



VSI Apache Web Server (based on Apache HTTP Server)

Version 2.4-58A for OpenVMS Alpha and IA64, based on Apache 2.4.58

Release Notes

November 2023

VSI-AXPVMS-APACHEWS-V0204-58A-1.PCSI\$COMPRESSED

VSI-I64VMS-APACHEWS-V0204-58A-1.PCSI\$COMPRESSED

Introduction

VMS Software Inc. (VSI) are pleased to provide you with a new VSI-supported version of Secure Web Server for OpenVMS. This release of Secure Web Server for OpenVMS is based on Apache HTTP Server Version 2.4.58 from the Apache Software Foundation and represents a significant update from previous versions, providing many new features and numerous enhancements. Enhancements include reduced memory utilization and more flexible configuration, and there are a variety of new loadable modules providing new and enhanced functionality in areas such as session management, request filtering, and rate limiting. Secure Web Server for OpenVMS Version 2.4 also provides improved support for the development of custom loadable modules.

For a detailed description of new features and enhancements in Apache HTTP Server 2.4, please refer to http://httpd.apache.org/docs/2.4/new_features_2_4.html (but note that not all new features are provided in the version for VSI OpenVMS). For a description of changes and enhancements specific to the 2.4.58 release see https://downloads.apache.org/httpd/CHANGES_2.4.58.

Apache HTTP Server Documentation

For information about the Apache web server, see the Apache HTTP Server Version 2.4 documentation at <https://httpd.apache.org/docs/2.4/>.

After installing the APACHEWS on your OpenVMS system you can also view the web server documentation at <http://your.hostname/manual>, where “your.hostname” is that server host name (or IP address) and port number applicable to your installation.

Summary of New Features in Version 2.4-58A

This release of APACHEWS for OpenVMS is based on Apache HTTP Server Version 2.4.58 from the Apache Software Foundation and includes Secure Sockets Layer (SSL) `MOD_SSL` and OpenSSL based on OpenSSL 1.1.1p. Accordingly, this release of APACHEWS for OpenVMS supports higher levels of

encryption than those provided by previous versions, ensuring greater levels of security for clients connecting to your web server on OpenVMS. This release also includes various minor fixes as described in https://downloads.apache.org/httpd/CHANGES_2.4.54 and elsewhere in this document, along with the following new functionalities:

- The OpenVMS-specific authentication module `mod_authnz_openvms` can now optionally authenticate users (username and password) using the `SYS$ACMW` system service, thereby making it possible to authenticate OpenVMS users that are configured to use ACME LDAP or similar providers for user authentication. In order to enable this facility, it is necessary to define the logical name `APACHE$AUTH_USE_ACM` (to anything) in `LOGIN.COM` for the web server account.
- LDAP-related modules now use the new OpenLDAP client API, providing better security and enhanced functionality. It should be noted that these modules are statically linked with the OpenLDAP client libraries, and it is therefore not necessary to install OpenLDAP for VSI OpenVMS in order to make use of this new feature.
- The module `mod_socache_redis` is included in this release, allowing the high-performance Redis object cache to be used for caching various data such as SSL session information. Redis may be installed on the same OpenVMS system as the web server or reside on another server that is accessible from the system hosting the web server.
- The module `mod_wsgi` is included in this release, and can be used in conjunction with Python 3.10 for VSI OpenVMS to create powerful Python-based web applications using the WSGI (Web Server Gateway Interface) framework. It should be noted that the `mod_wsgi` module is available only for Integrity at this time.

For a full list of new features and new and changed modules in Apache HTTP Server Version 2.4.58, please see https://httpd.apache.org/docs/2.4/new_features_2_4.html. This release also addresses various security-related CVE's and other such matters. For a complete list of these changes, see http://www.apache.org/dist/httpd/CHANGES_2.4.

Note that not all of the new modules are provided with APACHEWS for OpenVMS. The list of modules provided with this release of APACHEWS is as follows:

- `mod_access_compat`
- `mod_actions`
- `mod_alias`
- `mod_allowmethods`
- `mod_asis`
- `mod_authnz_ldap`
- `mod_authnz_openvms`
- `mod_authn_anon`
- `mod_authn_core`
- `mod_authn_dbd`
- `mod_authn_dbm`
- `mod_authn_file`
- `mod_authn_socache`
- `mod_authz_core`
- `mod_authz_dbd`
- `mod_authz_dbm`
- `mod_authz_groupfile`

