

## Building an Apache webserver with PHP and mod\_ssl on OS X 10.0.3

disclaimer:

This how-to describes how I have done it on three OS X boxes. I can't however give any guarantee that it will work on your OS X box.

English isn't my mother tongue, so if the english isn't grammatically correct, so be it:) I'm absolutely not an unix wizzkidd, so it could be that certain steps could be much easier or are not necessary. Please be so kind to give me serious feedback.

I've done everything as a root user.

% is the symbol for the commandline prompt.

I've used mainly the bash shell, but most of this should work with tcsh shell as well, be alert for this with configure apache.

I also assume that you have already installed openssh 2.9p1 if not see [www.stepwise.com](http://www.stepwise.com) for an excellent how to.

First for making things easier create a /usr/local/src directory.

```
% mkdir /usr/local/src
```

Download the needed tarballs. You'll need to download:

OpenSSL-0.9.5a-3.1.tar.gz

apache\_1.3.19.tar.gz

mod\_ssl-2.8.3-1.3.19.tar.gz

php-4.0.5.tar.gz

gdbm-1.8.0.tar.gz

optional:

mysql-3.23.36.tar.gz

the latest release is 3.23.38 but I have 3.23.36 installed and it works.....never change a winning team:) Installation how-to's for mySQL are all over the place, so won't be covered here.

Move all the tarballs to your freshly created /usr/local/src directory.

And untar;

```
% gnutar -xzvf name.of.tarball.tar.gz
```

Do this for all the tarballs.

Now leave your commandline and open netinfo manager.

Enter your administrator password and go to: /users

duplicate the unknown user, and change in the property window the name to bin.  
save and close netinfo manager, your done here.  
You could also use niutil, but didn't work for me....

Go back to your terminal which is waiting for you in /usr/local/src (isn't it?)

```
% cd gdbm-1.8.0 (please use your tab key for completion:)  
% cp /usr/libexec/config* .  
% ./configure  
% make  
% make install  
% ln -s /usr/local/lib/libgdbm.a /usr/local/lib/libdbm.a
```

Now you have installed GNU dbm, a database utility which mod\_ssl needs for compiling.

```
% cd ..  
% cd OpenSSL-3-1/openssl  
% ./config  
% make (NOT Install!!!!!!)  
% cd /usr/local/src
```

Now we are going to pre-configure apache, so apache knows all his needed paths

```
% cd apache_1.3.19  
% ./configure  
% cd ..
```

Now we will configure, compile and install PHP, I'm showing the options I used, feel free to add any other options....some however are really needed.  
First we need a fix for php4.0.5 as described at stepwise.

```
% wget http://graphics.stepwise.com/Articles/Workbench/php-4.0.5-genif.sh  
% mv php-4.0.5-genif.sh php-4.0.5/build/genif.sh  
% cd php-4.0.5  
% ./configure \  
    --with-xml \  
    --with-mysql=/path/to/mysql \  
    --with-apache=./apache_1.3.19 \  
    --with-zlib \  
    --enable-track-vars  
% make  
% make install
```

```
% cp php.ini-dist /usr/local/lib/php.ini
% cd ..
```

Now we have to configure mod\_SSL and patch the apache source

```
% cd mod_ssl-2.8.3-1.3.19
% ./configure --with-apache=../apache_1.3.19
% cd..
```

Now we are ready to build apache (and are we having fun!?)

Important: I've used the BASH shell

```
% cd apache_1.3.19
% SSL_BASE=../OpenSSL-3-1/openssl \
  ./configure \
  --enable-module=ssl \
  --activate-module=src/modules/php4/libphp4.a \
  --enable-module=php4 \
  --enable-shared=ssl
% make
```

For TCSH shell replace SSL\_BASE with:

```
% setenv SSL_BASE=path to Openssl/openssl \
  ./configure \
  etc....
```

If everything went ok, this is what you should see at the end of the make step

```
+-----+
| Before you install the package you now should prepare the SSL |
| certificate system by running the 'make certificate' command. |
| For different situations the following variants are provided: |
| |
| % make certificate TYPE=dummy (dummy self-signed Snake Oil cert)|
| % make certificate TYPE=test (test cert signed by Snake Oil CA) |
| % make certificate TYPE=custom (custom cert signed by own CA) |
| % make certificate TYPE=existing (existing cert) |
| CRT=/path/to/your.crt [KEY=/path/to/your.key] |
| Use TYPE=dummy when you're a vendor package maintainer, |
| the TYPE=test when you're an admin but want to do tests only, |
| the TYPE=custom when you're an admin willing to run a real server|
| and TYPE=existing when you're an admin who upgrades a server. |
```

```
(The default is TYPE=test)
```

```
Additionally add ALGO=RSA (default) or ALGO=DSA to select  
the signature algorithm used for the generated certificate.
```

```
Use 'make certificate VIEW=1' to display the generated data.
```

```
Thanks for using Apache & mod_ssl. Ralf S. Engelschall  
rse@engelschall.com  
www.engelschall.com
```

(at least, I hope so:)

### `% make certificate`

This could be a little bit tricky....

I choose the RSA certificate, You will need to answer some questions.

The one really really important is the common name!

You have to enter your "FQDN" or for normal people Your Fully Qualified Domainname leading to the IP of your Server.

I do have a steady IP, but don't have a DNS entry to my IP. So I registred a domain for free at [www.dyndns.org](http://www.dyndns.org) and entered as FQDN my dyndns.org domain.

You should also try to enter your IP, but localhostname didn't work for me.....

The "commonname" (somewhat misleading in my eyes) should also be your apache ServerName, but we will cover that later.

### `% make install`

If you follow the install closely, you'll notice that all your existing apache files are preserved, including your original httpd.conf

This conf file isn't however configured for mod\_ssl and your newly PHP.

