SciConPHP Installation Document

Denice C. Deatrich

Mar 30, 2002

Last update: Oct 12, 2003

$Id: sciconphp-install.lyx,v 1.12 2003/10/12 16:01:12 deatrich Exp $

Contents

1. Introduction

SciConPHP was designed and written for the 2002 International Symposium on Information Theory (ISIT) conference in Lausanne, Switzerland. At the outset our goal was to produce something that would live beyond the end of the conference. We ourselves are serious users of open-source software, and we would like to return something to the community. Thus the SciConPHP project on SourceForge is our open-source contribution to the Academic community.

Conference software, and in particular scientific conference software, have a few problems:

- The solftware is only used actively for the preparation time leading up to the conference - about a year. Thereafter it falls into disuse, and the only remaining trace is the web server and the conference web pages, assuming that the server is even left in service. No one is very keen to spend money on software that is needed for a such a short period.
- Scientific conferences are frequently hosted by a few groups in a university setting. Students are often pressed into service as webmasters and content-providers. They usually have other time-consuming jobs to do, and consequently cannot spend the time on the conference web site that is needed.
- University groups do not necessarily have the money to spend on manpower, software or equipment, and a home-brewed solution where suboptimal security, web content and site management can be the result.

Perhaps this software will address some of these problems, and therefore be of use to future conference hosting groups.

Note that a conference web site is really just a serial event, in which only certain tasks are important at any time. Thus there is time to modify and tweak the software for your needs – even if your group does not have the current expertise to provide a fully functional web service, there is time to adapt – but do not underestimate the time required for the list of tasks. A typical ordering of web events is:

- provide detailed information about timelines and paper submission requirements
- open up the paper submission pages
- close the paper submission pages, deal with the lose ends, and open up the reviewer pages
- finalize the review process and prepare the technical program
- inform the authors whether or not their paper was rejected and why

- open up the camera-ready paper uploads for accepted authors and provide any other services that are part of preparing the conference proceedings (example, copyright delegation)
- provide detailed information about registration procedures and options, and turn on conference registration pages. Prepare information for local arrangements: hotels, transportation, etc.
- close the camera-ready paper uploads and prepare for conference proceedings publication
- follow up on local arrangement issues, and polish the conference information on your web site. Print name badges, receipts, etc. and finalize the attendee dossiers.
- After the conference, clean up the web site for the long dormant state to follow.

1.1. Thanks!

This software project was made possible by the good-natured support of the organizers and participants of these two conferences hosted at the Swiss Federal Institute of Technology - Lausanne (E.P.F.L.), as well as the excellent network and computing infrastructure and support at E.P.F.L.:

- The 2002 International Symposium on Information Theory, co-chaired by Prof. Bixio Rimoldi and Prof. James Massey and hosted by the laboratories of Communication Theory, Information Theory and Mobile Communications at E.P.F.L. In particular, Professors Emre Telatar and Rüdiger Urbanke both initiated and supported this project, and contributed software and ideas to the project.
- The 2002 IEEE/RSJ International Conference on Intelligent Robots and Systems, chaired by Prof. Roland Siegwart and hosted by The Autonomous Systems Laboratory of the Institute of Systems Engineering at E.P.F.L. In particular, the chief webmaster Thomas Estier is an important contributor of software and ideas to the project.

1.2. License

This project is now hosted at SourceForge.net at the URL:
http://scicon-php.sourceforge.net/.

```
This program is free software; you can redistribute it and/or
modify it under the terms of the GNU General Public License
as published by the Free Software Foundation; either version 2
of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with this program; if not, write to the Free Software
Foundation, Inc., 59 Temple Place - Suite 330, Boston, MA
02111-1307, USA.
```

1.3. Conventions used in this document

- Areas of the documentation that are incomplete for one reason or another are marked thusly:

  (to be completed)

- pathnames are shown in `/fixed/font/`
- commands, names and empasized items *are shown in italics*
- things that should be noticed are sometimes colored red
- comments in configuration file examples in the appendices are `colored blue` while the directives or configuration items are left in black, `but the font is fixed and small` in these examples. Again, items which wish to `draw your attention are colored red`.
- when working at the command-line as an ordinary user the command prompt is a $ and as user root the command prompt is a #

  $ some_command as_an_ordinary_user
  # some_command as_the_superuser

1.4. What is assumed in this document

What is documented here assumes the following environment:

- a linux-based server
- the distribution used is assumed to be Red Hat circa version 9 (it is assumed that, if you are using another distro., then you know or will acquaint yourself with low-level management issues for your distro. I will not document minor issues of other distros.)

In general, various open-source tools are used in the background for this project. The list of tools includes:

- PHP and MySQL
- Apache with mod_php and mod_ssl (this doc. has been updated for apache 2)
- GNU fileutils
- Ghostscript
- bash
- PostScript

Any freenix environment should work admirably. If, however, you wish to press into service a commercial UNIX server, then you may need to compile and configure these tools first. Moreover you need to consider how you will handle security warnings as they arise. You may find yourself recompiling mod_php, for example, because of a widely-known vulnerability. If you 'roll your own' server, then you must also be prepared to patch it as the need arises. Also, compiling and configuring this list of software can be a daunting job if you are not an accomplished configuration manager, but if you want to do it, then it is your choice. I have done enough of this in the past to say 'been there - done that', but it has a certain, rather valuable, educational value **:-)**

This software will also work fine with the apache version one server. Any apache1 issues are discussed in the appendix.