- mod_authz_host
- mod_authz_owner
- mod_authz_user
- mod_auth_basic
- mod_auth_digest
- mod_auth_form
- mod_autoindex
- mod_buffer
- mod_cache
- mod_cache_disk
- mod_cache_socache
- mod_cern_meta
- mod_cgi
- mod_charset_lite
- mod_dav
- mod_dav_fs
- mod_dbd
- mod_deflate
- mod_dir
- mod_dumpio
- mod_echo
- mod_env
- mod_expires
- mod_ext_filter
- mod_file_cache
- mod_filter
- mod_headers
- mod_include
- mod_info
- mod_isapi
- mod_lbmethod_bybusyness
- mod_lbmethod_byrequests
- mod_lbmethod_bytraffic
- mod_lbmethod_heartbeat
- mod_ldap
- mod_logio
- mod_log_config
- mod_log_debug
- mod_macro
- mod_mime
- mod_mime_magic
- mod_negotiation
- mod_osuscript
- mod_proxy
- mod_proxy_ajp
- mod_proxy_balancer

- `mod_proxy_connect`
- `mod_proxy_express`
- `mod_proxy_fcgi`
- `mod_proxy_ftp`
- `mod_proxy_http`
- `mod_proxy_scgi`
- `mod_proxy_wstunnel`
- `mod_ratelimit`
- `mod_remoteip`
- `mod_reqtimeout`
- `mod_request`
- `mod_rewrite`
- `mod_sed`
- `mod_session`
- `mod_session_cookie`
- `mod_session_dbd`
- `mod_setenvif`
- `mod_slotmem_shm`
- `mod_socache_dbm`
- `mod_socache_memcache`
- `mod_socache_redis`
- `mod_socache_shmcb`
- `mod_speling`
- `mod_ssl`
- `mod_status`
- `mod_substitute`
- `mod_suexec`
- `mod_unique_id`
- `mod_unixd`
- `mod_userdir`
- `mod_usertrack`
- `mod_version`
- `mod_vhost_alias`
- `mod_wsgi` (OpenVMS Integrity with Python 3.8.2F only)

For details of how to configure and use these modules, refer to the documentation provided on the Apache HTTP Server web site at <https://httpd.apache.org/docs/2.4/mod/>.

Changed Features

This section summarises important differences between this release of the APACHEWS for OpenVMS and previous versions. Previous versions of the APACHEWS were referred to as CSWS.

- Changes are required in `httpd.conf` when upgrading from previous versions
In APACHEWS Version 2.4, some dynamically loadable modules provided with previous releases are no longer available or are not loaded by default.

You must uncomment the modules in `httpd.conf` to load them. See the file `httpd-vms.conf` to load other modules.

- Removal of `AcceptMutex` and related directives

In pre-2.4 versions of Apache HTTPD, the `AcceptMutex` directive was used in `httpd.conf` to specify the method used by the web server to serialize multiple child processes accepting requests on network sockets. In version 2.4, the `AcceptMutex`, `LockFile`, `RewriteLock`, `SSLMutex`, `SSLStaplingMutex`, and `WatchdogMutexPath` directives have been replaced with a single `Mutex` directive.

With previous versions of the APACHEWS for OpenVMS, the value `vmsdlm` could be specified for `AcceptMutex` to instruct the web server to use the OpenVMS Distributed Lock Manager to coordinate access to network sockets and other shared resources. For version 2.4 of the APACHEWS for OpenVMS, the Distributed Lock Manager is always used to coordinate access to network sockets and does not need to be explicitly specified unless using the DBM or shared memory cache modules, `mod_socache_dbm` and `mod_socache_shmcb`, respectively.

Other permitted (non-default) values for the `Mutex` directive are:

- `sem`, which causes the web server to use semaphores for coordination of access to shared resources.
- `lock:/path/to/lockfile`, which instructs the web server to use file-based locking for coordination (where `/path/to/lockfile` is the directory where lock files will be created, specified in UNIX syntax).

Note that if no `Mutex` type is defined and you attempt to use any modules that require a `Mutex` to be defined, the web server will silently fail to start. It is therefore recommended that you always define a `Mutex`, even if you do not currently make use of it.

- Changes to OpenVMS authentication module

The authentication and authorization module used by Apache HTTP Server version 2.4 differs from that of previous versions. In version 2.4, it is necessary to register the specific authentication (or authorization) provider that you wish to use for a particular directory or location. This way, it is possible to configure and use different providers with different directories or locations.