But you can find a patched httpd.conf in:

```
/usr/local/src/apache_1.3.19/conf/httpd.conf-dist
```

This httpd.conf doesn't reflect the settings needed for your OS X box.

So I did the following:

```
% cd /etc/httpd/
```

```
% mv httpd.conf httpd.conf_old
```

```
% cp /usr/local/src/apache_1.3.19/conf/httpd.conf-dist httpd.conf
```

After this I opened another terminal window and opened httpd.conf\_old with vi

```
% vi httpd.conf_old
```

In our original working terminal I opened httpd.conf

```
% vi httpd.conf
```

And compared the both httpd.conf's line for line and altered the values in our new httpd.conf so that it will reflect the proper settings for our OS X box.

Also you will need to uncomment the php4 lines.....

For making things easier for you, I will write down my own httpd.conf at the end of this how-to....feel free to use this one, but do not forget to alter some variables...for instance your ServerName!!!!

IMPORTANT: use the same FQDN as you entered in making your certificate.

Otherwise you never get your secure webserver to work!!!!!!

close and save your httpd.conf.

now some testing:

```
% apachectl stop (if running off course;)
```

```
% apachectl configtest
```

Hopefully you'll get an: syntax OK

Otherwise check your httpd.conf

```
% apachectl start
```

Test with your browser that apache is really running and serving.

you might also want to check that your PHP is working with for instance a page containing: <? phpinfo(); ?>

If this worked fine, we are now ready to test our secure apache.

```
% apachectl stop
```

```
% apachectl startssl
```

If you entered a passphrase during the make certificate, you'll be prompted for your passphrase.

Go to your browser and enter as URL: https://your.domain.co

If it worked you should see a notice that you entered a secure site.

if it doesn't work, you could check /var/log/httpd/error\_log or /var/log/httpd/ssl\_engine\_log.

If does all work, go get yourself a cool glass of beer! You earned it (if you have the legal age off course:)

You have now a secure webserver, however, you only use a test-certificate. for the real world use you should buy a real certificate at ie. Verisign or similar company.

Big thanks to :                Scott Anguish (www.stepwise.com)  
   Morten Ronseth  
   the people at www.devshed.com  
   the people at the macOSX admin list

Yours sincerely,

Merlijn Tishauser  
gwyrddin@mac.com

As promised, my httpd.conf:  
as example only!!!!!!  
Feel free to copy it, but make sure you check it!!!!  
I deleted my servername and email adress etc.

```
##  
## httpd.conf -- Apache HTTP server configuration file  
##  
  
#  
# Based upon the NCSA server configuration files originally by Rob McCool.  
#  
# This is the main Apache server configuration file. It contains the  
# configuration directives that give the server its instructions.  
# See <URL:http://www.apache.org/docs/> for detailed information about  
# the directives.  
#  
# Do NOT simply read the instructions in here without understanding  
# what they do. They're here only as hints or reminders. If you are unsure  
# consult the online docs. You have been warned.  
#  
# After this file is processed, the server will look for and process  
# /usr/conf/srm.conf and then /usr/conf/access.conf
```

```
# unless you have overridden these with ResourceConfig and/or
# AccessConfig directives here.
#
# The configuration directives are grouped into three basic sections:
# 1. Directives that control the operation of the Apache server process as a
#    whole (the 'global environment').
# 2. Directives that define the parameters of the 'main' or 'default' server,
#    which responds to requests that aren't handled by a virtual host.
#    These directives also provide default values for the settings
#    of all virtual hosts.
# 3. Settings for virtual hosts, which allow Web requests to be sent to
#    different IP addresses or hostnames and have them handled by the
#    same Apache server process.
#
# Configuration and logfile names: If the filenames you specify for many
# of the server's control files begin with "/" (or "drive:/" for Win32), the
# server will use that explicit path. If the filenames do *not* begin
# with "/", the value of ServerRoot is prepended -- so "logs/foo.log"
# with ServerRoot set to "/usr/local/apache" will be interpreted by the
# server as "/usr/local/apache/logs/foo.log".
#
```

### ### Section 1: Global Environment

```
#
# The directives in this section affect the overall operation of Apache,
# such as the number of concurrent requests it can handle or where it
# can find its configuration files.
#
```

```
#
# ServerType is either inetd, or standalone. Inetd mode is only supported on
# Unix platforms.
```

```
#
ServerType standalone
```

```
#
# ServerRoot: The top of the directory tree under which the server's
# configuration, error, and log files are kept.
#
# NOTE! If you intend to place this on an NFS (or otherwise network)
# mounted filesystem then please read the LockFile documentation
# (available at <URL:http://www.apache.org/docs/mod/core.html#lockfile>);
# you will save yourself a lot of trouble.
```

```
#
ServerRoot "/usr"

#
# The LockFile directive sets the path to the lockfile used when Apache
# is compiled with either USE_FCNTL_SERIALIZED_ACCEPT or
# USE_FLOCK_SERIALIZED_ACCEPT. This directive should normally be left at
# its default value. The main reason for changing it is if the logs
# directory is NFS mounted, since the lockfile MUST BE STORED ON A LOCAL
# DISK. The PID of the main server process is automatically appended to
# the filename.
#
#LockFile /var/run/httpd.lock

#
# PidFile: The file in which the server should record its process
# identification number when it starts.
#
PidFile "/private/var/run/httpd.pid"

#
# ScoreBoardFile: File used to store internal server process information.
# Not all architectures require this. But if yours does (you'll know because
# this file will be created when you run Apache) then you *must* ensure that
# no two invocations of Apache share the same scoreboard file.
#
ScoreBoardFile "/private/var/run/httpd.scoreboard"

#
# In the standard configuration, the server will process httpd.conf (this
# file, specified by the -f command line option), srm.conf, and access.conf
# in that order. The latter two files are now distributed empty, as it is
# recommended that all directives be kept in a single file for simplicity.
# The commented-out values below are the built-in defaults. You can have the
# server ignore these files altogether by using "/dev/null" (for Unix) or
# "nul" (for Win32) for the arguments to the directives.
#
#ResourceConfig conf/srm.conf
#AccessConfig conf/access.conf

#
# Timeout: The number of seconds before receives and sends time out.
#
```

Timeout 300

#

# KeepAlive: Whether or not to allow persistent connections (more than  
# one request per connection). Set to "Off" to deactivate.

