oaserver2
192.168.1.50   oaserver2.myschool.dnsalias.net    oaserver2
```

Update sources using the command `apt-get update`

Optional: If you didn't install the `openssh-server` as part of the base server installation (a new option with Ubuntu-server-edition 7.10), and you want to access your server using `ssh`, install the `openssh-server` using:

```
apt-get install openssh-server
```

Use `apt-get` to install the following additional packages and their automatically installed dependencies: `apg`, `make`, `gcc`, `zip`, `unzip`, `openssl`, `lynx`, `ncftp`, `tetex-base`, `tetex-bin`, and `tetex-extra` using:

```
apt-get install apg
apt-get install gcc
apt-get install lynx
apt-get install make
apt-get install ncftp
apt-get install openssl
apt-get install tetex-base
apt-get install tetex-bin
apt-get install tetex-extra
apt-get install zip
apt-get install unzip
```

Note – the following is **not** required for OpenAdmin 2.50 and higher Get (various web sources) `isolatin1.sty` and copy it into `/usr/share/texmf-texlive/tex/latex/base`. Edit the file `/usr/share/texmf-texlive/ls-R` adding `isolatin1.sty` to the `./tex/latex/base:` section (Note: for Ubuntu 7.04 and possibly lower change the above paths from `texmf-texlive` to `texmf-tetex`)

(b) Installing the necessary Perl modules

You may get more information about Perl at <http://www.perl.org/>

Most of the perl modules needed by OpenAdmin can be installed from the Debian repositories using `apt-get install modulename` as follows (the portion in parenthesis shows perl module name):

```
apt-get install libxml-libxml-perl (LibXML2 and XML::LibXML)
apt-get install libcgi-perl (CGI)
apt-get install libcgi-simple-perl (CGI - may not strictly be
    required)
apt-get install libdbd-mysql-perl (DBI and DBD for MySQL -
    installed by default)
apt-get install libtext-csv-perl (Text::CSV_XS)
apt-get install libfreezethaw-perl (FreezeThaw)
apt-get install libcgi-session-perl (CGI::Session)
apt-get install libwww-perl (Bundle::LWP - installed by default)
apt-get install libcrypt-ssleay-perl (Crypt::SSLeay)
```

```
apt-get install libio-socket-ssl-perl (may not strictly be
    required)
apt-get install libxml-writer-perl (XML::Writer)
apt-get install libdata-uuid-perl (Data::UUID)
```

The following perl modules were not identified in the Debian repositories and may be retrieved from CPAN. The first time you invoke CPAN you will be walked through a configuration dialog. The defaults suggested are correct and work fine. Enter appropriate responses when prompted for your geographic location and the servers you want to use.

```
perl -MCPAN -e 'install Date::Business'
perl -MCPAN -e 'install Time::JulianDay'
perl -MCPAN -e 'install XML::Writer::String'
```

You now have a working Apache2, MySQL server on port 80 (normal http port) that meets all the program requirements of OpenAdmin 2.50. You may test it from another computer on your network by visiting `http://192.168.1.50/` with a web browser (substitute your own IP address if different). You may also test it locally using `lynx` from the shell if you don't have another computer on your network handy:

```
lynx http://192.168.1.50/
```

3. Configuring Apache2 for OpenAdmin using SSL

The following changes are all applied in `/etc/apache2`: `cd /etc/apache2`

Create symbolic links to ssl modules in `/etc/apache2/mods-enabled`:

```
cd mods-enabled
ln -s /etc/apache2/mods-available/ssl.load ssl.load
ln -s /etc/apache2/mods-available/ssl.conf ssl.conf
```

Use your favorite editor to edit `dir.conf` adding `index.shtml` to the list of automatically served files:

```
<IfModule mod\_dir.c>
# Keep the two lines below joined as one line in dir.conf.
    DirectoryIndex index.html index.shtml index.cgi index.pl
index.php index.xhtml
</IfModule>
```

Return to `/etc/apache2`: `cd ..`

Create a site to use ssl by copying the default site/file to the ssl site/file:

```
cd sites-available
cp default ssl
```

Edit ssl using your favorite editor to modify existing lines or adding, as needed, to read:

```
NameVirtualHost *:443
<VirtualHost *:443>
    ServerAdmin webmaster@localhost

    SSLEngine on
    SSLCertificateFile /etc/apache2/ssl/apache.pem

    DocumentRoot /var/www/
    <Directory />
# Options FollowSymLinks
        Options FollowSymLinks Includes
# AllowOverride None
        AllowOverride All
    </Directory>

    <Directory /var/www/>
        Options Indexes FollowSymLinks MultiViews
# AllowOverride None
        AllowOverride All
        Order allow,deny
        allow from all
        # This directive allows us to have apache2's default
# start page in /apache2-default/, but still have /
# go to the right place
        RedirectMatch ^/$ /apache2-default/
    </Directory>

<IfModule alias_module>
## Note: This entire section to the end is required to allow
## openadmin to run perl scripts from the right path operating
## in a single SSL protected site. It is not necessary for the
## purpose of getting the SSL site running. The portion that is
## comment out below is from the original file and must remain so.
#     ScriptAlias /cgi-bin/ /usr/lib/cgi-bin/
#     <Directory "/usr/lib/cgi-bin">
#         AllowOverride None
#         Options +ExecCGI -MultiViews +SymLinksIfOwnerMatch
#         Order allow,deny
#         Allow from all
#     </Directory>
#
# ScriptAlias: This controls which directories contain server
# scripts. ScriptAliases are essentially the same as Aliases,
# except that documents in the realname directory are treated
# as applications and run by the server when requested rather
```

```
# than as documents sent to the client. The same rules about
# trailing "/" apply to ScriptAlias directives as to Alias.
#
#ScriptAlias /cgi-bin/ /var/www/localhost/cgi-bin/

ScriptAlias /cgi-bin/ /var/www/openadmin/myschool/cgi/
ScriptAlias /tcgi-bin/ /var/www/openadmin/myschool/tcgi/
ScriptAlias /pcgi-bin/ /var/www/openadmin/myschool/pcgi/
</IfModule>

<Directory "/var/www/openadmin/myschool/cgi/">
AllowOverride All
Options FollowSymLinks Includes
Order allow,deny
Allow from all
</Directory>

<Directory "/var/www/openadmin/myschool/tcgi/">
AllowOverride All
Options FollowSymLinks Includes
Order allow,deny
Allow from all
</Directory>

<Directory "/var/www/openadmin/myschool/pcgi/">
AllowOverride All
Options FollowSymLinks Includes
Order allow,deny
Allow from all
</Directory>
```

No further changes from `ErrorLog /var/log/apache2/error.log` to the end.

Now, create a symbolic link `/etc/apache2/sites-enabled/000-ssl` to `/etc/apache2/sites-available/ssl`:

```
cd ..
cd sites-enabled
ln -s /etc/apache2/sites-available/ssl 000-ssl
cd ..
```

Generate a self-signed SSL certificate using the following commands:

```
mkdir /etc/apache2/ssl
```

The following comand must all on **one** commandline:

```
openssl req $a -new -x509 -days 3650 -nodes -out /etc/apache2/ssl/
    apache.pem -keyout /etc/apache2/ssl/apache.pem
```

Use the same fully qualified domain name when prompted for Common Name that you used in `/etc/hosts` (i.e. `oaserver2.myschool.dnsalias.net`) to keep `apache2` from giving an error about the domain name.

or you can go to <http://launchpadlibrarian.net/7477840/apache2-ssl.tar.gz> and grab the old `apache2-ssl-certificate` which is an easier way to generate a good self-signed SSL certificate.

Extract the package and put `ssleay.cnf` in `/usr/share/apache2/`
put `apache2-ssl-certificate` in `/usr/sbin`.

Create `/etc/apache2/ssl` with `mkdir /etc/apache2/ssl`

Then the `apache2-ssl-certificate` script should run from the commandline.

Finally, restart `apache2` with the following command:

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

Test your work from another computer on your network with:

```
https://192.168.1.50/
```

If it does not work, check the error message on the screen and the logs in `/var/log/apache2`.

Assuming that you do not also want an unsecured `http` connection to port 80, delete or comment out the `Listen 80` line in `/etc/apache2/ports`

Also delete the symbolic link `000-default` in `/etc/apache2/sites-enabled` with:

```
rm /etc/apache2/sites-enabled/000-default
```

Next - setup your school's MySQL database...

4. Setting Up the MySQL database

Create a database for your school where the database name is `myschool` and the root MySQL password in the example is (not the same as the system root password) `oa99` which you established when installing the server:

```
mysqladmin create -poa99 myschool
```

Now create a non-root password for this school by using the `mysql grant` command by first entering the MySQL client:

```
mysql -poa99 -u root myschool
```

Then type (on one line):

```
grant all on myschool.* to myuser@localhost identified by  
'mypassword';
```

like the following (on one line, of course):

```
grant all on myschool.* to oaadmin@localhost identified  
by 'oaadmin99';
```

where `myschool` is the name of the database, and `myuser` (`oaadmin` in the example) and `mypassword` (`oaadmin99` in the example) are the user and password to get access to the program. They do not have to exist on the system as a real user, but are merely used by the scripts. These are the `$user` and `$password` variables used in the `etc/admin.conf` file we will set up later.

And then:

A new mysql identity called `rouser` (read only user) should be created that has only *read* (sql select) capability for that school's database. This is particularly important for use with the publicly accessible parent page/site that allows students and parents to view attendance, report and gradebook marks. Other scripts may use this as well as development of OpenAdmin continues.

```
grant select on myschool.* to rouser@localhost identified
by 'ropassword';
```

as in the example below (on one line):

```
grant select on myschool.* to oouser@localhost identified
by 'oouser99';
```

where `rouser` (`oouser` in the example) and `ropassword` (`oouser99` in the example) are the values also entered in the `etc/admin.conf` configuration file for `$rouser` and `$ropassword`.

Also enter a global user with read-only capability to be used in `global.conf` as shown:

```
grant select on myschool.* to global@localhost identified
by 'global99';
```

Exit MySQL by entering: `exit;` or `quit;`

Be sure to write down all the names and passwords you just created. You will need them again to configure OpenAdmin.

Now that you have a database name and users, you are now ready to set up OpenAdmin.

5. Downloading, Installing, and Setting up OpenAdmin

The following is a simplified version of what is found in the official OpenAdmin documentation for setting up a single school on a single server. For more complex arrangements, please see the official OpenAdmin documentation. It is recommended that you at least read through the official OpenAdmin documentation in order to better understand how OpenAdmin works.

Download the latest Open Admin version from <http://richtech.ca/openadmin> and then untar it into `/opt` (We will create symlinks from `/var/www` to the installation path a little later).

```
cd /opt
tar xzf /path/to/download/openadmin-2.50.tgz
```

To make it easier to work with the new directory, create a symlink to the resulting `openadmin` directory:

```
ln -s openadmin-2.50 openadmin
```

Change the owner and group of the OpenAdmin directory and everything below it to the `apache2` user:

```
chown -R www-data:www-data openadmin
```

Enter the OpenAdmin directory and copy the base school directory to a school directory for your own school:

```
cd openadmin
```

```
cp -a school myschool
```

The new myschool directory is where OpenAdmin will run and where we will do almost all of our configuration. You may substitute any short single word for myschool if you would like to better distinguish your school identification, i.e da, stpeter, stmary, hhcs and so on.

Next, we'll create symbolic links from the apache2 root httpd directory, /var/www/, to our installed OpenAdmin in /opt/openadmin where openadmin is an existing symbolic link to the current installation:

```
cd /var/www
```

```
ln -s /opt/openadmin openadmin
```

```
ln -s /opt/openadmin/myschool/admin oaadmin
```

```
ln -s oaadmin/js js
```

```
ln -s oaadmin/images/favicon.ico favicon.ico
```

```
ln -s /opt/openadmin/myschool/tadmin oateacher
```

```
ln -s /opt/openadmin/myschool/padmin oastudent
```

Change to the /opt/openadmin/myschool/etc directory and edit the admin.conf file:

```
cd /opt/openadmin/myschool/etc
```

Use your favorite editor to configure admin.conf

```
nano admin.conf ( or vi admin.conf )
```

The first 3 entries of the admin.conf configuration file following the added section at the top for release 2.50 are for user, password, and database,. These were established when we set up the database section above.

Although many of the entries/changes that are needed are specific to a particular school, others are common to single site SSL operation. A model is included at the end of this guide. All of the entries are important, but most important are those providing paths to OpenAdmin's files.

When finished with admin.conf, change to the myschool/tadmin directory and edit the index.shtml file, changing all the references to cgi-bin to tcgi-bin.

```
cd /opt/openadmin/myschool/tadmin
```

Such a global search and replace is easily done in vi as follows:

```
vi index.shtml
```

```
:%s/cgi-bin/tcgi-bin/g
```

```
ZZ
```

Also edit and configure the global.conf file in /opt/openadmin/global

```
cd /opt/openadmin/global
```

```
nano global.conf or vi global.conf
```

6. Create the database tables

Change directories to `/opt/openadmin/utility` and enter the following to setup the database tables:

```
mysql -pooadmin99 -u oaadmin myschool <blank.sql.250.mysql5x
```

Also, do the same with `meta.sql` to create the starting student and staff table values:

```
mysql -pooadmin99 -u oaadmin myschool <meta.sql
```

If you don't get any errors, you may be done. You can check by looking in `/var/lib/mysql/myschool` to see if it is populated with a screen full of tables.

7. How to get started

If everything is working, you are now ready to enter dates, teachers, courses, schedule, and students, in that order. Then enroll students in classes. See the administration manual section 6, and the user manual in the OpenAdmin documentation for details.

8. Files available

The following files are all included in the archive **oasslfiles.tgz** found in the utility directory. Current versions are also maintained at <http://cathedralsoul.us/oafiles/>. Use the command `tar xzpf oasslfiles.tgz` and they will be extracted into your current path directory `oasslfiles` (i.e. `~/oasslfiles`), which it will create if it doesn't exist. To use, edit the files as may be necessary for your own situation and copy them into their respective directories.