1.5. What this software will not do for you..

The software is approaching a state which could be called 'feature-complete' – which is to say that much of the web functionality needed to host a conference is implemented. However, web software is not really like other genres of software. You cannot simply install it and use it without getting your hands dirty, so to speak. A list of tasks that require your intervention includes:

- You need to provide your own content. Thus you must roll up your sleeves and learn enough HTML and PHP for this purpose.
- Moreover, not all aspects of conference serving are implemented. Some tasks should NOT be implemented, because they are too dangerous (at least in my opinion), or are likely to need extensive customisation for your particular needs. An example is mass-e-mailing to your conference participants. At some point you need to send out your notifications of acceptance or rejection to your candidate authorlist. This sort of task is better accomplished via a manually-launched, intelligent script, rather

than the click of a button in a web page. Thus you will need to become somewhat proficient at shell scripts. Some example shell scripts are provided with the source code, but you will almost certainly have to alter them.

- You cannot escape learning enough SQL to survive. Some database changes will need to be done manually. You must learn MySQL, and you must learn it sooner rather than later. It is easy to wipe out your database completely if you don't know what you are doing. If you use this software then it is highly recommended that you become comfortable with SQL (and as my mother would say: 'it won't kill you').

2. Installation

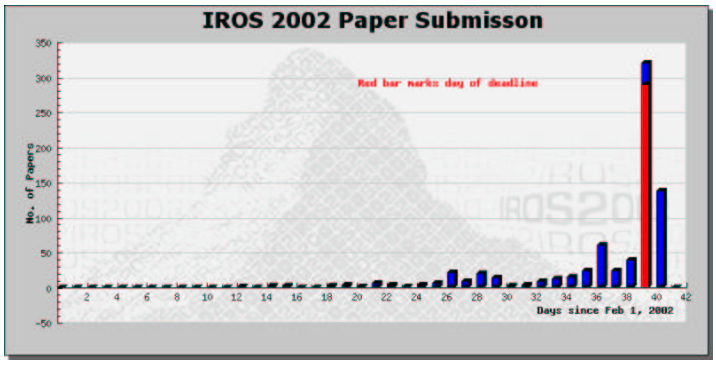
You may be wondering why so much time is spent documenting the steps leading up to the installation of SciCon^{PHP}. However, it is an important part of the process that is frequently glossed over with web projects, and this is an opportunity for me to expound on some issues that I think are frequently neglected (and besides, I happen to like mind-numbing details :-). I have helped a lot of newbie web masters here at EPFL in the past few years, and I have frequently been amazed at how little needs to be known to run a web server. An important distinction here, however, is that forms-based, dynamic web services provided by software like SciCon^{PHP} are very different from simple content provision (static HTML pages). Forms-based services are potentially dangerous, and the web master should make a serious effort to know the technology he/she is using. Experienced unix admins and webmasters will skim through much of this document; others may find this 'digital verbosity' useful (I hope).

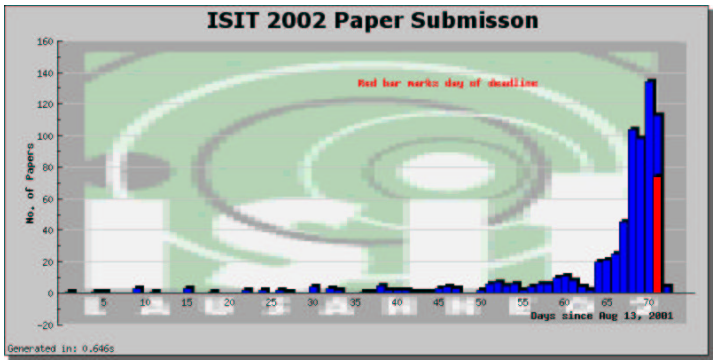
2.1. Choice of Hardware and Server Installation

Theoretically any old intel machine could be your web server. There are a few issues to consider, however, before you free up that circa 1998 PII, 300 MHz machine from it's printing duties and turn it into a web server:

- Do you have a backup solution if your aging PC decides to part with its soul? Of course, you will need a backup solution anyway, but it is always nicer if the machine is under a support contract, and you at least have to comfort of knowing that it can be serviced within 24 hours.
- Do you have enough CPU power to support file testing on uploaded papers? SciCon^{PHP} supports ghostscript testing of PDF and PostScript files for 'printability'. This task is rather CPU intensive, and requires present-day compute power.
- Do you have the disk space and the network bandwidth to cope with the last day deluge of uploaded papers? A few graphs illustrate how 'wild' the

few days around the deadline can be – in particular more than a gigabyte of file uploads (approximately 330 papers) occurred during the day of the deadline for the 2002 IROS conference, while the 2002 ISIT conference received more than 100 papers per day for 4 days:





For each of these conferences a Dell PowerApp was used (I believe it is now known as the PowerEdge 1650 server). The systems were purchased with 3 fast-wide SCSI disks and 256 MB of RAM. The cost at the time was approximately 3500 $US for this configuration. Though this is perhaps 'overkill', the machine can be adapted for other services after the conference enddate.

Though the systems were delivered with linux pre-installed, they were re-installed to allow us to set up software raid according to our needs. A RAID 1 (one) was configured on two of the disks to create a set of redundant, mirrored system partitions. Thus in case one disk failed the other disk could maintain the service. The third disk was used primarily for local backups. Red Hat Linux was installed and all patches were applied. Patching your system is of course very important. There are many resources on the Internet for keeping up-to-date your freenix installation. I hope that your institute makes these resources as readily available as E.P.F.L. does for us.