#

KeepAlive On

#

# MaxKeepAliveRequests: The maximum number of requests to allow  
# during a persistent connection. Set to 0 to allow an unlimited amount.  
# We recommend you leave this number high, for maximum performance.

#

MaxKeepAliveRequests 100

#

# KeepAliveTimeout: Number of seconds to wait for the next request from the  
# same client on the same connection.

#

KeepAliveTimeout 15

#

# Server-pool size regulation. Rather than making you guess how many  
# server processes you need, Apache dynamically adapts to the load it  
# sees --- that is, it tries to maintain enough server processes to  
# handle the current load, plus a few spare servers to handle transient  
# load spikes (e.g., multiple simultaneous requests from a single  
# Netscape browser).

#

# It does this by periodically checking how many servers are waiting  
# for a request. If there are fewer than MinSpareServers, it creates  
# a new spare. If there are more than MaxSpareServers, some of the  
# spares die off. The default values are probably OK for most sites.

#

MinSpareServers 1

MaxSpareServers 5

#

# Number of servers to start initially --- should be a reasonable ballpark  
# figure.

#

StartServers 1

```
#
# Limit on total number of servers running, i.e., limit on the number
# of clients who can simultaneously connect --- if this limit is ever
# reached, clients will be LOCKED OUT, so it should NOT BE SET TOO LOW.
# It is intended mainly as a brake to keep a runaway server from taking
# the system with it as it spirals down...
#
MaxClients 150
```

```
#
# MaxRequestsPerChild: the number of requests each child process is
# allowed to process before the child dies. The child will exit so
# as to avoid problems after prolonged use when Apache (and maybe the
# libraries it uses) leak memory or other resources. On most systems, this
# isn't really needed, but a few (such as Solaris) do have notable leaks
# in the libraries. For these platforms, set to something like 10000
# or so; a setting of 0 means unlimited.
#
# NOTE: This value does not include keepalive requests after the initial
# request per connection. For example, if a child process handles
# an initial request and 10 subsequent "keptalive" requests, it
# would only count as 1 request towards this limit.
#
MaxRequestsPerChild 100000
```

```
#
# Listen: Allows you to bind Apache to specific IP addresses and/or
# ports, in addition to the default. See also the <VirtualHost>
# directive.
#
#Listen 3000
#Listen 12.34.56.78:80
```

```
#
# BindAddress: You can support virtual hosts with this option. This directive
# is used to tell the server which IP address to listen to. It can either
# contain "*", an IP address, or a fully qualified Internet domain name.
# See also the <VirtualHost> and Listen directives.
#
#BindAddress *
```

```
#
# Dynamic Shared Object (DSO) Support
```

```
#
# To be able to use the functionality of a module which was built as a DSO you
# have to place corresponding `LoadModule' lines at this location so the
# directives contained in it are actually available _before_ they are used.
# Please read the file README.DSO in the Apache 1.3 distribution for more
# details about the DSO mechanism and run `httpd -l' for the list of already
# built-in (statically linked and thus always available) modules in your httpd
# binary.
#
# Note: The order in which modules are loaded is important. Don't change
# the order below without expert advice.
#
# Example:
# LoadModule foo_module libexec/mod_foo.so
<IfDefine SSL>
LoadModule ssl_module      libexec/httpd/libssl.so
</IfDefine>

# Reconstruction of the complete module list from all available modules
# (static and shared ones) to achieve correct module execution order.
# [WHENEVER YOU CHANGE THE LOADMODULE SECTION ABOVE UPDATE
THIS, TOO]
ClearModuleList
AddModule mod_env.c
AddModule mod_log_config.c
AddModule mod_mime.c
AddModule mod_negotiation.c
AddModule mod_status.c
AddModule mod_include.c
AddModule mod_autoindex.c
AddModule mod_dir.c
AddModule mod_cgi.c
AddModule mod_asis.c
AddModule mod_imap.c
AddModule mod_actions.c
AddModule mod_userdir.c
AddModule mod_alias.c
AddModule mod_access.c
AddModule mod_auth.c
AddModule mod_so.c
AddModule mod_setenvif.c
<IfDefine SSL>
AddModule mod_ssl.c
```

```
</IfDefine>
AddModule mod_php4.c

#
# ExtendedStatus controls whether Apache will generate "full" status
# information (ExtendedStatus On) or just basic information (ExtendedStatus
# Off) when the "server-status" handler is called. The default is Off.
#
#ExtendedStatus On

### Section 2: 'Main' server configuration
#
# The directives in this section set up the values used by the 'main'
# server, which responds to any requests that aren't handled by a
# <VirtualHost> definition. These values also provide defaults for
# any <VirtualHost> containers you may define later in the file.
#
# All of these directives may appear inside <VirtualHost> containers,
# in which case these default settings will be overridden for the
# virtual host being defined.
#

#
# If your ServerType directive (set earlier in the 'Global Environment'
# section) is set to "inetd", the next few directives don't have any
# effect since their settings are defined by the inetd configuration.
# Skip ahead to the ServerAdmin directive.
#

#
# Port: The port to which the standalone server listens. For
# ports < 1023, you will need httpd to be run as root initially.
#
Port 80

##
## SSL Support
##
## When we also provide SSL we have to listen to the
## standard HTTP port (see above) and to the HTTPS port
##
<IfDefine SSL>
Listen 80
```

