mod_gnutls Manual

Version 0.1

July 2011

Contents:

Compilation & Installation:

mod_gnutls uses the "configure/make/make install" mechanism common to many Open Source programs. Most of the dirty work is handled by either configure or Apache's apxs utility. If you have built Apache modules before, there shouldn't be any surprises for you.

The interesting options you can pass to configure are:

- --with-apxs=PATH
  This option is used to specify the location of the apxs utility that was installed as part of apache. Specify the location of the binary, not the directory it is located in.
- --with-libgnutls=PATH
  Full path to the libgnutls-config program.
- --with-apr-memcache=PREFIX
  Prefix to where apr_memcache is installed.
- --help
  Provides a list of all available configure options.

Integration into Apache:

To activate **mod_gnutls** just add the following line to your httpd.conf and restart Apache:

```
LoadModule gnutls_module modules/mod_gnutls.so
```

Configuration Directives:

GnuTLSCache

Description:	Configure SSL Session Cache.
Syntax:	GnuTLSCache [dbm|gdbm|memcache|none] [path|server list|-]
Default:	GnuTLSCache gdbm "conf/gnutls_cache"
Context:	global config

This directive configures the SSL Session Cache for **mod_gnutls**.

This could be shared between machines of different architectures.

dbm (Requires Berkeley DBM)

Uses the default Berkeley DB backend of APR DBM to cache SSL Sessions results.

The argument is a relative or absolute path to be used as the DBM Cache file.

This is compatible with most operating systems, but needs the Apache Runtime to be compiled with Berkeley DBM support.

gdbm (Default)

Uses the GDBM backend of APR DBM to cache SSL Sessions results.

The argument is a relative or absolute path to be used as the DBM Cache file.

This is the default and recommended option.

memcache

Uses a memcached server to cache the SSL Session.

The argument is a space separated list of servers. If no port number is supplied, the default of 11211 is used.

This can be used to share a session cache between all servers in a cluster.

none

Turns off all caching of SSL Sessions.

This can significantly reduce the performance of **mod_gnutls** since even followup connections by a client must renegotiate parameters instead of reusing old ones.

GnuTLSCacheTimeout

Description:	Timeout for SSL Session Cache expiration.
Syntax:	GnuTLSCacheTimeout *seconds*
Default:	300
Context:	global config

Sets the timeout for SSL Session Cache entries expiration. This directive is valid even if Session Tickets are used, and indicates the expiration time of the ticket.

GnuTLSSessionTickets

Description:	Enable Session Tickets for the server.
Syntax:	GnuTLSSessionTickets [*on* | *off*]
Default:	off
Context:	server config, virtual host

To avoid storing data for TLS session resumption it is allowed to provide client with a ticket, to use on return. Use for servers with limited storage, and don't combine with GnuTLSCache. For a pool of servers this option is not recommended since the tickets are unique for the issuing server only.

GnuTLSCertificateFile

Description:	Set to the PEM Encoded Server Certificate.
Syntax:	GnuTLSCertificateFile *file-path*
Default:	none
Context:	server config, virtual host

Takes an absolute or relative path to a PEM Encoded Certificate to use as this Server's Certificate.

GnuTLSKeyFile

Description:	Set to the Server Private Key.
Syntax:	GnuTLSKeyFile *file-path*
Default:	none
Context:	server config, virtual host

Takes an absolute or relative path to the Server Private Key. This key cannot currently be password protected.

Security Warning: This private key must be protected. It is read while Apache is still running as root, and does not need to be readable by the nobody or apache user.

GnuTLSPGPCertificateFile

Description:	Set to a base64 Encoded Server OpenPGP Certificate.
Syntax:	GnuTLSPGPCertificateFile *file-path*
Default:	none
Context:	server config, virtual host

Takes an absolute or relative path to a base64 Encoded OpenPGP Certificate to use as this Server's Certificate.

GnuTLSPGPKeyFile

Description:	Set to the Server OpenPGP Secret Key.
Syntax:	GnuTLSPGPCertificateFile *file-path*
Default:	none
Context:	server config, virtual host

Takes an absolute or relative path to the Server Private Key. This key cannot currently be password protected.

Security Warning: This private key must be protected. It is read while Apache is still running as root, and does not need to be readable by the nobody or apache user.