Standard Red Hat web packages were used, which is to say that no software was compiled. The disk partitioning was the following:

(Mirrored) Partitions	Size in **MB**
/	300
/usr	4000
/tmp	500
/var	6000 - 15000

2.2. General installation issues

Ideally a web server suitable for running a conference should be a stand-alone machine. From both a security and a reliability point of view the machine should not be doing other tasks, like samba, general mail service (imap, pop,..), print serving, home directory serving, etc. If you also use the *conference registration* software, and decide to take people's credit cards online (see the section on 'credit card security' near the end of the document), then it is extremely important to do a secure 'nail down' of your machine. This almost certainly means that it is a web server, and only a web server.

If you are not processing credit cards, then these issues are less important, but should not be ignored. Here is a small list of what should be done to generally secure your machine before you put it into production. These are very easily implemented.

- at installation time on a Red Hat system, do a custom installation or a server installation. If you do a custom installation only install those packages that are relevant.
- Specifically, do not enable anything that *xinetd* controls, like *telnetd*, *rlogind* or *ftpd*. Happily these services are not enabled by default, and you have to hunt and peck through the documentation to figure out how to enable them. They will not work for a remote *root* user anyway unless you _really_ know how to sift through the documentation. Secure shell (and thus it's brethen like 'secure copy' (*scp*)) are infinitely more secure, are configured by default, and just 'work'. There are excellent secure shell clients like PuTTY for Windows, for example – thus there is no need to fiddle with aging daemons like *telnetd*. The syntax of *scp*, for example, greatly resembles its cousin *rcp*. Thus you can copy a directory and everything below it with a command like the following, thus avoiding *ftp* entirely:

  scp -pr thisDirectory webowner@sciconphp.somewhere.zz:/path/to/someplace/

- use the so-called 'libwrap' configuration files, `/etc/hosts.allow` and `/etc/hosts.deny` to close off your machine. Examples of these files are in the appendix at item F. Note that you should not allow global secure-shell access to your machine either. This is just lazy. Use instead the *hosts.allow* functionality to allow access from your list of trusted machines onto your web server.

If your secure shell daemon was not compiled with *libwrap* support, then you will have to secure it using other means, like the configuration file for *sshd*.

2.3. Apache/PHP/MySQL Installation

The prerequisite web packages on a Red Hat system for SciConPHP are: *httpd*, *mod_ssl*, *php*, *php-mysql* and *mysql*. If the mysql server daemon will be running on the same machine then the *mysql-server* package is also required. *mod_ssl* is only required if you plan on providing some SSL-secured web services. Other web packages that are not used should be removed; for example, *mod_perl* or unnecessary *php-** packages. If you want to generate some nice site statistics graphs, then you should also install jpgraph jpgraph.

NOTE: For an **apache 2** installation (circa Red Hat 9) **it is very important to use the latest patches for both apache and for php**. As of July 2003 the last php patch from red hat was version 4.2.2-17.2. Before the end of 2003 there should be another patch that will be worth applying. In other words, follow the patch trail very closely.

`/var` is the home of the web tree. If the mysql database is also installed on the system then it also lives in `/var`. If you decide to put the downloaded papers here as well, then this partition needs to be large.

Note that almost all internet services on a Red Hat system are not enabled by default, even though you may have chosen to install them. You typically have to both manually launch them the first time, and also configure them to start authomatically after subsequent reboots. Here are some examples with *named*, *mysqld* and *httpd*:

Manually-started daemons:

/etc/rc.d/init.d/named start
/etc/rc.d/init.d/mysqld start
/etc/rc.d/init.d/httpd start

Enable auto-startup daemons on subsequent reboots:

chkconfig –level 2345 named on
chkconfig –level 345 mysqld on
chkconfig –level 345 httpd on

2.3.1. PHP Configuration

The configuration directives for PHP are set by the `php.ini` file, which is found in `/etc/` on a Red Hat system. While you can do your PHP configuration here, it is better to set the relavent parameters in your apache configuration file instead. This has the advantage of allowing you to host many web trees on the

same machine, while also allowing you to massively change your php options for each web server. Thus the best thing to do is to secure the `php.ini` file completely, and rely on your apache configuration to set specific parameters. Some suggested options to change to have a well-secured global `php.ini` file can be found in appendix item A. The needed PHP options for the web server are documented in the next subsection on Apache configuration.

If, however, your web server is only doing conference web serving, then you can make your changes globally in /etc/php.ini

2.3.1.1. Notes about additional PHP directives for large-file downloads.

In order to allow files larger than 2 MB to be uploaded to the server, you must adjust the default PHP values for the following directives copied from the example in the appendix. The following values are suitable for files up to 30 MB in size. Note that whether or not you need to set the *memory_limit* item depends on how php was compiled on your system.

```
#
          php_admin_value memory_limit          30000000
          php_admin_value max_execution_time    300
          php_admin_value post_max_size         30000000
          php_admin_value upload_max_filesize   30000000
#
```

Additionally, if you have the *LimitRequestBody* directive for php files set in your apache2 configuration, then you will need to bump up its value, or comment it out.

```
#
<Files *.php>
         SetOutputFilter PHP
         SetInputFilter PHP
         ## commented out:  LimitRequestBody 8000000
</Files>
```

2.3.2. Apache Configuration

I assume that you do not run apache as user ***root***; it is totally unnecessary, and potentially dangerous. This sort of thing is passé.