Listen 443  
</IfDefine>

```
#  
# If you wish httpd to run as a different user or group, you must run  
# httpd as root initially and it will switch.  
#  
# User/Group: The name (or #number) of the user/group to run httpd as.  
# . On SCO (ODT 3) use "User nouser" and "Group nogroup".  
# . On HPUX you may not be able to use shared memory as nobody, and the  
#   suggested workaround is to create a user www and use that user.  
# NOTE that some kernels refuse to setgid(Group) or semctl(IPC_SET)  
# when the value of (unsigned)Group is above 60000;  
# don't use Group nobody on these systems!  
#  
User nobody  
Group nobody  
  
#  
# ServerAdmin: Your address, where problems with the server should be  
# e-mailed. This address appears on some server-generated pages, such  
# as error documents.  
#  
ServerAdmin your@emailaddress  
  
#  
# ServerName allows you to set a host name which is sent back to clients for  
# your server if it's different than the one the program would get (i.e., use  
# "www" instead of the host's real name).  
#  
# Note: You cannot just invent host names and hope they work. The name you  
# define here must be a valid DNS name for your host. If you don't understand  
# this, ask your network administrator.  
# If your host doesn't have a registered DNS name, enter its IP address here.  
# You will have to access it by its address (e.g., http://123.45.67.89/  
# anyway, and this will make redirections work in a sensible way.  
#  
# 127.0.0.1 is the TCP/IP local loop-back address, often named localhost. Your  
# machine always knows itself by this address. If you use Apache strictly for  
# local testing and development, you may use 127.0.0.1 as the server name.  
#  
ServerName your.domain.com
```

```
#
# DocumentRoot: The directory out of which you will serve your
# documents. By default, all requests are taken from this directory, but
# symbolic links and aliases may be used to point to other locations.
#
DocumentRoot "/Library/WebServer/Documents"

#
# Each directory to which Apache has access, can be configured with respect
# to which services and features are allowed and/or disabled in that
# directory (and its subdirectories).
#
# First, we configure the "default" to be a very restrictive set of
# permissions.
#
<Directory />
    Options FollowSymLinks
    AllowOverride None
</Directory>

#
# Note that from this point forward you must specifically allow
# particular features to be enabled - so if something's not working as
# you might expect, make sure that you have specifically enabled it
# below.
#

#
# This should be changed to whatever you set DocumentRoot to.
#
<Directory "/Library/WebServer/Documents">

#
# This may also be "None", "All", or any combination of "Indexes",
# "Includes", "FollowSymLinks", "ExecCGI", or "MultiViews".
#
# Note that "MultiViews" must be named *explicitly* --- "Options All"
# doesn't give it to you.
#
    Options Indexes FollowSymLinks MultiViews

#
# This controls which options the .htaccess files in directories can
```

```
# override. Can also be "All", or any combination of "Options", "FileInfo",
# "AuthConfig", and "Limit"
#
    AllowOverride AuthConfig

#
# Controls who can get stuff from this server.
#
    Order allow,deny
    Allow from all
</Directory>

#
# UserDir: The name of the directory which is appended onto a user's home
# directory if a ~user request is received.
#
<IfModule mod_userdir.c>
    UserDir "Sites"
</IfModule>

#
# Control access to UserDir directories. The following is an example
# for a site where these directories are restricted to read-only.
#
#<Directory /home/*/public_html>
#   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>

#
# DirectoryIndex: Name of the file or files to use as a pre-written HTML
# directory index. Separate multiple entries with spaces.
#
<IfModule mod_dir.c>
    DirectoryIndex index.html
```

```
</IfModule>
```

```
#
```

```
# AccessFileName: The name of the file to look for in each directory  
# for access control information.
```

```
#
```

```
AccessFileName .htaccess
```

```
#
```

```
# The following lines prevent .htaccess files from being viewed by  
# Web clients. Since .htaccess files often contain authorization  
# information, access is disallowed for security reasons. Comment  
# these lines out if you want Web visitors to see the contents of  
# .htaccess files. If you change the AccessFileName directive above,  
# be sure to make the corresponding changes here.
```

```
#
```

```
# Also, folks tend to use names such as .htpasswd for password  
# files, so this will protect those as well.
```

```
#
```

```
<Files ~ "\.ht">
```

```
    Order allow,deny
```

```
    Deny from all
```

```
</Files>
```

```
#
```

```
# CacheNegotiatedDocs: By default, Apache sends "Pragma: no-cache" with each  
# document that was negotiated on the basis of content. This asks proxy  
# servers not to cache the document. Uncommenting the following line disables  
# this behavior, and proxies will be allowed to cache the documents.
```

```
#
```

```
#CacheNegotiatedDocs
```

```
#
```

```
# 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 On
```

```
#
```

```
# TypesConfig describes where the mime.types file (or equivalent) is
# to be found.
#
<IfModule mod_mime.c>
    TypesConfig /private/etc/httpd/mime.types
</IfModule>

#
# DefaultType is the default MIME type the server will use for a document
# if it cannot otherwise determine one, such as from filename extensions.
# If your server contains mostly text or HTML documents, "text/plain" is
# a good value.  If most of your content is binary, such as applications
# or images, you may want to use "application/octet-stream" instead to
# keep browsers from trying to display binary files as though they are
# text.
#
DefaultType text/plain

#
# The mod_mime_magic module allows the server to use various hints from the
# contents of the file itself to determine its type.  The MIMEMagicFile
# directive tells the module where the hint definitions are located.
# mod_mime_magic is not part of the default server (you have to add
# it yourself with a LoadModule [see the DSO paragraph in the 'Global
# Environment' section], or recompile the server and include mod_mime_magic
# as part of the configuration), so it's enclosed in an <IfModule> container.
# This means that the MIMEMagicFile directive will only be processed if the
# module is part of the server.
#
<IfModule mod_mime_magic.c>
    MIMEMagicFile /private/etc/httpd/magic
</IfModule>

#
# HostnameLookups: Log the names of clients or just their IP addresses
# e.g., www.apache.org (on) or 204.62.129.132 (off).
# The default is off because it'd be overall better for the net if people
# had to knowingly turn this feature on, since enabling it means that
# each client request will result in AT LEAST one lookup request to the
# nameserver.
#
HostnameLookups Off
