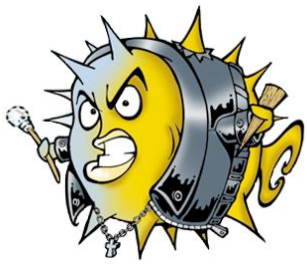

Installation Guide

OpenBSD 3.9 with Gallery v2.1.1a

(Apache chroot'd with Perl, Hardened-PHP & PostgreSQL database)

Version	Date	Author	Notes
0.1 DRAFT	10 June 2006	Nick	First draft
0.2 DRAFT	01 July 2006	Nick	Add perl to chroot (pnmflip rotating in G2 fails otherwise)
1.0	15 July 2006	Nick	Final tidy up for first release

Planned Future Changes
Apply, test and document *mod_security* extension compatibility with Gallery2 Limit HTTP methods in httpd.conf to only those needed by Gallery2 Move the g2data out of htdocs aka "the gallery firewall" Add section on tight pf ruleset for HTTP in from all and SSH in from only one authorised workstation

Development & Testing Systems
1) Shuttle SB75G2, Intel Pentium 4 3GHz Processor, 1GB RAM, WD Raptor 72GB SATA HDD 2) P4 2.4GHz Processor, 512MB RAM, Seagate 400GB PATA HDD

Contents

1. Introduction 3
1.1. Overview 3
1.2. Limitations & Notes 3
1.3. BACKUP!! 3
2. Setup the OpenBSD Base System 4
3. Add your quirks (OPTIONAL) 5
3.1. Overview 5
3.2. Install the nano editor 5
3.3. Customise ksh prompt 5
3.4. Lockdown system 5
3.4.1. Add non-administrative user for SSH access 5
3.4.2. Harden SSH Configuration (/etc/ssh/sshd_config) 5
3.4.3. Remove unnecessary services 5
3.4.4. Restart System 5
4. Configure Apache 6
5. Install & Configure PostgreSQL 7
5.1. Install PostgreSQL from ports 7
5.2. Configure rc.conf.local 7
5.3. Configure rc.local 7
5.4. Configure rc.shutdown 7
5.5. Delete existing database and initialize new database 7
5.6. Setup PostgreSQL UNIX Socket 7
5.7. Start it all up 7
6. Install & Configure PHP 8
6.1. Build hardened PHP 5.1.4 8
6.2. Add PEAR package just built 8
6.3. Build and install hardened extensions 8
6.4. Modify Apache Configuration 8
6.5. Modify PHP Configuration 8
6.6. Test 8
7. Install & Configure Perl, NetPBM and jhead 9
7.1. Create needed directories 9
7.2. Copy existing Perl binaries into chroot 9
7.3. Copy the libraries Perl needs into the chroot 9
7.4. Install NetPBM 9
7.5. Catalog NetPBM package components and copy files in chroot 9
7.6. Install jhead 9
7.7. Catalog jhead package components and copy files in chroot 9
7.8. Catalog libraries for netpbm and jhead & copy them into chroot 9
7.9. Copy and update hints file 9
8. Install & Configure Gallery2 10
8.1. Create data directory and set permissions 10
8.2. Create the gallery2 user role and database 10
8.3. Extract the Gallery2 files 10
8.4. Installation 10
8.5. Enjoy! 10
9. Other Info 11
9.1. Backups 11

1. Introduction

1.1. Overview

This document describes the installation of the Gallery2 photo album software onto an OpenBSD system, utilizing the default security chroot of Apache. The installation uses PostgreSQL, as I prefer this over MySQL.

1.2. Limitations & Notes

Unfortunately, owing to a bug in PHP 5.0.5 (the version included with OpenBSD 3.9 release) there is a requirement to run OpenBSD-CURRENT to get everything working properly. This enables you to use the –CURRENT ports tree, which includes PHP 5.1.4 that does not suffer from the problem (see Gallery's documentation for more info on problems with PHP 5.0.5). There is of course the positive side effect that 5.1.4 has many bug fixes plus the port now also supports the hardening patches. Build it in as a flavour! =)

At the time of writing PHP 5.1.4 hadn't made it to –STABLE, but I would imagine that by OpenBSD 4.0 a working version of PHP will be available in the RELEASE stream.

We also have to copy Perl into the chroot. This is because utilities such as pnmflip (which is called to perform rotations of images by Gallery) are actually Perl scripts which invoke other binaries, hence the need for Perl to be copied in too.

