Installing a secure Apache webserver.
by neuro: neuro@seclab.jp

Concept exposed in that paper were tested under GNU Linux Slackware 8.1, running a 2.4.18 kernel with GRsecurity patch. Gcc version is 2.95.3 and glibc is 2.2.5. Apache version is 1.3.27, with PHP version 4.2.1 and MySQL 3.23.53. It should run under any recent GNU Linux system with recent 1.3.x Apache version. This should not work under Red Hat as it uses a special gcc version, the 2.96.

The purpose of this paper is to get a well secured and configured httpd, running PHP and MySQL. For that reason, Apache, PHP and MySQL database will be installed and compiled toegether.

Part 1: creating some users.

At first, we gonna create some new users and groups usefull for what we wanna do: every single piece of the server will be launched under a particular unpriviledge user.

```
(root@sai:~) groupadd -g 27 mysql
(root@sai:~) groupadd -g 28 www
(root@sai:~) useradd -u 27 -g 27 -s /dev/null mysql
(root@sai:~) useradd -u 28 -g 28 -s /dev/null www
```

Part 2: compiling the stuff all toegether.

We shall now compile and install MySQL database. We assume that sources are located in /usr/src. I have choosen MySQL because it's the most common free database on the internet. For other *SQL databases, compilation options may be a bit different.

```
(root@sai:/usr/src) tar xzf mysql-3.23.53-pc-linux-gnu-i686.tar.gz
(root@sai:/usr/src) mv mysql-3.23.53-pc-linux-gnu-i686 mysql
(root@sai:/usr/src) cd mysql
(root@sai:/usr/src/mysql) ./configure --prefix=/usr/local/mysql \
                    --with-low-memory --with-mysqld-user=mysql
(root@sai:/usr/src/mysql)make && make install
```

Nothing really hard there. What we do is telling MySQL to run under an unpriviledged user mysql that we created sooner. All the libs and binaries are located into / usr/local/mysql. This is good because we shall move that soon. The --with-low-memory option avoids a local Denial of Service bug, telling mySQL that the system has a low memory so it musn't use cache and other memory greedy stuffs. Otherway, you should have a local Denial Of Service, MySQL quickly running out of memory.

Once MySQL compiled, we can compile PHP. MySQL server configuration will come later.

```
(root@sai:/usr/src) tar xzf /usr/src/apache_1.3.27.tar.gz
(root@sai:/usr/src) ln -s apache_1.3.27 /usr/local/apache
(root@sai:/usr/src) cd apache_1.3.27
(root@sai:/usr/src/apache_1.3.27) ./configure
```

When compiling, PHP will create a special module, libphp4.a, and will put it in /usr/local/apache. As we need to compile Apache with that lib statically linked in the binary, we do that simlink just to have PHP to compile properly. We shall remove it just after as we install Apache webserver at the same location. We tun the configure script so php will find every information it needs to install properly.

```
(root@sai:/usr/src/apache_1.3.27) cd ..
(root@sai:/usr/src) tar xzf php-4.2.1.tar.gz
(root@sai:/usr/src) cd php-4.2.1
(root@sai:/usr/src/php-4.2.1) ./configure --prefix=/usr/local/php \
                      --with-apache=/usr/local/apache \
                      --with-mysql=/usr/local/mysql
(root@sai:/usr/src/php-4.2.1) make && make install
```

We have installed PHP in /usr/local. We tell PHP libraries where Apache and MySQL are located. That's the reason why we ran just a configure for Apache sooner: we needed PHP to discover where it was.

```
(root@sai:/usr/src/php-4.2.1) cd ..
(root@sai:/usr/src) rm /usr/local/apache
(root@sai:/usr/src) cd apache_1.3.27
(root@sai:/usr/src/apache_1.3.27) ./configure \
                        --prefix=/usr/local/apache \
                        --enable-module=all \
                        --activate-module=src/modules/php4/libphp4.a
(root@sai:/usr/src/apache_1.3.27) make && make install
(root@sai:/usr/src/apache_1.3.27) cd bin
(root@sai:/usr/src/apache_1.3.27) ln -s httpd /usr/sbin/httpd
```

We install all httpd stuff in /usr/local/apache. That will be very usefull when we will chroot the whole shit. We enable all built-in modules, and activate specifically the php4 library we built sooner. That way, PHP will be statically linked with Apache binary.

Part 3: some configuration.