```

```
#
# ErrorLog: The location of the error log file.
# If you do not specify an ErrorLog directive within a <VirtualHost>
# container, error messages relating to that virtual host will be
# logged here. If you *do* define an error logfile for a <VirtualHost>
# container, that host's errors will be logged there and not here.
#
ErrorLog /var/log/httpd/error_log

#
# LogLevel: Control the number of messages logged to the error_log.
# Possible values include: debug, info, notice, warn, error, crit,
# alert, emerg.
#
LogLevel warn

#
# The following directives define some format nicknames for use with
# a CustomLog directive (see below).
#
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

#
# The location and format of the access logfile (Common Logfile Format).
# If you do not define any access logfiles within a <VirtualHost>
# container, they will be logged here. Contrariwise, if you *do*
# define per-<VirtualHost> access logfiles, transactions will be
# logged therein and *not* in this file.
#
CustomLog "/private/var/log/httpd/access_log" common
#
# If you would like to have agent and referer logfiles, uncomment the
# following directives.
#
#CustomLog /var/log/httpd/referer_log referer
#CustomLog /var/log/httpd/agent_log agent

#
# If you prefer a single logfile with access, agent, and referer information
```

```
# (Combined Logfile Format) you can use the following directive.
#
#CustomLog "/private/var/log/httpd/access_log" combined

#
# Optionally add a line containing the server version and virtual host
# name to server-generated pages (error documents, FTP directory listings,
# mod_status and mod_info output etc., but not CGI generated documents).
# Set to "EMail" to also include a mailto: link to the ServerAdmin.
# Set to one of: On | Off | EMail
#
ServerSignature EMail

#
# Aliases: Add here as many aliases as you need (with no limit). The format is
# Alias fakename realname
#
<IfModule mod_alias.c>

#
# Note that if you include a trailing / on fakename then the server will
# require it to be present in the URL. So "/icons" isn't aliased in this
# example, only "/icons/"..
#
Alias /icons/ "/usr/share/httpd/icons/"

<Directory "/usr/share/httpd/icons">
    Options Indexes MultiViews
    AllowOverride None
    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/ "/Library/WebServer/CGI-Executables/"
```

```
#
# "/Library/WebServer/CGI-Executables" should be changed to whatever your
ScriptAliased
# CGI directory exists, if you have that configured.
#
<Directory "/Library/WebServer/CGI-Executables">
    AllowOverride None
    Options None
    Order allow,deny
    Allow from all
</Directory>

</IfModule>
# End of aliases.

#
# Redirect allows you to tell clients about documents which used to exist in
# your server's namespace, but do not anymore. This allows you to tell the
# clients where to look for the relocated document.
# Format: Redirect old-URI new-URL
#

#
# Directives controlling the display of server-generated directory listings.
#
<IfModule mod_autoindex.c>

#
# FancyIndexing is whether you want fancy directory indexing or standard
#
IndexOptions FancyIndexing

#
# AddIcon* directives tell the server which icon to show for different
# files or filename extensions. These are only displayed for
# FancyIndexed directories.
#
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 is which icon to show for files which do not have an icon
# explicitly set.
#
DefaultIcon /icons/unknown.gif
```

```
#
# AddDescription allows you to place a short description after a file in
# server-generated indexes. These are only displayed for FancyIndexed
# directories.
# Format: AddDescription "description" filename
#
#AddDescription "GZIP compressed document" .gz
#AddDescription "tar archive" .tar
#AddDescription "GZIP compressed tar archive" .tgz
```

```
#
# ReadmeName is the name of the README file the server will look for by
# default, and append to directory listings.
#
```

```
# HeaderName is the name of a file which should be prepended to
# directory indexes.
```

```
#
```

```
# If MultiViews are amongst the Options in effect, the server will
# first look for name.html and include it if found. If name.html
# doesn't exist, the server will then look for name.txt and include
# it as plaintext if found.
```

```
#
```

```
ReadmeName README
HeaderName HEADER
```

```
#
```

```
# IndexIgnore is a set of filenames which directory indexing should ignore
# and not include in the listing. Shell-style wildcarding is permitted.
```

```
#
```

```
IndexIgnore .??* *~ *# HEADER* README* RCS CVS *,v *,t
```

```
</IfModule>
```

```
# End of indexing directives.
```

```
#
```

```
# Document types.
```

```
#
```

```
<IfModule mod_mime.c>
```

```
#
```

```
# AddEncoding allows you to have certain browsers (Mosaic/X 2.1+)
uncompress
```

```
# information on the fly. Note: Not all browsers support this.
```

```
# Despite the name similarity, the following Add* directives have nothing
# to do with the FancyIndexing customization directives above.
```

```
#
```

```
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.
```

```
#
```

```
# Note 1: The suffix does not have to be the same as the language
# keyword --- those with documents in Polish (whose net-standard
# language code is pl) may wish to use "AddLanguage pl .po" to
```

```
# avoid the ambiguity with the common suffix for perl scripts.
#
# Note 2: The example entries below illustrate that in quite
# some cases the two character 'Language' abbreviation is not
# identical to the two character 'Country' code for its country,
# E.g. 'Danmark/dk' versus 'Danish/da'.
#
# Note 3: In the case of 'ltz' we violate the RFC by using a three char
# specifier. But there is 'work in progress' to fix this and get
# the reference data for rfc1766 cleaned up.
#
# Danish (da) - Dutch (nl) - English (en) - Estonian (ee)
# French (fr) - German (de) - Greek-Modern (el)
# Italian (it) - Korean (kr) - Norwegian (no)
# 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 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 .se
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. We have
# more or less alphabetized them here. You probably want to change this.
```

```
#
```

```
<IfModule mod_negotiation.c>
```

```
    LanguagePriority en da nl et fr de el it ja kr no pl pt pt-br ru ltz ca es sv tw
</IfModule>
```

```
#
```

```
# AddType allows you to tweak mime.types without actually editing it, or to
# make certain files to be certain types.
```

```
#
```

```
# For example, the PHP 3.x module (not part of the Apache distribution - see
# http://www.php.net) will typically use:
```

```
#
```

```
#AddType application/x-httpd-php3 .php3
```

```
#AddType application/x-httpd-php3-source .phps
```

```
#
```

```
# And for PHP 4.x, use:
```

```
#
```