In general, even if you do not plan to do virtual hosting from your web server, you should still *virtualize* your web host anyway. This means that you move as much of the relavent configuration (including the PHP directives) into the virtual host section of the apache configuration file as you can, instead of leaving it scattered through the file. Thus if you ever change your mind and want to add other virtual hosts to your server, then it is trivially done. Moreover you will have gathered the important information together in one place, and it will be easier to understand and find (in my opinion anyway). A complete example configuration file can be found at (http://sciconphp.epfl.ch/confegs/). There are a few items worth noting:

- in section 2 (the 'Main' server configuration) access to the root of the file system is closed, according to the recommendations of the Apache group:

```
<Directory />
        Options None
        AllowOverride None
        Order deny,allow
        Deny from all
    </Directory>
```

- **No** DocumentRoot directives are found outside of section 3 (the 'virtual host' section)
- the PHP section(s) has also been moved into section 3 (the 'virtual host' section)

A smaller example extraction with the PHP directives can be found in appendix item B. It contains 3 example virtual hosts. Suppose that the machine's hostname is *confserv.somewhere.zz*

1. A PHP-enabled virtual host suitably configured for hosting a SciCon^{PHP} site. It has a hostname alias *sciconphp.somewhere.zz* and is rooted at `/var/www/html/sciconphp/`
2. A PHP-enabled secured host suitably configured for hosting a SciCon^{PHP} site. Because it uses the SSL module it must use the assigned hostname *confserv.somewhere.zz* and is rooted at `/var/www/html/secure/`
3. A general vhost that is not PHP-enabled. It has hostname alias *misc.sciconphp.zz* and is rooted at `/var/www/html/misc/`

2.3.2.1. Additional noteworthy Apache issues.

- The file permissions of the web tree are very important; they are discussed in the subsection on SciCon^{PHP} Installation ahead.
- The document root of the apache tree on a Red Hat system is at `/var/www/html/`. I find it preferable to root the virtual web trees in subdirectories below this level.
- If you decide to enable DNS hostname lookup in `/etc/httpd/conf/httpd.conf`, then it is a good idea to install and enable a caching DNS daemon on your web server. I find *resolved* hostnames useful for monitoring the apache log files and also for understanding strange problems that users sometimes have when downloading papers to the site. The RPM packages required are ***bind*** and ***caching-nameserver***. Once again, the naming server daemon ***named*** should not run as root – it has not run as root on a Red Hat system since at least RH 7.0. Moreover your bind configuration should force the daemon to only listen on localhost, and ignore all internet traffic. An example named configuration file, `/etc/named.conf`, can be found in appendix item D.

- The SciCon^{PHP} web server will be sending out a lot of email. Thus a functional setup for sending mail out must also be configured. The default setup for *sendmail* on recent version of Red Hat linux works just fine for this purpose, and nothing need be done. By default the *sendmail* daemon listens only the localhost port. It does not receive mail, and will blythly ignore all external attempts to connect to the sendmail port. This is a Good Thing for a web server.
- If you want to keep more than 4 log rotations of the apache logs, then change the rotate option in `/etc/logrotate.d/httpd`. For example, to keep 8 rotations, add the option:
  `rotate 8`

2.3.3. MySQL Configuration

Whether you install the MySQL server daemon on your web server, or on another server, there are some important issues to address at installation time related to database security.

- Once again, the MySQL server daemon ***mysqld*** must not run as *root* – it does not run as *root* on recent Red Hat systems, but rather as user '*mysql*' with the database installed at `/var/lib/mysql`
- You must set a password for the so-called MySQL super user: ***root*** (Note that this is **_not_** the same user as the ***UNIX root user***). By default a password for the MySQL superuser is not set. This is a Bad Thing. You will typically only use the mysql root account for setting up the initial database – after that you will rarely use it. To set a MySQL root password, do the following. Note that it sets a password for the mysql user root:
  - *mysqladmin -u root password 'putYourPasswordHere!'*
- You should eliminate all traces of the test database and the test user which are installed by default. This test database has no password, and may connect from anywhere. We will also delete the entry for the *root* user not coming from *localhost*:

```
# mysql -u root -p mysql
Enter password:
mysql> select host,db,user,select_priv from db where db like 'test%';
+------+---------+------+-------------+
| host | db      | user | select_priv |
+------+---------+------+-------------+
| %    | test    |      | Y           |
| %    | test\_% |      | Y           |
+------+---------+------+-------------+
```

```
mysql> select host,user,password,select_priv
from user where password= "";
+-----------+---------+----------+-------------+
| host      | user    | password | select_priv |
+-----------+---------+----------+-------------+
| myhost    | root    |          | Y           |
| localhost |         |          | N           |
| myhost    |         |          | N           |
+-----------+---------+----------+-------------+
mysql> delete from user where password="";
mysql> drop database test;
```

2.3.3.1. Setting up the database for SciCon^{PHP} WE need to create the database, and then we need to create the 2 mysql user accounts with appropriate passwords and privileges. Moreover we assume that these two users will be able to see all the tables within this database. We will create the tables shortly, but first we must set up a skeletal MySQL environment. We suppose that the database will be called *'confdb'* and that there will be two users called *writer* and *reader*.

- *writer* is a user with full write-privileges to the database.
- *reader* is a user with read-only access (ie. SELECT only) to the database.

We will allow connections to the database for two sources – the localhost address and any machine in our subnet.