Ports rather than packages are used to get things up and running. Throughout this document there are references to specific ports and their current version numbers. These of course may change as CURRENT is frequently updated. Be sure to check version numbers and revision levels of ports before attempting installation to avoid unnecessary errors! Also, in places there may be references to i386 architecture, so change this if needed.

I haven't bothered with the sudo route for installation etc. If you want to do that then simply modify the command lines as appropriate after setting up your sudoers config.

This procedure worked for me, it may not work for you. As always, your mileage may vary and I take no responsibility for using or failing to use properly, any of the steps noted within this guide. If you find any errors, or you think you have a more elegant way to achieve the results (very likely, all things considered) then please feel free to e-mail me and I will review submissions and consider them for inclusion into future versions of this document.

1.3. BACKUP!!

If you are not installing OpenBSD/Gallery onto a new, clean box then ensure that you have backups of the entire system first. Once you have that backup, check it to make sure that it works. As with any software installation, make sure that you can return your system to a known-good state before modifying ANYTHING!

2. Setup the OpenBSD Base System

For this portion of the install guide, refer to the official OpenBSD documentation and install FAQ.

This is going to be very much site dependent, as you'll have specific needs based on hardware available and amount of data to be stored. Below is an example partition layout used during the OpenBSD install process on the box where I had a 400GB disk to play with:

Partition	Mount Point	Size
a	/	256M
b	swap	512M
d	/tmp	512M
e	/home	128M
g	/usr	6G
h	/var	the rest

I deselect `gameXX.tgz` from the sets list, and answer NO to the "Use X.org" question at the end of the install.

I will assume that you will now update the system to –CURRENT by whatever method suits you (maybe you installed using a recent snapshot?) and that you fully understand the process of grabbing a tree from CVS including ports plus the building of a kernel plus userland. If you are not familiar with these concepts, ensure that you read the OpenBSD FAQ **first** before attempting any of the procedures in this guide.

3. Add your quirks (OPTIONAL)

3.1. Overview

Whenever I install a new OpenBSD system, I tend (as most people do) to apply some modifications to make things more comfortable. **This step is entirely optional** and simply covers a few bits that I like.

3.2. Install the nano editor

```
# cd /usr/ports/editors/nano
# make install
```

3.3. Customise ksh prompt

Append the following to **~/.profile**

```
export PS1='\u@\h:\w $ '
```

3.4. Lockdown system

3.4.1. Add non-administrative user for SSH access

Now add in a regular user account for yourself, follow the defaults. Afterwards, generate and add in the public key for this user, allowing you to PubKeyAuth when you SSH in (this is site dependent and therefore not covered here)

```
# adduser
# usermod -G wheel mynewuser
```

3.4.2. Harden SSH Configuration (/etc/ssh/sshd_config)

Uncomment and change `#Protocol 2,1` to `Protocol 2`
Uncomment and change `#PermitRootLogin yes` to `PermitRootLogin no`
Uncomment `MaxAuthTries 6`
Uncommment `PubkeyAuthentication yes`
Uncomment and change `#PasswordAuthentication yes` to `PasswordAuthentication no`
Uncomment `UsePrivilegeSeparation yes`
Uncomment and change Banner to one of your choice `Banner /etc/ssh/sshd.banner`
Now `touch /etc/ssh/sshd.banner` and enter the following into the new file:

```
**************************************************************

                  ===== ! WARNING ! =====

 This is a private computer system and all connected devices are
 the  property of   ABC123 Company Ltd.  Authorised  users only.
 Persons  attempting to  connect to  this system  without proper
 authorisation may be subject to civil and/or criminal penalties.
 Any  unauthorised  access (or  attempts  to gain  such)  may be
 prosecuted  to the  fullest extent  of the law. If  you are  not
 authorised to  use this  computer, disconnect  immediately.

**************************************************************
```

3.4.3. Remove unnecessary services

Edit **/etc/inetd.conf**
Comment out the following entries:

```
127.0.0.1:comsat
[::1]:comsat
daytime (both tcp and tcp6)
time (both tcp and tcp6)
```

3.4.4. Restart System

4. Configure Apache

In **/etc/rc.conf**, change `httpd_flags=NO` to `httpd_flags="", then:`

```
# mkdir -p /var/www/tmp
# chmod -R 777 /var/www/tmp
# mkdir -p /var/www/bin
# cp /bin/sh /var/www/bin/
```

Edit **/var/www/conf/httpd.conf** to disable banners, lock down directories:

```
<snip – this is only a portion of httpd.conf – only modify as necessary!>

ServerSignature Off
ServerTokens Prod
<Directory />
    AllowOverride None
    Options None
    Order allow,deny
    Allow from all
</Directory>
<Directory /htdocs/gallery2>
        AllowOverride Options FileInfo
</Directory>
```