We first gonna deal with MySQL stuff. We won't create adatabases or so, that's not the purpose of that paper. We'll just ensure that no one will come and take over the root account.

```
(root@sai:/usr/src) cd /usr/local/mysql
(root@sai:/usr/local/mysql) chown -R mysql.mysql var
(root@sai:/usr/local/mysql) cd bin
(root@sai:/usr/local/mysql/bin) ./mysql_install_db
(root@sai:/usr/local/mysql/bin) ./safe_mysqld&
(root@sai:/usr/local/mysql/bin) ./mysql_admin -u root myelitepassword
```

Then to start the MySQL daemon, we just need to run the secure MySQL script:

```
(root@sai:/usr/local/mysql/bin) ./safe_mysqld&
```

That's it, now, your mysql database is running perfectly, you can start to work.

Looks good now. We can now customize the httpd.conf file. That's now my main purpose there so I won't detail it, but I want to give you some tips anyway, so let's have a look at it.

The file is: /var/chrooted-httpd/usr/local/apache/conf/httpd.con

```
----------------------------------------------------------------------
# the server won't be launched by the inetd daemon
ServerType standalone

# directory where apache is located
ServerRoot "/usr/local/apache"

# if that file exists, no other httpd will be started.
PidFile /usr/local/apache/logs/httpd.pid

# the scoreborad file
ScoreBoardFile /usr/local/apache/logs/httpd.scoreboard

# everything concerning incoming connections.
Timeout 300
KeepAlive On
MaxKeepAliveRequests 100
KeepAliveTimeout 15

# number of servers to be launched at starting, and maximum servers
# launched at the same time
MinSpareServers 5
MaxSpareServers 10
StartServers 5

# maximum number of clients connecting to a server.
MaxClients 50
MaxRequestsPerChild 0

# Some system related stuffs
Port 80
User www
Group www
ServerAdmin root@seclab.jp

# the main webpage directory
# and some proprieties.
<Directory "/home/www/">
    Options Indexes FollowSymLinks MultiViews
    AllowOverride None
    Order allow,deny
    Allow from all
</Directory>

# Normal users w3 directories.
<IfModule mod_userdir.c>
    UserDir www
</IfModule>

# Normal users directories proprieties.
```

```
<Directory /home/*/www>
    AllowOverride FileInfo AuthConfig Limit
    Options MultiViews Indexes SymLinksIfOwnerMatch IncludesNoExec
    <Limit GET POST OPTIONS PROPFIND>
        Order allow,deny
        Allow from all
    </Limit>
    <LimitExcept GET POST OPTIONS PROPFIND>
        Order deny,allow
        Deny from all
    </LimitExcept>
</Directory>
<IfModule mod_dir.c>
    DirectoryIndex index.html
</IfModule>
AccessFileName .htaccess
<Files ~ "^\.ht">
    Order allow,deny
    Deny from all
    Satisfy All
</Files>

# UseCanonicalName:  (new for 1.3)  With this setting turned on, whenever
# Apache needs to construct a self-referencing URL (a URL that refers
back
# to the server the response is coming from) it will use ServerName and
# Port to form a "canonical" name.  With this setting off, Apache will
# use the hostname:port that the client supplied, when possible.  This
# also affects SERVER_NAME and SERVER_PORT in CGI scripts.

UseCanonicalName Off

# Mime types related modules.
<IfModule mod_mime.c>
    TypesConfig /usr/local/apache/conf/mime.types
</IfModule>
DefaultType text/plain
<IfModule mod_mime_magic.c>
    MIMEMagicFile /usr/local/apache/conf/magic
</IfModule>

# Looking for hostname?
HostnameLookups On

# Logging facilities.
ErrorLog /usr/local/apache/logs/error_log
LogLevel warn
LogFormat "%h %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-Agent}i\""
combined
LogFormat "%h %l %u %t \"%r\" %>s %b" common
LogFormat "%{Referer}i -> %U" referer
LogFormat "%{User-agent}i" agent
CustomLog /usr/local/apache/logs/access_log common

# The server version won't appear on 404 pages an other GET
#requests. Usefull for hiding from scriptkidies.
ServerSignature Off
```

```
# Usefull aliases for apache stuffs.
<IfModule mod_alias.c>
    Alias /icons/ "/usr/local/apache/icons/"
    <Directory "/usr/local/apache/icons">
        Options Indexes MultiViews
        AllowOverride None
        Order allow,deny
        Allow from all
    </Directory>
    Alias /manual/ "/usr/local/apache/htdocs/manual/"
    <Directory "/usr/local/apache/htdocs/manual">
        Options Indexes FollowSymlinks MultiViews
        AllowOverride None
        Order allow,deny
        Allow from all
    </Directory>
    ScriptAlias /cgi-bin/ "/usr/local/apache/cgi-bin/"
    <Directory "/usr/local/apache/cgi-bin">
        AllowOverride None
        Options None
        Order allow,deny
        Allow from all
    </Directory>
</IfModule>

# File types to icons. While brewsing a directory, various icons will
# apear beside files, showing their type.
<IfModule mod_autoindex.c>
    IndexOptions FancyIndexing
    AddIconByEncoding (CMP,/icons/compressed.gif) x-compress x-gzip
    AddIconByType (TXT,/icons/text.gif) text/*
    AddIconByType (IMG,/icons/image2.gif) image/*