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

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

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

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

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

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

```
AddType application/x-tar .tgz
```

```
#
# AddHandler allows you to map certain file extensions to "handlers",
# actions unrelated to filetype. These can be either built into the server
# or added with the Action command (see below)
#
# If you want to use server side includes, or CGI outside
# ScriptAliased directories, uncomment the following lines.
#
# To use CGI scripts:
#
#AddHandler cgi-script .cgi

#
# To use server-parsed HTML files
#
#AddType text/html .shtml
#AddHandler server-parsed .shtml

#
# Uncomment the following line to enable Apache's send-asis HTTP file
# feature
#
#AddHandler send-as-is asis

#
# If you wish to use server-parsed imagemap files, use
#
#AddHandler imap-file map

#
# To enable type maps, you might want to use
#
#AddHandler type-map var

</IfModule>
# End of document types.

#
# Action lets you define media types that will execute a script whenever
# a matching file is called. This eliminates the need for repeated URL
# pathnames for oft-used CGI file processors.
# Format: Action media/type /cgi-script/location
# Format: Action handler-name /cgi-script/location
```

```
#

#
# MetaDir: specifies the name of the directory in which Apache can find
# meta information files. These files contain additional HTTP headers
# to include when sending the document
#
#MetaDir .web

#
# MetaSuffix: specifies the file name suffix for the file containing the
# meta information.
#
#MetaSuffix .meta

#
# Customizable error response (Apache style)
# these come in three flavors
#
# 1) plain text
#ErrorDocument 500 "The server made a boo boo.
# n.b. the single leading (") marks it as text, it does not get output
#
# 2) local redirects
#ErrorDocument 404 /missing.html
# to redirect to local URL /missing.html
#ErrorDocument 404 /cgi-bin/missing_handler.pl
# N.B.: You can redirect to a script or a document using server-side-includes.
#
# 3) external redirects
#ErrorDocument 402 http://some.other_server.com/subscription_info.html
# N.B.: Many of the environment variables associated with the original
# request will *not* be available to such a script.

#
# 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>
# End of browser customization directives

#
# Allow server status reports, with the URL of http://servername/server-status
# Change the ".your_domain.com" to match your domain to enable.
#
#<Location /server-status>
#   SetHandler server-status
#   Order deny,allow
#   Deny from all
#   Allow from .your_domain.com
#</Location>

#
# Allow remote server configuration reports, with the URL of
# http://servername/server-info (requires that mod_info.c be loaded).
# Change the ".your_domain.com" to match your domain to enable.
#
#<Location /server-info>
#   SetHandler server-info
#   Order deny,allow
#   Deny from all
#   Allow from .your_domain.com
#</Location>

#
```

```
# There have been reports of people trying to abuse an old bug from pre-1.1
# days. This bug involved a CGI script distributed as a part of Apache.
# By uncommenting these lines you can redirect these attacks to a logging
# script on phf.apache.org. Or, you can record them yourself, using the script
# support/phf_abuse_log.cgi.
#
#<Location /cgi-bin/phf*>
#   Deny from all
#   ErrorDocument 403 http://phf.apache.org/phf_abuse_log.cgi
#</Location>

#
# Proxy Server directives. Uncomment the following lines to
# enable the proxy server:
#
#<IfModule mod_proxy.c>
#   ProxyRequests On

#   <Directory proxy:*>
#       Order deny,allow
#       Deny from all
#       Allow from .your_domain.com
#   </Directory>

#
# Enable/disable the handling of HTTP/1.1 "Via:" headers.
# ("Full" adds the server version; "Block" removes all outgoing Via: headers)
# Set to one of: Off | On | Full | Block
#
#   ProxyVia On

#
# To enable the cache as well, edit and uncomment the following lines:
# (no cacheing without CacheRoot)
#
#   CacheRoot "/var/run/proxy"
#   CacheSize 5
#   CacheGcInterval 4
#   CacheMaxExpire 24
#   CacheLastModifiedFactor 0.1
#   CacheDefaultExpire 1
#   NoCache a_domain.com another_domain.edu joes.garage_sale.com
```

```
#</IfModule>
# End of proxy directives.

### Section 3: Virtual Hosts
#
# 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.
#
# Please see the documentation at <URL:http://www.apache.org/docs/vhosts/>
# for further details before you try to setup virtual hosts.
#
# You may use the command line option '-S' to verify your virtual host
# configuration.

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

#
# VirtualHost example:
# Almost any Apache directive may go into a VirtualHost container.
# The first VirtualHost section is used for requests without a known
# server name.
#
#<VirtualHost *>
#   ServerAdmin webmaster@dummy-host.example.com
#   DocumentRoot /www/docs/dummy-host.example.com
#   ServerName dummy-host.example.com
#   ErrorLog logs/dummy-host.example.com-error_log
#   CustomLog logs/dummy-host.example.com-access_log common
#</VirtualHost>

Include /private/etc/httpd/users

#<VirtualHost _default_.*>
#</VirtualHost>

##
## SSL Global Context
##
```

```
## All SSL configuration in this context applies both to
## the main server and all SSL-enabled virtual hosts.
##

#
# Some MIME-types for downloading Certificates and CRLs
#
<IfDefine SSL>
AddType application/x-x509-ca-cert .crt
AddType application/x-pkcs7-crl .crl
</IfDefine>

<IfModule mod_ssl.c>

# Pass Phrase Dialog:
# Configure the pass phrase gathering process.
# The filtering dialog program ('builtin' is a internal
# terminal dialog) has to provide the pass phrase on stdout.
SSLPassPhraseDialog builtin

# Inter-Process Session Cache:
# Configure the SSL Session Cache: First the mechanism
# to use and second the expiring timeout (in seconds).
#SSLSessionCache none
#SSLSessionCache shmht:/var/run/ssl_scache(512000)
#SSLSessionCache shmcb:/var/run/ssl_scache(512000)
SSLSessionCache dbm:/var/run/ssl_scache
SSLSessionCacheTimeout 300

# Semaphore:
# Configure the path to the mutual exclusion semaphore the
# SSL engine uses internally for inter-process synchronization.
SSLMutex file:/var/run/ssl_mutex

# Pseudo Random Number Generator (PRNG):
# Configure one or more sources to seed the PRNG of the
# SSL library. The seed data should be of good random quality.
# WARNING! On some platforms /dev/random blocks if not enough entropy
# is available. This means you then cannot use the /dev/random device
# because it would lead to very long connection times (as long as
# it requires to make more entropy available). But usually those
# platforms additionally provide a /dev/urandom device which doesn't
# block. So, if available, use this one instead. Read the mod_ssl User
```

```
# Manual for more details.
SSLRandomSeed startup builtin
SSLRandomSeed connect builtin
#SSLRandomSeed startup file:/dev/random 512
#SSLRandomSeed startup file:/dev/urandom 512
#SSLRandomSeed connect file:/dev/random 512
#SSLRandomSeed connect file:/dev/urandom 512

# Logging:
# The home of the dedicated SSL protocol logfile. Errors are
# additionally duplicated in the general error log file. Put
# this somewhere where it cannot be used for symlink attacks on
# a real server (i.e. somewhere where only root can write).
# Log levels are (ascending order: higher ones include lower ones):
# none, error, warn, info, trace, debug.
SSLLog /var/log/httpd/ssl_engine_log
SSLLogLevel info

</IfModule>

<IfDefine SSL>

##
## SSL Virtual Host Context
##

<VirtualHost _default_:443>

# General setup for the virtual host
DocumentRoot "/Library/WebServer/Documents"
ServerName #same as your Apache ServerName
ServerAdmin #Same as your Apache ServerAdmin
ErrorLog /var/log/httpd/error_log
TransferLog /var/log/httpd/access_log

# SSL Engine Switch:
# Enable/Disable SSL for this virtual host.
SSLEngine on

# SSL Cipher Suite:
# List the ciphers that the client is permitted to negotiate.
# See the mod_ssl documentation for a complete list.
SSLCipherSuite
```