GnuTLSClientVerify

Description:	Enable Client Certificate Verification.
Syntax:	GnuTLSClientVerify [*ignore*|*request*|*require*]
Default:	*ignore*
Context:	server config, virtual host, directory, .htaccess

This directive controls the use of SSL Client Certificate Authentication. If used in the .htaccess context, it can force TLS re-negotiation.

ignore

mod_gnutls will ignore the contents of any SSL Client Certificates sent. It will not request that the client sends a certificate.

request

The client certificate will be requested, but not required. The Certificate will be validated if sent. The output of the validation status will be stored in the SSL_CLIENT_VERIFY environment variable and can be "SUCCESS", "FAILED" or "NONE".

require

A Client certificate will be required. Any requests without a valid client certificate will be denied. The SSL_CLIENT_VERIFY environment variable will only be set to "SUCCESS".

GnuTLSClientCAFile

Description:	Set to the PEM Encoded Certificate Authority Certificate.
Syntax:	GnuTLSClientCAFile *file-path*
Default:	none
Context:	server config, virtual host

Takes an absolute or relative path to a PEM Encoded Certificate to use as a Certificate Authority with Client Certificate Authentication. This file may contain a list of trusted authorities.

GnuTLSPGPKeyringFile

Description:	Set to a base64 Encoded Server OpenPGP Certificate.
Syntax:	GnuTLSPGPCertificateFile *file-path*
Default:	none
Context:	server config, virtual host

Takes an absolute or relative path to a base64 Encoded Certificate list (key ring) to use as a means of verification of Client Certificates. This file should contain a list of trusted signers.

GnuTLSEnable

Description:	Enable GnuTLS for this virtual host.
Syntax:	GnuTLSEnable [*on* | *off*]
Default:	*off*
Context:	virtual host

This directive enables SSL/TLS Encryption for a Virtual Host.

GnuTLSDHFile

Description:	Set to the PKCS #3 encoded Diffie Hellman parameters.
Syntax:	GnuTLSDHFile *file-path*
Default:	none
Context:	server config, virtual host

Takes an absolute or relative path to a PKCS #3 encoded DH parameters. Those are used when the DHE key exchange method is enabled. You can generate this file using "certtool --generate-dh-params --bits 2048". If not set mod_gnutls will use the included parameters.

GnuTLSRSAFile

Description:	Set to the PKCS #1 encoded RSA parameters for 'EXPORT' ciphersuites.
Syntax:	GnuTLSRSAFile *file-path*
Default:	none
Context:	server config, virtual host

Takes an absolute or relative path to a PKCS #1 encoded RSA parameters. Those are used when the RSA-EXPORT key exchange method is enabled. You can generate this file using "certtool --generate-privkey --bits 512". These parameters should not contain key of longer of 512 bits (due to the export restrictions). If not set mod_gnutls will not negotiate the 'EXPORT' ciphersuites. It is recommended not to enable those ciphersuites. If you do make sure you regenerate this file at every few hours.

GnuTLSSRPPasswdFile

Description:	Set to the SRP password file for SRP ciphersuites.
Syntax:	GnuTLSSRPPasswdFile *file-path*
Default:	none
Context:	server config, virtual host

Takes an absolute or relative path to an SRP password file. This is the same format as used in libsrp. You can generate such file using the command "srptool --passwd /etc/tpasswd --passwd-conf /etc/tpasswd.conf -u test" to set a password for user test. This password file holds the username, a password verifier and the dependency to the SRP parameters.

GnuTLSSRPPasswdConfFile

Description:	Set to the SRP password.conf file for SRP ciphersuites.
Syntax:	GnuTLSSRPPasswdConfFile *file-path*
Default:	none
Context:	server config, virtual host

Takes an absolute or relative path to an SRP password.conf file. This is the same format as used in libsrp. You can generate such file using the command "srptool --create-conf /etc/tpasswd.conf". This file holds the SRP parameters and is associate with the password file (the verifiers depends on these parameters).

GnuTLSPriorities

Description:	Set the allowed ciphers, key exchange algorithms, MACs and compression methods.
Syntax:	GnuTLSPriorities *+cipher0:+cipher1:...:+cipherN*
Default:	none
Context:	server config, virtual host

Takes a semi-colon separated list of ciphers, key exchange methods.

Message authentication codes and compression methods to enable.

The allowed keywords are specified in the gnutls_priority_init() function of GnuTLS.

Its documentation can be found at Core GnuTLS functions.

In brief you can specify a set of ciphersuites from the choices:

NONE: The empty list.