We connect as the MySQL user '*root*' to the '*mysql*' database and we set up the environment:

```
# mysql -u root -p mysql
Enter password:
mysql> create database confdb;
mysql> grant SELECT,INSERT,UPDATE,DELETE,CREATE,DROP,INDEX,ALTER
on confdb.*
to writer@localhost identified by 'yourWriterPassword',
 writer@'192.168.196.%' identified by 'yourWriterPassword';

mysql> grant SELECT
on confdb.*
to reader@localhost identified by 'yourReaderPassword',
 reader@'192.168.196.%' identified by 'yourReaderPassword';
mysql> select host,db,user,update_priv from db;
```

```
+---------------+-----------+-------+-------------+
| host          | db        | user  | update_priv |
+---------------+-----------+-------+-------------+
| localhost     | confdb    |writer | Y           |
| 192.168.196.% | confdb    |writer | Y           |
| localhost     | confdb    |reader | N           |
| 192.168.196.% | confdb    |reader | N           |
+---------------+-----------+-------+-------------+
```

```
mysql> select host,user,password from user;
+------------------+---------+------------------+
| host             | user    | password         |
+------------------+---------+------------------+
| localhost        | root    | 4bfb845d75d861df |
| 192.168.196.%    | writer  | 5983377a23c52280 |
| localhost        | writer  | 5983377a23c52280 |
| localhost        | reader  | 7e4ed4ff4a749632 |
| 192.168.196.%    | reader  | 7e4ed4ff4a749632 |
+------------------+---------+------------------+
```

Now we do a quick test – exit the mysql shell and connect as user '*writer*' to the newly created database '*confdb*' using its password:

```
# mysql -u writer -p confdb
Enter password:
mysql> show tables;
Empty set (0.00 sec)
```

It is time to create the tables. Stay connected as 'writer' and use the *source* command to create the table definitions. We are skipping ahead a bit here, since we haven't installed the source code yet for SciCon^{PHP}. However we assume for the moment that we have installed it, and that the included SQL script file for creating the database tables is available at `/var/www/outside/misc/db/createtables.sql`

```
mysql> source /var/www/outside/misc/db/createtables.sql;
mysql> show tables;
+------------------+
| Tables_in_isitreg |
+------------------+
| attendee         |
| authorlist       |
| chairs           |
| finalpaper       |
| paper            |
```

```
| people            |
| regcat            |
| reviewer          |
| revpaper          |
| revpoints         |
| revquality        |
| sessionlist       |
| sessions          |
| sessiontimes      |
+-------------------+
16 rows in set (0.00 sec)
```

(to be completed)

2.4. SciCon^{PHP} Installation

2.4.1. Directory Structure

When the SciCon^{PHP} package is unpacked it has the following major directory structure. Two, and only 2, of these directories must be owned by *apache*. All other directories must be owned by someone else! The directories which must be surrendered to the *apache* daemon are marked in red:

1. `inside`
    - `inside/protect`
2. `outside`
    - `outside/misc`
    - `outside/inc`
    - `outside/forms`
    - `outside/papers`
    - `outside/scripts`

2.4.1.1. 'inside' The directory named `inside` corresponds to the root of the web tree html files as configured for your server. One particularly important directory is:

`inside/secured` This is the location for secured access to your site's administration tools. In the example apache configuration I have treated this directory in a special way to make it easier to synchronize php code changes from my development source files to the web tree. Thus I have disabled access to this tree from the regular web port (port 80), and then I have

rooted the secure-web port (port 443) in this subdirectory. When maintaining both a test http and https web site and a production http and https web site it is far easier to compare and/or synchronize code between the sites if all code for all sites resides under the fewest number of web trees possible. Thus all code really does live under only /outside/inc and /inside trees.
If you are not going to enable credit-card processing, and you decide to forgo any secure web (https) access then you can simplify your apache configuration a little for this directory.

Another noteworthy directory is:

`inside/secured/protect` is the location of web pages that require password-protected access, as provided by the apache server. You must edit the `.htaccess` file in this directory to suit your needs, and you must also create the apache password entry corresponding to this file using the apache utility *htpasswd*. You must also configure apache to use this access file. The `httpd.conf` example as previously mentioned in the appendix includes the authorisation directives for this 'protected' directory. There is also an example .htaccess file in the appendix at item C. An example of the command used to create the password entry is:

htpasswd -m /etc/httpd/conf/passwd YourUserName

2.4.1.2. 'outside' The directory named `outside` corresponds to the location of the files referenced by the SciCon^{PHP} software, but which do not need to be in the apache document root.

It contains four directories:

`outside/misc` is the location of documentation and database description files. This is the directory that contains the MySQL script that you will use initially to create the database structures. Note that the web server never actually references anything in this sub-directory.

`outside/inc` is the location of the libraries and include files used by SciCon^{PHP}

`outside/papers` is the place where the downloaded papers will be stored. The papers do not have to go here; it is just one possibility. All that matters is that it is owned by apache, that it not be in the document root, and that it has sufficient disk space for all downloaded papers.

`outside/forms` is the location of temporary downloadable fill-in forms created and formatted on demand as postscript or pdf by the web server. You must configure a crontab entry to clean this directory regularly. An example crontab entry is available in the Appendix at item E. Note that this directory is aliased in `httpd.conf` as `/forms` for your web server.

`outside/scripts` is the location of various supporting scripts used while managing SciCon^{PHP}

2.4.2. Assumptions

We will assume for the purpose of describing how to install the software, and how to set filesystem permissions, that you are using the following configuration:

- the apache daemon is configured à la Red Hat, and is therefore owned by a user:group named:

  apache:apache

- the owner of the web tree and the external tree will be a user:group which we have previously created (with a utility like '*useradd*'). This is the user account which can edit files in the web tree:

  webowner

- we assume that the ownership of the web tree and the external tree is shared (to be explained shortly) with a group we have previously created (with a utility like '*groupadd*'). The name of the group is *webgrp* and we add the user *webowner* to the group. Thus the user:group name will be:

  webowner:webgrp

- the web tree will be located at:

  `/var/www/html/inside/`

- the external directory tree will be located at:

  `/var/www/outside/`

2.4.2.1. Web Tree Ownership and Shared Access In academic setups, (I have observed that) it is common for groups to have shared access to web trees (amongst other things :-). It is also common enough for many people to potentially have login access to servers, including possibly students you may not trust. These issues must be considered when you are setting up a conference web server!