ALL:!ADH:!EXPORT56:RC4+RSA:+HIGH:+MEDIUM:+LOW:+SSLv2:+EXP:+eNULL

# Server Certificate:

# Point SSLCertificateFile at a PEM encoded certificate. If  
# the certificate is encrypted, then you will be prompted for a  
# pass phrase. Note that a kill -HUP will prompt again. A test  
# certificate can be generated with `make certificate' under  
# built time. Keep in mind that if you've both a RSA and a DSA  
# certificate you can configure both in parallel (to also allow  
# the use of DSA ciphers, etc.)

SSLCertificateFile /private/etc/httpd/ssl.crt/server.crt

#SSLCertificateFile /etc/httpd/ssl.crt/server-dsa.crt

# Server Private Key:

# If the key is not combined with the certificate, use this  
# directive to point at the key file. Keep in mind that if  
# you've both a RSA and a DSA private key you can configure  
# both in parallel (to also allow the use of DSA ciphers, etc.)

SSLCertificateKeyFile /private/etc/httpd/ssl.key/server.key

#SSLCertificateKeyFile /etc/httpd/ssl.key/server-dsa.key

# Server Certificate Chain:

# Point SSLCertificateChainFile at a file containing the  
# concatenation of PEM encoded CA certificates which form the  
# certificate chain for the server certificate. Alternatively  
# the referenced file can be the same as SSLCertificateFile  
# when the CA certificates are directly appended to the server  
# certificate for convinience.

#SSLCertificateChainFile /private/etc/httpd/ssl.crt/ca.crt

# Certificate Authority (CA):

# Set the CA certificate verification path where to find CA  
# certificates for client authentication or alternatively one  
# huge file containing all of them (file must be PEM encoded)  
# Note: Inside SSLCACertificatePath you need hash symlinks  
# to point to the certificate files. Use the provided  
# Makefile to update the hash symlinks after changes.