    AddIconByType (SND,/icons/sound2.gif) audio/*
    AddIconByType (VID,/icons/movie.gif) video/*
    AddIcon /icons/binary.gif .bin .exe
    AddIcon /icons/binhex.gif .hqx
    AddIcon /icons/tar.gif .tar
    AddIcon /icons/world2.gif .wrl .wrl.gz .vrml .vrm .iv
    AddIcon /icons/compressed.gif .Z .z .tgz .gz .zip
    AddIcon /icons/a.gif .ps .ai .eps
    AddIcon /icons/layout.gif .html .shtml .htm .pdf
    AddIcon /icons/text.gif .txt
    AddIcon /icons/c.gif .c
    AddIcon /icons/p.gif .pl .py
    AddIcon /icons/f.gif .for
    AddIcon /icons/dvi.gif .dvi
    AddIcon /icons/uuencoded.gif .uu
    AddIcon /icons/script.gif .conf .sh .shar .csh .ksh .tcl
    AddIcon /icons/tex.gif .tex
    AddIcon /icons/bomb.gif core
    AddIcon /icons/back.gif ..
    AddIcon /icons/hand.right.gif README
    AddIcon /icons/folder.gif ^^DIRECTORY^^
    AddIcon /icons/blank.gif ^^BLANKICON^^
    DefaultIcon /icons/unknown.gif
    AddDescription "GZIP compressed document" .gz
```

```
    AddDescription "tar archive" .tar
    AddDescription "GZIP compressed tar archive" .tgz
    ReadmeName README
    HeaderName HEADER
    IndexIgnore .??* *~ *# HEADER* README* RCS CVS *,v *,t
</IfModule>
<IfModule mod_mime.c>

# Uncompressing zipped files on the fly.
AddEncoding x-compress Z
AddEncoding x-gzip gz tgz

# AddLanguage allows you to specify the language of a document. You
# can then use content negotiation to give a browser a file in a language
# it can understand.
# Danish (da) - Dutch (nl) - English (en) - Estonian (ee)
# French (fr) - German (de) - Greek-Modern (el)
# Italian (it) - Korean (kr) - Norwegian (no) - Norwegian Nynorsk (nn)
# Portugese (pt) - Luxembourgeois* (ltz)
# Spanish (es) - Swedish (sv) - Catalan (ca) - Czech(cz)
# Polish (pl) - Brazilian Portuguese (pt-br) - Japanese (ja)
# Russian (ru)
AddLanguage da .dk
AddLanguage nl .nl
AddLanguage en .en
AddLanguage et .ee
AddLanguage fr .fr
AddLanguage de .de
AddLanguage el .el
AddLanguage he .he
AddCharset ISO-8859-8 .iso8859-8
AddLanguage it .it
AddLanguage ja .ja
AddCharset ISO-2022-JP .jis
AddLanguage kr .kr
AddCharset ISO-2022-KR .iso-kr
AddLanguage nn .nn
AddLanguage no .no
AddLanguage pl .po
AddCharset ISO-8859-2 .iso-pl
AddLanguage pt .pt
AddLanguage pt-br .pt-br
AddLanguage ltz .lu
AddLanguage ca .ca
AddLanguage es .es
AddLanguage sv .sv
AddLanguage cz .cz
AddLanguage ru .ru
AddLanguage zh-tw .tw
AddLanguage tw .tw
AddCharset Big5         .Big5    .big5
AddCharset WINDOWS-1251 .cp-1251
AddCharset CP866        .cp866
AddCharset ISO-8859-5   .iso-ru
AddCharset KOI8-R       .koi8-r
AddCharset UCS-2        .ucs2
AddCharset UCS-4        .ucs4
```

```
AddCharset UTF-8        .utf8

# LanguagePriority allows you to give precedence to some languages
# in case of a tie during content negotiation. Just list the languages
# in decreasing order of preference.
<IfModule mod_negotiation.c>
        LanguagePriority fr en
</IfModule>

# AddType allows you to tweak mime.types without actually editing it, or
to
# make certain files to be certain types.
AddType application/x-tar .tgz

# The followin lines will enable php scripts using. Now, apach is
going to recognise .php files as php interpreted files.
<IfModule mod_php4.c>
   AddType application/x-httpd-php .php4 .php3 .phtml .php
   AddType application/x-httpd-php-source .phps
   </IfModule>

</IfModule>


# Customize behaviour based on the browser
<IfModule mod_setenvif.c>

# The following directives modify normal HTTP response behavior.
# The first directive disables keepalive for Netscape 2.x and browsers
that
# spoof it. There are known problems with these browser implementations.
# The second directive is for Microsoft Internet Explorer 4.0b2
# which has a broken HTTP/1.1 implementation and does not properly
# support keepalive when it is used on 301 or 302 (redirect) responses.

BrowserMatch "Mozilla/2" nokeepalive
BrowserMatch "MSIE 4\.0b2;" nokeepalive downgrade-1.0 force-response-1.0

# The following directive disables HTTP/1.1 responses to browsers which
# are in violation of the HTTP/1.0 spec by not being able to grok a
# basic 1.1 response.
BrowserMatch "RealPlayer 4\.0" force-response-1.0
BrowserMatch "Java/1\.0" force-response-1.0
BrowserMatch "JDK/1\.0" force-response-1.0
</IfModule>

# VirtualHost: If you want to maintain multiple domains/hostnames on your
# machine you can setup VirtualHost containers for them. Most
configurations
# use only name-based virtual hosts so the server doesn't need to worry
about
# IP addresses. This is indicated by the asterisks in the directives
below.

# Use name-based virtual hosting.
NameVirtualHost *
```

```
<VirtualHost *>
    ServerAdmin neuro@seclab.jp
    DocumentRoot /home/neuro/seclab
    ServerName seclab.jp
</VirtualHost>

<VirtualHost *>
    ServerAdmin neuro@seclab.jp
    DocumentRoot /home/neuro/seclab
    ServerName www.seclab.jp
</VirtualHost>
```