For now let us consider a situation where a few trusted people will be editting the web tree. Under linux you can add these people to a group, in our example *webgrp*. Then you would set any shared file trees with mode 'g+w' (allowing group write permissions) and set the directory permissions to be 'setgid' with mode 'g+s'. This generally preserves file and directory permissions for the group.

2.4.3. Installing the software

The installation of the software may either be done by the supplied script, or manually. Either way you need to create a few directories and set the permissions properly as root first.

2.4.3.1. Prepare the directory structure

- make the supporting directory structure (this example assumes a Red Hat system with a large `/var` directory as previously mentioned).

  # mkdir /var/www/outside /var/www/html/inside/

- you may want to set a couple of symbolic links to make file access easier:

  # ln -s /var/www/html/inside /inside
  # ln -s /var/www/outside /outside

- now set the preliminary permissions:

  # chown webowner:webgrp /outside /inside

2.4.3.2. Unpack the software

- unpack the sources anywhere; for example in `/home/webowner/src/` As user *webowner* do the following:

  $ cd $HOME/src
  $ tar zxf sciconphp-release-xyz.tgz
  $ cd sciconphp-release-xyz

2.4.3.3. Script-based installation

- cd into the directory where the install script resides:

  $ cd /usr/local/src/sciconphp-release-xyz/outside/scripts/install/

- copy the example install configuration file `scicondir.in` to a file called `scicondir` in the same directory, and edit it to indicate where the two major directories are, as well the uid and gid of the owner:group, the desired file modes and the path to a backup directory. It should be self-explanatory.

- then run the install program in demo mode (does not install itself yet)

  $ sh install.sh

- if the output looks okay, then really 'do it'

  $ sh install.sh -doit

- if there were no errors, then repeat the install as user *root*. Once again run it twice – in demo and in real mode. You would normally only ever have to do this once as *root*. Any future installs into the existing web trees as *webowner* cannot change what root has configured:

  # sh install.sh
  # sh install.sh -doit

2.4.3.4. Manual installation

- cd into the top directory of the software:

```
$ cd $HOME/src/sciconphp-release-xyz/
```

- copy files to their respective homes

```
$ alias cp='cp -i'
$ cp -a inside/* /var/www/html/inside/
$ cp -a outside/* /var/www/outside/
```

- set ownership

```
$ chown -R webowner:webgrp /inside/. /outside/.
$ chmod -R g+w /inside/. /outside/.
$ find /inside/. /outside/. -type d -exec chmod g+s {} \;
```

- now as user *root*, set the ownership of files which should not be owned by *webowner*. Note that we additionally remove all permission to read the database connect file (`i_dbConnect.php`) except for the apache web server and anyone in group 'webgrp'. Whether or not you decide to do this depends on the level of trust you have in people with login access to your web server:

```
# chown -R apache:apache /outside/papers /outside/forms
# chmod 660 /outside/inc/i_dbConnect.php
# chown apache:webgrp /outside/inc/i_dbConnect.php
```

2.4.4. Configuring the software

- edit the 2 files that describe your setup.[1] There are two files to edit; they are documented internally, and should be self explanatory:

```
$ editor /outside/i_dbConnect.php
$ editor /outside/i_definesSite.php
```

Note that subsequent installs of the software will not overwrite these two files. You may need to compare these two files with the distributed versions to see if you need to manually merge any changes into them.

[1] There are a zillion editors in the unix world (almost :-). The best for general system management is ***vi***. If you don't know it, then think about learning this editor. It is _the_ universal editor for UNIX/freenix systems. It works very well on other systems as well; in fact, Graphical Vi is brilliant on Windows; I can't imagine _not_ having it installed on a Windows system. Nothing is better than having it in your right-mouse menu, ready to edit any file you feel like, disregarding artificial file extension naming. There are many tutorials on the web. After half an hour spent with a tutorial you will already be functional enough to work in vi.

2.4.5. Updating the software

This is unknown territory – be very careful about blindly updating your web tree with any new versions of the software. You have probably extensively customized your software, and should only do an update on a test web site, and only after looking through the source code and the demo web site for structural and functional changes.

3. Configuring SciConPHP

3.1. Security issues

3.1.1. Wither SSL..

If you are not taking credit cards online, then you might decide to *not* configure SSL into your server. This is especially true if you do not purchase a commercial-grade certificate from a certifying authority. Some users will be confused when their browser complains about your server's certificate being invalid or untrustworthy. Moreover it could be argued that the contact author's password is unlikely to be target for crackers snooping the network. Any changes to a submitted paper's details are emailed back to the contact author anyway.

You might decide to configure SSL just for the administrative tree(s) of your server, which are only visited by your group. This is useful because the traffic between the server and your clients will be encrypted, including the login access to your administrative pages. In this case you will need to provide two web tree configurations: one for the general web tree, and one for the administrative pages. For this case you do not need to worry about purchasing a certificate for your server. You may instead sign your own certificate.

3.1.2. Credit card security

The necessary code for taking 'semi-online' credit card information is included. By 'semi-online' I mean that a form for credit card information allows us to retrieve and store the credit card details. However the credit card transaction is manually done using debit machines rented for this purpose. It is of course preferable to avoid credit card issues where possible. Thus you should investigate online services available in your country which transfer the burden of online transactions elsewhere. You should aim to make the decision about your payment policies, and purchase the necessary service contracts, at least 4 months prior to opening online registration on your web site.

If you use the code provided for taking credit cards, then you MUST purchase a commerical-grade certificate for your server, and you must configure the registration pages to force users over the secure connection while taking their credit

card information. In my opinion you must also allow people to fax their credit card information instead, if they so chose. The code is included which allows this. There will always be a few people who will claim that your secure server "doesn't work". In fact, we have seen a few cases where it seems likely that firewall configurations on their site interfere with connections over the HTTP SSL port (number 443).