```
/etc/network/interfaces  
/etc/hosts  
/etc/apache2/mods-available/dir.conf  
/etc/apache2/sites-available/ssl  
/etc/apache2/ports.conf  
/opt/openadmin/myschool/etc/admin.conf  
/opt/openadmin/myschool/tadmin/index.shtml
```

script: `symlink.scr`

Creates the necessary symlinks in `/etc/apache2`, `/var/www/`, `/opt` and creates the directory `/etc/apache2/ssl`.

(Note: Look at this script before you run it. Edit it if necessary for your particular installation. You will have to delete and recreate the symlink at `/opt/openadmin` if your installed version of OpenAdmin changes from `openadmin-2.50`. The other symlinks should never have to change except to suit your own situation.)

Optional: `apache2-ssl.tgz`

A script included in earlier versions of Apache for easily generating a self-signed certificate. The files from the archive `ssleay.cnf` is to be copied into `/usr/share/apache2/` and `apache2-ssl-certificate` into `/usr/sbin` Execute the script by running `apache2-ssl-certificate` from the shell as described in the Apache2 configuration section, Section 3 above.

Customize OA - Adding New Fields to Students and Staff

Overview

Beginning with the early 2.00 versions, Open Admin can have both student demographics and staff information extended with additional fields with very little effort. The staff table already contains several additional fields for use by international schools for storing staff certification, etc.

The only negative to this process is that a report must be created to actually display this information if the reports are not based on customizable templates. Later versions of OA will fix this omission.

This is made possible by:

1. **Templates** – files located in a *template* directory hold layouts for student addition and editing, student search, and student viewing. These templates are loaded by scripts to enable editing of the student table(s): student, studentwd(withdrawn students) and prereg (preregistered students).
2. A **Metadata** table – a table storing information about the structure of the staff and student tables in OA. There are additional fields in each record that are used to control form element length and type, default values, etc. This information can be modified by using the meta scripts (edit and update) on the Eoy page.

Extending Student Demographics

As a result of this, additional fields may be added to the end of the student tables. All 3 student tables must remain identical since records are moved directly from one table to another during withdrawal, re-enrollment, etc. Fields should not be removed from the current student table format since older scripts may depend on this organization.

The templates are then updated to add these additional field(s) to the templates. The final part of the process would include, possibly, some additional custom reports that support these new fields. OA really requires a report generator, but that must lie in the future.

The following scripts are now *template aware* and will display the additional fields added to the student tables and templates:

1. **Student Search** - cgi/studsearch.pl (uses the studsearch.tpl template).
2. **Student Enrollment** - cgi/entry/sentryX.pl scripts (use the student.tpl template).
3. **Student Editing** - cgi/studed.pl (and studeled.pl now supports editing of any of the 3 student tables)(also uses the student.tpl template)
4. **Student Viewing** - cgi/studview.pl (using studentview.tpl)

The templating *technology* is currently making use of simple regex matching and replacement. Future use of templating modules is being considered as other needs develop and the pros and cons of each module are weighed.

The templates themselves are simply HTML markup blocks with the following tags that are replaced during script execution:

1. `<*desc*>` tags that are replaced by a fieldname descriptor from the meta table. These descriptors will be replaced during system configuration with alternate language versions stored in the meta table.
2. `<@form@>` tags that are replaced with an entry form with features controlled by the meta table: input type, field length, maxlength, etc.
3. `<#value#>` tags that are replaced with actual values from tables during search and viewing.

The meta table currently has 2 scripts that modify it's contents:

1. `metaupdate.pl` – a script to read the current database and update the metatable to match any additional fields added to or fields removed from the underlying tables.
2. `metaedit.pl` – a script to allow edits to any of the field settings in the meta table. Currently this only applies to the student and staff tables. (studentwd and prereg tables are ignored and the settings apply to all 3 tables, since they are identical structurally).

This script allows you to:

- Change the field name descriptor if a `<* *>` tag exists in the template for it.
- Default values – these will be the values available if the form entry type is a selection list. The following rules apply for select entries:
 - Each entry must be separated from the other by a space.
 - Multiword entries they must be joined together with an underscore (.). It will be removed before displaying.
 - Initial tilde characters (~) force a blank at the start of a selection box.
- Formtypes – text input boxes, selection lists, checkboxes, textareas are allowed. Checkboxes in edit screens become text input boxes, since they are impossible to use to clear values.
- View size – controls the size of the box for text input form types.
- Required – controls whether this is a required field when enrolling students.

The **process of customization / table extension** consists of the following steps:

1. Create additional field definitions in a file (`update.sql`) . Use this file to extend the student table (`mysql -ppassword database < update.sql`). Copy and edit this file so that you also extend the studentwd and prereg tables in the same way. (`updatewd.sql`, `updatepre.sql`)

-
2. Edit the template folder to add the new field(s) to the templates at a desired location. This will allow for addition and editing of the new fields during enrollment and demographic updates as well as searching and viewing of the field(s). They can also be printed out on the student viewing page.
 3. Run the metaupdate.pl script to add those new field definitions to the meta table. Each new field in the student table will become a new record in the meta table. Now run the metaedit.pl script, choose to edit the student table and then add your new field characteristics.
 4. You now have a customized version of OA with more student fields. As new releases of OA come out, your custom templates will have to be placed in the new versions's template directory to match your extended student table(s). The meta data will not be altered so this may be left alone.
 5. **If you are using MySQL version 4.x**, the *studentall* table must be rebuilt due to the extended student table . MySQL version 5.x will not require any change since it uses a view. The simplest approach is to first get the data from the current student table and put into a text file:

```
mysqldump -d -ppassword -u user database student > student.sql
```

(where *password* is your mysql user password, *user* is your mysql user, and *database* is your database.)

Now do the same thing to create a data dump of the studentall table:

```
mysqldump -d -ppassword -u user database student > student.sql
```

Edit the student.sql file and change the name from *student* to *studentall*. At the bottom of this file copy the identical section of the studentall.sql file replacing the MyISAM section with the merge table information.

Now drop the studentall table from your database (drop table studentall), and create the new one:

```
mysqldump -d -ppassword -u user database student < student.sql
```

(the student.sql file, now changed, is actually the file used to create the new version of studentall)

You are now done.

Security

This section deals with the issues of making sure that your school data remains intact and accessible only to authorized individuals.

Web Security

Currently, both the Teacher and Admin sites are protected by basic authentication (as outlined in the installation section). This method of authentication sends passwords as clear text over the “wire”. The parent site is freely accessible, but protected by passwords passed to the running script.

This could be improved by moving to *Digest Authentication* which is a step up in security and doesn’t send clear text over the wire. The configuration setup for this is outlined in the Apache documentation.

The next step is to add full https (SSL) support to the server and use certificates to provide banking level protection to the transactions. These instructions are now provided by the current installation and are written by Andy Figueroa.

Another entire area of concern is the use of passwords. Access to the teacher site is by means of a single shared password (via a single password file). Other apache modules could be used to store separate passwords for each teacher in a mysql database (and in fact use the teacher table for these passwords). Apache 2.x has some advanced authentication modules to allow this. Thus the admin site could be used to directly manage teacher access to the website (as well as the attendance entry, marks entry, etc.)

Since passwords are the first line of defense, they should be carefully chosen so that people can remember them, but not be so simple that they can easily be guessed or cracked by a brute force attack.

Some other ideas:

1. Don’t provide links on any publicly visible web sites to your admin or teacher sites. This is only security by obscurity, but it can’t hurt.
2. For the truly paranoid, don’t place the virtual admin and teacher sites into your DNS records. Instead, configure client workstations by placing these records in your “hosts” file (either in /etc/hosts or c:\windows\hosts). The hosts file would contain the IP address and then the domain name of the school/division server.

Maintenance

Multiple Servers

In order to maintain multiple servers running Open Admin, I have done the following:

1. In the */schools* folder (where all schools are located), I have created an *updates* folder with 3 subfolders, *Notes*, *Incoming*, *Outgoing*.
2. In each of the *Incoming* and *Outgoing* folders, place a copy of the *oacopy.pl* perl script and modify them to copy files into all of your school folders.

An upcoming version will have to take into account modifications to a file that are specific to a school. (ie. customization). If those changed files are placed in a directory at the school level (ie. same level as *etc* and *cgi*) instead of where they normally run and then a symbolic link added to the normal folder pointing to the custom folder, a new *oafscopy.pl* script could *detect the presence of the symbolic link* and not copy the update for this school (although it should warn the user of this event).

3. When updating scripts, you would copy the update(s) to the *Outgoing* folder and use *oacopy* to update all of the other schools on the same server. If there are multiple scripts, they can then be tarred/zippped into a single archive file. This file is then copied to the other server using *scp*. I use date archive names (i.e. *051006a.tgz* for the first archive of October 6th, 2005).
4. On the other server, move the archive file into the *Incoming* folder, uncompress it, and then use *oacopy* to update all of the other schools. This can be done quite quickly for small numbers of servers.

In the *Notes* folder, mentioned above, I place notes to myself about particular settings in each school, any customizations for any school, etc. This is particularly important for student numbers since I assign each school a particular range of student numbers (which is stored in the */cgi/entry/studentnumber* file). I use a large range to that this setting would last at least 6 years or better. At that point, a centralized (per division) routine can be used to assign division student numbers. For now, this method of storing the next available student number in a file is simple and reliable.