#SSLCACertificatePath /etc/httpd/ssl.crt

#SSLCACertificateFile /etc/httpd/ssl.crt/ca-bundle.crt

# Certificate Revocation Lists (CRL):

# Set the CA revocation path where to find CA CRLs for client  
# authentication or alternatively one huge file containing all

```

# of them (file must be PEM encoded)
# Note: Inside SSLCARevocationPath you need hash symlinks
#     to point to the certificate files. Use the provided
#     Makefile to update the hash symlinks after changes.
#SSLCARevocationPath /etc/httpd/ssl.crl
#SSLCARevocationFile /etc/httpd/ssl.crl/ca-bundle.crl

# Client Authentication (Type):
# Client certificate verification type and depth. Types are
# none, optional, require and optional_no_ca. Depth is a
# number which specifies how deeply to verify the certificate
# issuer chain before deciding the certificate is not valid.
#SSLVerifyClient require
#SSLVerifyDepth 10

# Access Control:
# With SSLRequire you can do per-directory access control based
# on arbitrary complex boolean expressions containing server
# variable checks and other lookup directives. The syntax is a
# mixture between C and Perl. See the mod_ssl documentation
# for more details.
#<Location />
#SSLRequire (   %{SSL_CIPHER} !~ m/^(EXP|NULL)/ \
#     and %{SSL_CLIENT_S_DN_O} eq "Snake Oil, Ltd." \
#     and %{SSL_CLIENT_S_DN_OU} in {"Staff", "CA", "Dev"} \
#     and %{TIME_WDAY} >= 1 and %{TIME_WDAY} <= 5 \
#     and %{TIME_HOUR} >= 8 and %{TIME_HOUR} <= 20   ) \
#     or %{REMOTE_ADDR} =~ m/^192\.76\.162\. [0-9]+$/
#</Location>

# SSL Engine Options:
# Set various options for the SSL engine.
# o FakeBasicAuth:
#   Translate the client X.509 into a Basic Authorisation. This means that
#   the standard Auth/DBMAuth methods can be used for access control. The
#   user name is the `one line' version of the client's X.509 certificate.
#   Note that no password is obtained from the user. Every entry in the user
#   file needs this password: `xxj31ZMTZzkVA'.
# o ExportCertData:
#   This exports two additional environment variables: SSL_CLIENT_CERT and
#   SSL_SERVER_CERT. These contain the PEM-encoded certificates of the
#   server (always existing) and the client (only existing when client
#   authentication is used). This can be used to import the certificates

```

```

# into CGI scripts.
# o StdEnvVars:
# This exports the standard SSL/TLS related `SSL_*' environment variables.
# Per default this exportation is switched off for performance reasons,
# because the extraction step is an expensive operation and is usually
# useless for serving static content. So one usually enables the
# exportation for CGI and SSI requests only.
# o CompatEnvVars:
# This exports obsolete environment variables for backward compatibility
# to Apache-SSL 1.x, mod_ssl 2.0.x, Sioux 1.0 and Stronghold 2.x. Use this
# to provide compatibility to existing CGI scripts.
# o StrictRequire:
# This denies access when "SSLRequireSSL" or "SSLRequire" applied even
# under a "Satisfy any" situation, i.e. when it applies access is denied
# and no other module can change it.
# o OptRenegotiate:
# This enables optimized SSL connection renegotiation handling when SSL
# directives are used in per-directory context.
#SSLOptions +FakeBasicAuth +ExportCertData +CompatEnvVars +StrictRequire
<Files ~ "\.(cgi|shtml|phtml|php3?)$">
    SSLOptions +StdEnvVars
</Files>
<Directory "/Library/WebServer/CGI-Executables">
    SSLOptions +StdEnvVars
</Directory>

# SSL Protocol Adjustments:
# The safe and default but still SSL/TLS standard compliant shutdown
# approach is that mod_ssl sends the close notify alert but doesn't wait for
# the close notify alert from client. When you need a different shutdown
# approach you can use one of the following variables:
# o ssl-unclean-shutdown:
# This forces an unclean shutdown when the connection is closed, i.e. no
# SSL close notify alert is send or allowed to received. This violates
# the SSL/TLS standard but is needed for some brain-dead browsers. Use
# this when you receive I/O errors because of the standard approach where
# mod_ssl sends the close notify alert.
# o ssl-accurate-shutdown:
# This forces an accurate shutdown when the connection is closed, i.e. a
# SSL close notify alert is send and mod_ssl waits for the close notify
# alert of the client. This is 100% SSL/TLS standard compliant, but in
# practice often causes hanging connections with brain-dead browsers. Use
# this only for browsers where you know that their SSL implementation

```

```
# works correctly.
# Notice: Most problems of broken clients are also related to the HTTP
# keep-alive facility, so you usually additionally want to disable
# keep-alive for those clients, too. Use variable "nokeepalive" for this.
# Similarly, one has to force some clients to use HTTP/1.0 to workaround
# their broken HTTP/1.1 implementation. Use variables "downgrade-1.0" and
# "force-response-1.0" for this.
```

```
SetEnvIf User-Agent ".*MSIE.*" \
    nokeepalive ssl-unclean-shutdown \
    downgrade-1.0 force-response-1.0
```

```
# Per-Server Logging:
```

```
# The home of a custom SSL log file. Use this when you want a
# compact non-error SSL logfile on a virtual host basis.
```

```
CustomLog /var/log/httpd/ssl_request_log \
    "%t %h %{SSL_PROTOCOL}x %{SSL_CIPHER}x \"%r\" %b"
```

```
</VirtualHost>
```

```
</IfDefine>
```