There are some extra precautions to take as well to better secure credit card information on your server:

- limit login access to trusted people on your server.
- close all ports except for the web ports, 80 and 443. Limit the secure shell port (22) to trusted machines. Close all unnessary services on your machine. If your database server is on another machine, then apply the same policies to that server.
- do not log the credit card numbers in the transaction logs (`/path/to/outside/papers/log*`). Note that the provided code inserts a phony number into the transaction logs. Do not change this.
- If you make a copy of the database onto a separate test machine, then the *first* action you should take is to remove the credit card numbers on your test machine:
  `mysql> update reg set ccnum = '9999 9999 9999 9999';`
- When the conference finishes you should take one final hardcopy backup of the database, and then wipe the numbers from the database using the commands above. You should also remove other online archived backups of the database residing on the server.

4. Management Issues

4.1. Configuring a test web server too..

It is difficult to host a conference web site without editting the web pages from time to time. For static html pages you can usually fiddle with the real web pages without dire consequences. However when using PHP and especially when editting global include files it is easy to introduce parse errors. Your website will appear odd, unpleasant and unprofessional to anyone who is viewing pages at the time that you accidentally introduce these errors. Thus it is infinitely better to have a test web site where you may fiddle to your heart is content without detriment to your audience. Of course, you also need a test database for your test site. The very last thing you want in the world is to accidentally alter your data, or worse...

Setting up a test site is not difficult. You have 2 choices:

1. Set up a test site and test database on the same host. You use a different port for the test web server. Of course you must be careful to distinguish the test site from the real site when you are editting files! I don't recommend this solution unless you are an apache pro.

2. Set up the test site on another machine in your local network. This is safer, but less convenient for merging your changes between the test site and the real site. Happily the test site does not have to be very powerful. And it can be invaluable for testing patches. If the latest security patch for *xyz* comes out, and you wish to apply and test the patch first in a safe environment, then a remote test site is exactly what you need for this situation. Moreover, a test server can be used temporarily as the conference web server in case your web server has a hardware failure.

(to be completed)

4.2. Backing up your server and your data

Before you open your web site for production you must implement a backup plan. There are a few kinds of 'backups' to consider:

1. Backing up the data to archival media
2. Mirroring the data locally where possible
3. Planning for an alternate service in case your server dies

(to be completed)

5. Miscellaneous

5.1. Demo server

There is a demo server at
http://sciconphp.epfl.ch/

The database is frequently wiped, and a default database restored.

6. Appendix

A. php.ini

```
[PHP]
; Only the relevant parts of the php.ini file are shown.
;;; see the default /etc/php.ini file for the full documentation ;;;

; Changing the following options is rather crippling, but it does
; protect your machine from accidentally allowing a non-secured web
; tree from 'escaping'.
; You can set more lenient options you need in the apache configuration
; file.

;;; Safe Mode ;;;
;
safe_mode = On

;;; Error handling and logging ;;;

; Print out errors (as a part of the output).   For production web sites,
; you're strongly encouraged to turn this feature off, and use error logging
; instead.
display_errors = Off

;; Data Handling ;;
;
; You should do your best to write your scripts so that they do not require
; register_globals to be on;  Using form variables as globals can easily lead
; to possible security problems, if the code is not very well thought of.
register_globals = Off

;;; Paths and Directories ;;;

include_path = "."
;
;;; File Uploads ;;;

; Whether to allow HTTP file uploads.
file_uploads = Off

;;; Fopen wrappers ;;;

; Whether to allow the treatment of URLs (like http:// or ftp://) as files.
allow_url_fopen = Off
```

B. httpd.conf - Apache 2

```
## httpd.conf -- Apache ver 2 HTTP server configuration file, see
## (http://sciconphp.epfl.ch/confegs/)

### Section 1:  Global Environment

## <snipped>

### Section 2:  'Main' server configuration

User apache

Group apache

ServerAdmin webmaster@somewhere.zz

UseCanonicalName Off

## DocumentRoot stuff removed; see virtual hosts section instead

## Lock up primary permissions:

<Directory />

      Options None

      AllowOverride None

</Directory>

## <some snipped out>

Alias /forms "/var/www/outside/forms"

<Directory "/var/www/outside/forms">

      Options None

      AllowOverride None

      Order allow,deny

      Allow from all

</Directory>

## <some snipped out>

### Section 3:  Virtual Hosts

## virtual hosts have their own configuration file

NameVirtualHost 192.168.190.10:80

Include conf.d/sciconphp.conf

NameVirtualHost 192.168.190.10:80

Include conf.d/misc.conf

<IfModule mod_ssl.c>

      Include conf.d/secure.conf

</IfModule>

### end
```

C. sciconphp.conf (apache)

```
## Included PHP configuration for conference, see
## (http://sciconphp.epfl.ch/confegs/)
<VirtualHost 192.168.190.10:80>
ServerName sciconphp.somewhere.zz
ServerAlias sciconphp
DocumentRoot /var/www/html/sciconphp
DirectoryIndex index.php index.html
<Directory "/var/www/html/sciconphp">
      Options SymLinksIfOwnerMatch
      AllowOverride AuthConfig
      Order allow,deny
      Allow from all
</Directory>
<Directory "/var/www/html/sciconphp/secured">
      Options None
      AllowOverride None
      Order deny,allow
      Deny from all
</Directory>
# Cause the PHP interpreter handle files with a .php extension.
<Files *.php>
      SetOutputFilter PHP
      SetInputFilter PHP
      # LimitRequestBody 8000000
</Files>
<IfModule sapi_apache2.c>
      php_admin_flag safe_mode Off
      php_admin_value include_path "/outside/inc:."
      php_admin_flag file_uploads On
      # php_admin_flag display_errors Off
      php_admin_flag display_errors On
      php_admin_flag log_errors On
      php_admin_value error_log "/tmp/phperrors.log"
      php_admin_value max_execution_time 120
      php_admin_value post_max_size 30M
      php_admin_value upload_max_filesize 30M
      php_admin_value memory_limit 30M
</IfModule>
Include conf.d/conferrors.conf
ErrorLog /var/log/httpd/sciconphp.error_log
CustomLog /var/log/httpd/sciconphp.access_log combined
</VirtualHost>
```