EXPORT: A list with all the supported cipher combinations including the "EXPORT" strength algorithms.

PERFORMANCE: A list with all the secure cipher combinations sorted in terms of performance.

NORMAL: A list with all the secure cipher combinations sorted with respect to security margin (subjective term).

SECURE: A list with all the secure cipher combinations including the 256-bit ciphers sorted with respect to security margin.

Additionally you can add or remove algorithms using the "+" and "!" prefixes respectively. That is in order to disable the ARCFOUR cipher from the "NORMAL" set you can use the string

NORMAL:!ARCFOUR-128.

Other options such as the protocol version and the compression method can be specified using the VERS- and COMP- prefixes. So in order to remove or add a specific TLS version from the "NORMAL" set use NORMAL:!VERS-SSL3.0. To enable zlib compression use NORMAL:+COMP-DEFLATE. However it is recommended not to add compression at this level.

With the "NONE" set, in order to be usable, you have to specify a complete set of combinations of protocol versions, cipher algorithms (AES-128-CBC), key exchange algorithms (RSA), message authentication codes (SHA1) and compression methods (COMP-NULL).

All the supported algorithms are:

Ciphers:

AES-256-CBC, AES-128-CBC, CAMELLIA-256-CBC, CAMELLIA-128-CBC, ARCFOUR-128, 3DES-CBC, ARCFOUR-40.

Key exchange methods:

RSA, DHE-RSA, DHE-DSS, SRP, SRP-RSA, SRP-DSS, ANON-DH

Message authentication codes:

SHA1, MD5.

Compression methods:

COMP-DEFLATE, COMP-NULL

Protocol versions:

VERS-TLS1.1, VERS-TLS1.0, VERS-SSL3.0

The special keyword "%COMPAT" will disable some security features such as protection against statistical attacks to ciphertext data in order to achieve maximum compatibility (some broken mobile clients need this).

GnuTLSExportCertificates

Description:	Export the PEM encoded certificates to CGIs.
Syntax:	GnuTLSExportCertificates [*on* | *off*]
Default:	*off*
Context:	virtual host

This directive enables exporting the full PEM encoded certificates of the server and the client to CGIs. This makes mod_gnutls export exactly the same environment variables as mod_ssl.

Configuration Examples:

Simple Standard SSL Example:

The following is an example of standard SSL Hosting, using one IP Addresses for each virtual host:

```
# Load the module into Apache.
LoadModule gnutls_module modules/mod_gnutls.so

GnuTLSCache gdbm /var/cache/www-tls-cache
GnuTLSCacheTimeout 500

# With normal SSL Websites, you need one IP Address per-site.
Listen 1.2.3.1:443
Listen 1.2.3.2:443
Listen 1.2.3.3:443
Listen 1.2.3.4:443

<VirtualHost 1.2.3.1:443>
   GnuTLSEnable on
   GnuTLSPriorities NONE:+AES-128-CBC:+3DES-CBC:+ARCFOUR-128:+RSA:+DHE-RSA:
+DHE-DSS:+SHA1:+MD5:+COMP-NULL
   DocumentRoot /www/site1.example.com/html
   ServerName site1.example.com:443
   GnuTLSCertificateFile conf/ssl/site1.crt
   GnuTLSKeyFile conf/ss/site1.key
</VirtualHost>
<VirtualHost 1.2.3.2:443>
# This virtual host enables SRP authentication
   GnuTLSEnable on
   GnuTLSPriorities NORMAL:+SRP
   DocumentRoot /www/site2.example.com/html
   ServerName site2.example.com:443
   GnuTLSSRPPasswdFile conf/ssl/tpasswd.site2
   GnuTLSSRPPasswdConfFile conf/ssl/tpasswd.site2.conf
</VirtualHost>
<VirtualHost 1.2.3.3:443>
# This server enables SRP, OpenPGP and X.509 authentication.
   GnuTLSEnable on
   GnuTLSPriorities NORMAL:+SRP:+SRP-RSA:+SRP-DSS
   DocumentRoot /www/site3.example.com/html
   ServerName site3.example.com:443
   GnuTLSCertificateFile conf/ssl/site3.crt
   GnuTLSKeyFile conf/ss/site3.key
   GnuTLSClientVerify ignore
   GnuTLSPGPCertificateFile conf/ss/site3.pub.asc
   GnuTLSPGPKeyFile conf/ss/site3.sec.asc
   GnuTLSSRPPasswdFile conf/ssl/tpasswd.site3
   GnuTLSSRPPasswdConfFile conf/ssl/tpasswd.site3.conf
</VirtualHost>
```

```
<VirtualHost 1.2.3.4:443>
   GnuTLSEnable on
# %COMPAT disables some security features to enable maximum compatibility with clients.
   GnuTLSPriorities NONE:+AES-128-CBC:+ARCFOUR-128:+RSA:+SHA1:+MD5:+COMP-NULL:
%COMPAT
   DocumentRoot /www/site4.example.com/html
   ServerName site4.example.com:443
   GnuTLSCertificateFile conf/ssl/site4.crt
   GnuTLSKeyFile conf/ss/site4.key
</VirtualHost>
```