The following example illustrates use of the OpenVMS authentication provider for the `"/test"` directory:

```
<Directory /test>
  Options FollowSymLinks
  AllowOverride AuthConfig
  AuthType Basic
  AuthName "OpenVMS authentication"
  AuthBasicProvider OpenVMS
  require valid-user
</Directory>
```

We have specified an `AuthBasicProvider` of `"OpenVMS"` (note that the name of the provider is case-sensitive). This causes the authentication infrastructure to use the OpenVMS password checking module (included in `mod_authnz_openvms.exe`).

Note that in order for this to work correctly, the following modules must be loaded:

- `mod_authn_core.exe`

- `mod_authz_core.exe`
- `mod_auth_basic.exe`
- `mod_authnz_openvms.exe`

The `mod_authz_core.exe` module is strictly only required for authorization (as opposed to authentication), but since `mod_authnz_openvms.exe` includes functionality to handle both authentication and authorisation, it is recommended to load `mod_authz_core.exe` whenever you use the `mod_authnz_openvms.exe` module.

It should also be noted that the OpenVMS authentication and authorization module now accepts no configuration commands (as a consequence of the changes in 2.4 affecting how authentication and authorization are handled, such commands are now superfluous). Specifically, the following directives have been removed:

- `AuthOpenVMSUser`
- `AuthOpenVMSGroup`

- Setting a value for `ServerName`

Upon starting the web server for the first time you see a message similar to the following (where `myhost.mydomain.com` is the value that the web server has determined for the server's fully qualified domain name):

```
"Could not reliably determine the server's fully qualified domain name, using
myhost.mydomain.com. Set the 'ServerName' directive globally to suppress this
message"
```

This message is informational and will not prevent the web server from starting; however, you may want to do as the message suggests and modify your `httpd.conf` file to explicitly specify a value for the `ServerName` directive in accordance with the notes included in `httpd.conf` (edit `httpd.conf` and search for `ServerName`). It is strongly recommended that you also specify a port number, as illustrated in `httpd.conf`, as this can prevent future issues if you plan to run multiple instances of the web server or if you plan to enable SSL, which uses a different port number (the value of `ServerName` in `httpd.conf` cannot be the same as that used by the SSL `VirtualHost` specified in `ssl.conf`).

- Logical names marked for deprecation

The logical names `APACHE$BG_PIPE_BUFFER_SIZE` and `APACHE$MB_PIPE_BUFFER_SIZE` have been marked for deprecation, and site-specific command procedures using these logical names should be modified to instead use the names `APR$BG_PIPE_BUFFER_SIZE` and `APR$MB_PIPE_BUFFER_SIZE`, respectively. Use of either name is supported by this release; however, the logical names with the `APACHE$` prefix will be removed from subsequent releases. This change has been made in order to better reflect the name of the software sub-system to which the logical names pertain.

- The logical name `APACHE$SSL_DBM_TYPE` has been deprecated

This logical name could be used in previous versions to define the DBM database manager to be used by the SSL session cache with `MOD_SSL`. `APACHEWS` Version 2.4 for OpenVMS only supports the SDBM database, therefore the logical name `APACHE$SSL_DBM_TYPE` is not required. However, for optimal performance, it is recommended that the shared memory cyclic buffer session cache is used. For additional information on setting up the SSL session cache, see https://httpd.apache.org/docs/2.4/mod/mod_ssl.html#sslsessioncache.

- All custom-written dynamically loaded modules must be rebuilt for Version 2.4

Most third-party modules designed for Version 2.x will otherwise work unchanged with the Apache HTTP Server version 2.4 (will only need to be recompiled); however, some modules may require changes. The document http://httpd.apache.org/docs/2.4/developer/new_api_2_4.html provides details of the API changes that have been made with 2.4. Also, see the notes about building custom modules elsewhere in this document (building modules has been made easier with this release and does not require developers to download the APACHEWS source code).

In addition to the above points that are largely specific to OpenVMS, be sure to review the document <http://httpd.apache.org/docs/2.4/upgrading.html> if you are upgrading from a previous version of the web server.

Known Problems and Restrictions in Version 2.4

- Do not attempt to use APACHEWS Version 2.4 with older APACHEWS optional kits

Do not use APACHEWS Version 2.4 with the following optional kits. Using these kits together causes a process crash.

- CSWS_PERL V2.1 or earlier
- CSWS_PHP V5.2-17A or earlier
- CSWS_JAVA (any)

VMS Software Inc. is working to provide updated versions of these optional kits that can be used with APACHEWS Version 2.4.