D. httpd.conf - Apache 1

```
##

## httpd.conf -- Apache ver 1 HTTP server configuration file, see
## (http://sciconphp.epfl.ch/confegs/)

### Section 1:  Global Environment

## <snipped>

### Section 2:  'Main' server configuration

## <snipped>

### Section 3:  Virtual Hosts

NameVirtualHost 192.168.190.10

<VirtualHost 192.168.190.10>

ServerName sciconphp.somewhere.zz

ServerAlias sciconphp

ErrorDocument 400 /error.php

ErrorDocument 401 /error.php

## ... <snipped>

DocumentRoot /var/www/html/sciconphp

<Directory "/var/www/html/sciconphp">

      Options SymLinksIfOwnerMatch

      AllowOverride AuthConfig

      Order allow,deny

      Allow from all

</Directory>

<Directory "/var/www/html/sciconphp/secured">

      Options none

      Order deny,allow

      Deny from all

</Directory>

DirectoryIndex index.html index.php

Alias /forms/ "/var/www/outside/forms/"

<Directory "/var/www/outside/forms">

      Options None

      AllowOverride None

      Order allow,deny

      Allow from all

</Directory>

<IfModule mod_php4.c>

      AddType application/x-httpd-php .php4 .php3 .php

      AddType application/x-httpd-php-source .phps

      php_admin_flag safe_mode Off

      php_admin_flag register_globals Off
```

```
        php_value include_path "/outside/inc:/var/www/more:."
        php_flag allow_url_fopen Off
        php_flag file_uploads On
        php_flag display_errors On
        php_value max_execution_time 500
        php_admin_value post_max_size 30M
        php_value upload_max_filesize 30M
        # php_flag display_errors Off
        php_flag log_errors On
        php_value error_log "/tmp/sciconphp-errors.log"
</IfModule>
ErrorLog /var/log/httpd/sciconphp.error_log
CustomLog /var/log/httpd/sciconphp.access_log combined
</VirtualHost>
<VirtualHost 192.168.190.10>
ServerName misc.somewhere.zz
ServerAlias misc
DocumentRoot /var/www/html/misc
<Directory "/var/www/html/misc">
        Options SymLinksIfOwnerMatch
        Order allow,deny
        Allow from all
</Directory>
DirectoryIndex index.html index.php
ErrorLog /var/log/httpd/misc.error_log
CustomLog /var/log/httpd/misc.access_log combined
</VirtualHost>
## SSL Virtual Host Context
<IfDefine HAVE_SSL>
## <snipped>
</IfDefine>
```

E. Example .htaccess file

```
AuthUserFile    /etc/httpd/conf/passwd
AuthName        "admin"
AuthType        basic
require         valid-user
```

F. Example named configuration

```
// /etc/named.conf:  generated by named-bootconf.pl
options {
        directory "/var/named";
        // put your own global DNS servers here; these are pretend ones:
        forwarders {
                192.168.1.10;
                192.168.2.10;
        };
        listen-on { 127.0.0.1; };
        /*
         * If there is a firewall you may need to uncomment query-source
         */
        // query-source address * port 53;
};

// a caching only nameserver config
controls {
        inet 127.0.0.1 allow { localhost; } keys { rndckey; };
};

zone "." IN {
        type hint;
        file "named.ca";
};

zone "localhost" IN {
        type master;
        file "localhost.zone";
        allow-update { none; };
};

zone "0.0.127.in-addr.arpa" IN {
        type master;
        file "named.local";
        allow-update { none; };
};
include "/etc/rndc.key";
```

G. Example crontab entries

```
# DO NOT EDIT THIS FILE - edit the master and reinstall.
# (/tmp/crontab.15133 installed on Thu Aug 23 10:21:38 2001)
#
## Example entry to clean up temporary areas in a web downloads area
## -> in this case:  /var/www/outside/forms
## -> we remove twice a day (at 12:20 and 00:20) any files older than 2 days
20 0-23/12 * * * find /var/www/outside/forms -path '*/forms/*' -mtime +2 -exec rm {} \; 2>/dev/null
#
```

H. hosts.allow and hosts.deny

H.1. /etc/hosts.deny

```
#
# /etc/hosts.deny
#               Start by denying EVERYTHING!
ALL : ALL
#
```

H.2. /etc/hosts.allow

```
#
# /etc/hosts.allow
# Now allow only those hosts and services that you trust!
#               We can trust the loopback interface (localhost)
ALL : localhost
#               On the test web server only, we may have some NFS client needs where we need
#              to allow portmap access (it needs IP addr format!)  on the local subnet
# portmap :  192.168.196.0/255.255.255.0
#               Now we configure secure shell access, include X11 tunnels.
#              Our trusted hosts will be the subnet (somewhere.zz) and
#               hosts matching xx*.mit.edu and 192.168.1.42
sshd sshdfwd-X11 :  *.somewhere.zz  xx*.mit.edu 192.168.1.42

#
```