Server Name Indication Example:

mod_gnutls can also use 'Server Name Indication', as specified in RFC 3546.
This allows hosting many SSL Websites, with a Single IP Address.
Currently all the recent browsers support this standard.
Here is an example, using SNI:

```
# Load the module into Apache.
LoadModule gnutls_module modules/mod_gnutls.so

# With normal SSL Websites, you need one IP Address per-site.
Listen 1.2.3.1:443
# This could also be 'Listen *:443',
# just like '*:80' is common for non-https

# No caching. Enable session tickets. Timeout is still used for
# ticket expiration.
GnuTLSCacheTimeout 600

# This tells apache, that for this IP/Port combination, we want to use
# Name Based Virtual Hosting.  In the case of Server Name Indication,
# it lets mod_gnutls pick the correct Server Certificate.
NameVirtualHost 1.2.3.1:443

<VirtualHost 1.2.3.1:443>
   GnuTLSEnable on
   GnuTLSSessionTickets on
   GnuTLSPriorities NORMAL
   DocumentRoot /www/site1.example.com/html
   ServerName site1.example.com:443
   GnuTLSCertificateFile conf/ssl/site1.crt
   GnuTLSKeyFile conf/ss/site1.key
</VirtualHost>
```

```
<VirtualHost 1.2.3.1:443>
   GnuTLSEnable on
   GnuTLSPriorities NORMAL
   DocumentRoot /www/site2.example.com/html
   ServerName site2.example.com:443
   GnuTLSCertificateFile conf/ssl/site2.crt
   GnuTLSKeyFile conf/ss/site2.key
</VirtualHost>
<VirtualHost 1.2.3.1:443>
   GnuTLSEnable on
   GnuTLSPriorities NORMAL
   DocumentRoot /www/site3.example.com/html
   ServerName site3.example.com:443
   GnuTLSCertificateFile conf/ssl/site3.crt
   GnuTLSKeyFile conf/ss/site3.key
</VirtualHost>
<VirtualHost 1.2.3.1:443>
   GnuTLSEnable on
   GnuTLSPriorities NORMAL
   DocumentRoot /www/site4.example.com/html
   ServerName site4.example.com:443
   GnuTLSCertificateFile conf/ssl/site4.crt
   GnuTLSKeyFile conf/ss/site4.key
</VirtualHost>
```

Performance Issues:

mod_gnutls by default uses conservative settings for the server.
You can fine tune the configuration to reduce the load on a busy server.
The following examples do exactly this:

```
# Load the module into Apache.
LoadModule gnutls_module modules/mod_gnutls.so

# Using 4 memcache servers to distribute the SSL Session Cache.
GnuTLSCache memcache "mc1.example.com mc2.example.com mc3.example.com
mc4.example.com"
GnuTLSCacheTimeout 600

Listen 1.2.3.1:443
NameVirtualHost 1.2.3.1:443
```

```
<VirtualHost 1.2.3.1:443>
   GnuTLSEnable on
# Here we disable the Perfect forward secrecy ciphersuites (DHE)
# and disallow AES-256 since AES-128 is just fine.
   GnuTLSPriorities NORMAL:!DHE-RSA:!DHE-DSS:!AES-256-CBC:%COMPAT
   DocumentRoot /www/site1.example.com/html
   ServerName site1.example.com:443
   GnuTLSCertificateFile conf/ssl/site1.crt
   GnuTLSKeyFile conf/ss/site1.key
</VirtualHost>
<VirtualHost 1.2.3.1:443>
   GnuTLSEnable on
# Here we instead of disabling the DHE ciphersuites we use
# Diffie Hellman parameters of smaller size than the default (2048 bits).
# Using small numbers from 768 to 1024 bits should be ok once they are
# regenerated every few hours.
# Use "certtool --generate-dh-params --bits 1024" to get those
   GnuTLSDHFile /etc/apache2/dh.params
   GnuTLSPriorities NORMAL:!AES-256-CBC:%COMPAT
   DocumentRoot /www/site2.example.com/html
   ServerName site2.example.com:443
   GnuTLSCertificateFile conf/ssl/site2.crt
   GnuTLSKeyFile conf/ss/site2.key
</VirtualHost>
```

Environment variables:

mod_gnutls exports the following environment variables to scripts.
These are compatible with **mod_ssl**.