5. Install & Configure PostgreSQL

5.1. Install PostgreSQL from ports

```
# cd /usr/ports/database/postgresql
# make install
# pkg_add /usr/ports/packages/i386/all/postgresql-server-8.1.4.tgz
```

5.2. Configure rc.conf.local

```
# touch /etc/rc.conf.local
# chown root:wheel rc.conf.local
# chmod 640 rc.conf.local
# echo 'postgresql=YES' > /etc/rc.conf.local
```

5.3. Configure rc.local

Add the following to **/etc/rc.local :**

```
if [ X"${postgresql}" = X"YES" -a -x /usr/local/bin/pg_ctl ]; then
 su -l _postgresql -c "/usr/local/bin/pg_ctl start -D /var/postgresql/data -l /var/postgresql/logfile -o \
      '-D /var/postgresql/data' -o '-k /var/www/tmp/'"
 ln -s /var/www/tmp/.s.PGSQL.5432 /tmp
 ln -s /var/www/tmp/.s.PGSQL.5432.lock /tmp
 echo -n ' postgresql'
fi
```

5.4. Configure rc.shutdown

Add the following to **/etc/rc.shutdown :**

```
if [ -f /var/postgresql/data/postmaster.pid ]; then
 su -l _postgresql -c "/usr/local/bin/pg_ctl stop -m fast -D /var/postgresql/data"
 rm -f /var/postgresql/data/postmaster.pid
fi
```

5.5. Delete existing database and initialize new database

```
# rm -rf /var/postgresql/data
# su _postgresql
$ initdb -D /var/postgresql/data
$ exit
```

5.6. Setup PostgreSQL UNIX Socket

```
# chmod a+rwt /var/www/tmp
# ln -s /var/www/tmp/.s.PGSQL.5432 /tmp
# ln -s /var/www/tmp/.s.PGSQL.5432.lock /tmp
```

5.7. Start it all up

```
# su -l _postgresql -c "/usr/local/bin/pg_ctl start -D /var/postgresql/data -l /var/postgresql/logfile -o \
      '-D /var/postgresql/data'"
```

6. Install & Configure PHP

6.1. Build hardened PHP 5.1.4

```
# cd /usr/ports/www/php5/core
# env FLAVOR="hardened" make install
# /usr/local/sbin/phpxs -s
# cp /usr/local/share/examples/php5/php.ini-recommended /var/www/conf/php.ini
```

6.2. Add PEAR package just built

```
# pkg_add /usr/ports/packages/i386/all/php5-pear-5.1.4p0.tgz
```

6.3. Build and install hardened extensions

```
# cd /usr/ports/www/php5/extensions/
# env FLAVOR="no_x11 hardened no_bz2 no_curl no_dba no_dbase no_filepro no_gd no_gmp no_imap \
      no_ldap no_mbstring no_mhash no_mysql no_mysqli no_ncurses no_odbc no_shmop no_soap \
      no_snmp no_sqlite no_sybase_ct no_xmlrpc no_xsl" make install
# pkg_add /usr/ports/packages/i386/all/php5-mcrypt-5.1.4-hardened.tgz
# pkg_add /usr/ports/packages/i386/all/php5-pgsql-5.1.4-hardened.tgz
# /usr/local/sbin/phpxs -a mcrypt
# /usr/local/sbin/phpxs -a pgsql
```

6.4. Modify Apache Configuration

Edit **/var/www/conf/httpd.conf**

Uncomment the following line :

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

Uncomment the following line and add .html :

```
AddType application/x-httpd-php .php .html
```

Add index.php to DirectoryIndex :

```
DirectoryIndex index.html index.php
```

6.5. Modify PHP Configuration

Modify **/var/www/conf/php.ini**

Change `expose_php=On` to:

```
expose_php=Off
```

Then change the max sizes:

```
post_max_size 10M
upload_max_filesize 10M
```

6.6. Test

```
echo '<?php phpinfo(); ?>' > /var/www/htdocs/phptest.html
```

Now browse to server's address e.g. http://192.168.0.200/phptest.html – You should see the php info page. If not, go back and see where things have gone wrong. If it does show up, delete it now as it could leak important info about your setup once you go live.