Part 3: chrooting the whole stuff.

Apache prior to 2.0 was not designed to be chrooted, and that operation can reveal hazardous. We'll have to use the chroot(1) provided by the system. First, we have to recreate the chrooted environment.

```
(root@sai:/) mkdir /var/chrooted-httpd
(root@sai:/) cd /var/chrooted-httpd/
(root@sai:/var/chrooted-httpd/) mkdir -p etc usr/lib usr/local/ lib
var/run var/pid dev
(root@sai:/var/chrooted-httpd/) egrep "(^root:|^www:|^mysql:) \
                              /etc/group > etc/group
(root@sai:/var/chrooted-httpd/) egrep "^root:|^www:|^mysql:) \
                              /etc/passwd > etc/passwd
(root@sai:/var/chrooted-httpd/) mknod dev/null -c 1 6
(root@sai:/var/chrooted-httpd/) cd lib
(root@sai:/var/chrooted-httpd/lib) cp -R /usr/local/mysql ../usr/local
(root@sai:/var/chrooted-httpd/lib) cp -R /usr/local/apache ../usr/local
(root@sai:/var/chrooted-httpd/lib) cp -R /usr/local/apache ../usr/local
(root@sai:/var/chrooted-httpd/lib) cp /lib/libm.so.6 .
(root@sai:/var/chrooted-httpd/lib) cp /lib/libcrypt.so.1 .
(root@sai:/var/chrooted-httpd/lib) cp /lib/libresolv.so.2 .
(root@sai:/var/chrooted-httpd/lib) cp /lib/libdl.so.2 .
(root@sai:/var/chrooted-httpd/lib) cp /lib/libnsl.so.1 .
(root@sai:/var/chrooted-httpd/lib) cp /lib/libdb.so.3 .
(root@sai:/var/chrooted-httpd/lib) cp /lib/libc.so.6 .
(root@sai:/var/chrooted-httpd/lib) cp /lib/ld-linux.so.2 .
(root@sai:/var/chrooted-httpd/lib) cp /lib/libpthread.so.0 .
(root@sai:/var/chrooted-httpd/lib) cp /lib/libm.so.6 .
(root@sai:/var/chrooted-httpd/lib) cp /lib/libncurses.so.5 .
(root@sai:/var/chrooted-httpd/lib) cd ../usr/lib
(root@sai:/var/chrooted-httpd/usr/lib) cp /usr/lib/libz.so.1 .
(root@sai:/var/chrooted-httpd/usr/lib) cp /usr/lib/libstdc++-libc6.2-
2.so.3 .
```

Now, let's start the whole stuff in chroot:

```
(root@sai:~) chroot /var/chrooted-httpd \
           /var/chrooted-httpd/usr/local/mysql/safe_mysqld
(root@sai:~) chroot /var/chrooted-httpd \
           /var/chrooted-httpd/usr/sbin/httpd
```

Now you should have a running and working chrooted httpd. As I said sooner, this is a bit hazardous and can not work under some

environments for obscure reasons.

Some links:
Apache: http://www.apache.org
PHP: http://www.php.net
MySQL: http://www.mysql.com
Slackware: http://www.slackware.com
Fast slackware ftp: ftp://ftp.lip6.fr/pub/linux/distributions/slackware.

Greetz: Turlut, kad, Spacewalker, banux and veins.