- **HTTPS:** Can be "on" or "off".
- **SSL_VERSION_LIBRARY:** The version of the gnutls library.
- **SSL_VERSION_INTERFACE:** The version of this module.
- **SSL_PROTOCOL:** The SSL or TLS protocol name (such as "TLS 1.0" etc.).
- **SSL_CIPHER:** The SSL or TLS cipher suite name.
- **SSL_COMPRESS_METHOD:** The negotiated compression method (NULL or DEFLATE).
- **SSL_SRP_USER:** The SRP username used for authentication.
- **SSL_CIPHER_USEKEYSIZE & SSL_CIPHER_ALGKEYSIZE:** The number if bits used in the used cipher algorithm. This does not fully reflect the security level since the size of RSA or DHE key exchange parameters affect the security level too.
- **SSL_CIPHER_EXPORT:** True or false. Whether the cipher suite negotiated is an export one.
- **SSL_SESSION_ID:** The session ID negotiated in this session. Can be the same during client reloads.
- **SSL_CLIENT_V_REMAIN:** The number of days until the client's certificate is expired.
- **SSL_CLIENT_V_START:** The activation time of client's certificate.
- **SSL_CLIENT_V_END:** The expiration time of client's certificate.
- **SSL_CLIENT_S_DN:** The distinguished name of client's certificate in RFC2253 format.

- **SSL_CLIENT_I_DN:** The distinguished name of client's issuer certificate in RFC2253 format.
- **SSL_CLIENT_S_AN%:** These will contain the alternative names of the client certificate (% is a number starting from zero). The values will be prepended by "DNSNAME:","RFC822NAME:" or "URI:" depending on the type. If it is not supported the value "UNSUPPORTED" will be set.
- **SSL_CLIENT_M_SERIAL:** The serial number of the client's certificate.
- **SSL_CLIENT_M_VERSION:** The version of the client's certificate.
- **SSL_CLIENT_A_SIG:** The algorithm used for the signature in client's certificate.
- **SSL_CLIENT_A_KEY:** The public key algorithm in client's certificate.
- **SSL_CLIENT_CERT:** The PEM-encoded client certificate.
- **SSL_CLIENT_VERIFY:** Whether the client's certificate was verified. (NONE if none was sent, or SUCCESS or FAILED).
- **SSL_CLIENT_CERT_TYPE:** The certificate type can be X.509 or OPENPGP.
- **SSL_SERVER_V_START:** The activation time of server's certificate.
- **SSL_SERVER_V_END:** The expiration time of server's certificate.
- **SSL_SERVER_S_DN:** The distinguished name of the server's certificate in RFC2253 format.
- **SSL_SERVER_I_DN:** The distinguished name of the server's issuer certificate in RFC2253 format.
- **SSL_SERVER_S_AN%:** These will contain the alternative names of the server certificate (% is a number starting from zero). The values will be prepended by "DNSNAME:", "RFC822NAME:" or "URI:" depending on the type. If it is not supported the value "UNSUPPORTED" will be set.
- **SSL_SERVER_M_SERIAL:** The serial number of the server's certificate.
- **SSL_SERVER_M_VERSION:** The version of the server's certificate.
- **SSL_SERVER_A_SIG:** The algorithm used for the signature in server's certificate.
- **SSL_SERVER_A_KEY:** The public key algorithm in server's certificate.
- **SSL_SERVER_CERT:** The PEM-encoded server certificate.
- **SSL_SERVER_CERT_TYPE:** The certificate type can be X.509 or OPENPGP.

Credits:

mod_gnutls was made possible and brought to you by the following contributors:

- Paul Querna (Original Developer)
- Edward Rudd (Paul's partner & collegue and Outoforder.cc Administrator)
- Nikos Mavrogiannopoulos (Previous Maintainer & Author of GNUTLS)
- Dash Shendy (Current Maintainer)