7. Install & Configure Perl, NetPBM and jhead

7.1. Create needed directories

```
# mkdir -p /var/www/usr/local/bin/
# mkdir -p /var/www/usr/local/include/
# mkdir -p /var/www/usr/local/lib/
# mkdir -p /var/www/usr/local/share/netpbm/
# mkdir -p /var/www/usr/local/share/doc/jhead/
# mkdir -p /var/www/usr/local/libdata
# mkdir -p /var/www/usr/bin/
# mkdir -p /var/www/usr/lib/
# mkdir -p /var/www/usr/libexec/
# mkdir -p /var/www/usr/libdata/
```

7.2. Copy existing Perl binaries into chroot

```
# cp /usr/bin/perl /var/www/usr/bin/
# cp /usr/bin/perl5.* /var/www/usr/bin/
```

7.3. Copy the libraries Perl needs into the chroot

```
# cp -R /usr/local/libdata/perl5 /var/www/usr/local/libdata/
# cp -R /usr/libdata/perl5 /var/www/usr/libdata/
# for i in /var/www/usr/bin/*; do ldd $i; done | grep 'lib' | cut -c 41-80 | sort -n | uniq > /tmp/libs
# for i in `cat /tmp/libs`; do cp $i /var/www$i; done
```

7.4. Install NetPBM

```
# cd /usr/ports/graphics/netpbm/
# make install
```

7.5. Catalog NetPBM package components and copy files in chroot

```
# pkg_info -K -L netpbm | grep -e '^@file' -e '^@lib'  | cut -d" " -f2 > /tmp/files
# for i in `cat /tmp/files`; do cp $i /var/www$i; done
```

7.6. Install jhead

```
# cd /usr/ports/graphics/jhead/
# make install
```

7.7. Catalog jhead package components and copy files in chroot

```
# pkg_info -K -L jhead | grep -e '^@file' -e '^@lib'  | cut -d" " -f2 > /tmp/files
# for i in `cat /tmp/files`; do cp $i /var/www$i; done
```

7.8. Catalog libraries for netpbm and jhead & copy them into chroot

```
# for i in /var/www/usr/local/bin/*; do ldd $i; done | grep 'lib' | cut -c 41-80 | sort -n | uniq > /tmp/libs
# for i in `cat /tmp/libs`; do cp $i /var/www$i; done
```

7.9. Copy and update hints file

```
# mkdir -p /var/www/var/run
# mkdir -p /var/www/sbin/
# cp /sbin/ldconfig /var/www/sbin/ldconfig
# cp /var/run/ld.so.hints /var/www/var/run/ld.so.hints
# cd /var/www
# chroot /var/www sh
# /sbin/ldconfig -m /usr/local/lib
# /sbin/ldconfig -m /usr/lib
# exit
```

8. Install & Configure Gallery2

8.1. Create data directory and set permissions

```
# cd /var/www/htdocs/
# mkdir g2data
# chown www:www g2data
# chmod 700 g2data
```

8.2. Create the gallery2 user role and database

```
# su _postgresql
$ createuser -P g2db_user
      (enter strong password)
$ createdb gallery2 -E UNICODE -O g2db_user
$ exit
```

8.3. Extract the Gallery2 files

```
# tar xfz (downloaddirectory)/gallery-2.1.1a-typical.tar.gz (or whichever package you selected)
```

8.4. Installation

Open web browser and go to http://yourservername/gallery2. At this point you should now follow the normal Gallery2 installation instructions which can be found at http://gallery.menalto.com. However, **it is STRONGLY recommended that you use DB Locking** - This is configurable under Site Admin.

8.5. Enjoy!

Hope you have fun with your new Gallery installation! Read the next section for some additional admin tips.

9. Other Info

9.1. Backups

Making backups is extremely important, especially if this system is going to be your only store of your precious digital photos. Below is a method I use for taking regular backups of the PostgreSQL database, Gallery2 directory and G2Data:

First of all I create a place to keep the backups. A main directory containing two subdirectories, with one being writable by the _postgresql superuser. All others writable by root only.

```
# mkdir /var/mybackups
# mkdir /var/mybackups/pgsql
# mkdir /var/mybackups/g2files
# chown _postgresql /var/mybackups/pgsql
```

I then create a new script called ~/dobackup.sh and chmod 700 the resulting file. Contents are below:

```
#!/bin/sh
cd /var/mybackups/pgsql
sudo -u _postgresql pg_dump g2data > ./g2data_backup.pgsql
cd /var/mybackups/g2files
tar cvfz website_backup.tar.gz /var/www/htdocs/
```

You can then add this script in as a cron job to run at appropriate intervals. You'd probably want to extend this script to copy the resulting backup files onto a CDR, across the network or onto a tape. Store any media offsite!

***** END OF DOCUMENT *****