- Installing APACHEWS Version 2.4 on an ODS-2 volume may corrupt an existing installation

You must install the APACHEWS Version 2.4 kit only on an ODS-5 target volume; if you install this kit on an ODS-2 volume, the installation will fail.

- Language variant filename restrictions

Specify language variants on an OpenVMS system in the same way as you do on a UNIX system, using multiple dots in the filename. For example, the French variant of a filename is filename.html.fr.

- WebDAV database manager type restriction

WebDAV support requires the SDBM database manager type. SDBM is the default and only DBM supported in this release.

Defining an alternative DBM using the logical name `APACHE$DAV_DBM_TYPE` will cause an error to be logged and the web server will fail to start. Other DBM types may be supported by future releases.

- If suEXEC is enabled in the initial configuration, APACHEWS cannot add a node in a cluster environment

If you enable suEXEC during the initial configuration of APACHEWS or by using Option 4 (Manage suEXEC users) of the APACHEWS Configuration Menu, then Option 10 of the APACHEWS Configuration Menu (Add a node to APACHEWS in a cluster environment) will fail.

As a workaround, use Option 4 to disable suEXEC, then use Option 10 to add the node, and then use Option 4 to re-enable suEXEC.

- Option 2 in the `APACHE$MENU.COM` menu ("Create an Apache instance") fails under the following circumstances:

- Specifying a non-existent target directory fails with the following error when the directory `[.FOO]` does not exist:

```
Root Location: dev:[APACHE.SPECIFIC.FOO]
%SYSTEM-W-NOSUCHFILE, no such file \_DEV0:[APACHE.SPECIFIC]FOO.DIR\
%DCL-W-UNDSYM, undefined symbol - check validity and spelling \INDID\
%DCL-W-UNDSYM, undefined symbol - check validity and spelling \INDID\
```

- Creating an instance under a name other than `APACHE$WWW` fails with the following error:

```
[DDD MMM DD HH:MM:SS YYYY] [error] (13) permission denied: Unable to
create input file dev:[directory.[000000]APACHE$xyz.COM
```

- The "Require user" directive for user authorization must specify usernames in uppercase with the `mod_authnz_ovenvms` module.
- The `ScoreBoardFile` directive is ignored

A shared memory scoreboard is used by `APACHEWS` to facilitate sharing of information between parent and child processes. The `ScoreBoardFile` directive in `HTTPD.CONF` is intended to allow file-based shared memory to be used for this purpose on platforms that do not support the use of anonymous shared memory. VSI OpenVMS does support anonymous shared memory and changes made in this release to improve `APACHEWS` performance by reducing inter-process coordination requirements have required the `ScoreBoardFile` directive to be ignored.

- For certain configuration errors, `APACHEWS` can fail silently on start-up without logging any details of the problem to the error log file, making it problematical to diagnose the problem. In such situations it is often useful to temporarily disable `APACHEWS` shared process logging by defining the logical name `APACHE$SPL_DISABLED` to "TRUE" in `APACHE$ROOT:[000000]LOGIN.COM`. This will ensure that all error messages are flushed to the error log file before the process terminates. For optimal performance be sure to re-enable shared process logging once the problem has been resolved.
- In order for the web server to use IPv6, the two logical names `TCPIP$IPV6_STARTED` and `APACHE$CAN_USE_IPV6` must be defined in the `SYSTEM` table (the logical names can be defined with any value). Having only `TCPIP$IPV6_STARTED` is not sufficient.

Requirements

The kit you are receiving has been compiled and built to operate on the operating system versions listed below. While it is highly likely that you will have no problems installing and using the kit on systems running higher versions of the operating system, we cannot say for sure that you will be so fortunate if your system is running older versions.

- VSI OpenVMS 8.4-2L1 or higher (IA64), VSI OpenVMS 8.4-2L1 or higher (Alpha)
- VSI TCP/IP, HPE TCP/IP Services for OpenVMS, or MultiNet TCP/IP stack
- VSI SSL111 V1.1-1p or later

Note that `MOD_SSL` is dynamically linked with `SSL111` (OpenSSL), and `SSL111` must therefore be installed in order to use the `MOD_SSL` module. Be aware that this is a change from previous releases of `APACHEWS`, which had statically linked the OpenSSL libraries with `MOD_SSL`.

- If you wish to run Python applications under `APACHEWS` using `MOD_WSGI`, it should be noted that this facility is currently only supported for Python 3.8.2F for VSI OpenVMS. It is envisaged that future releases of `APACHEWS` and `MOD_WSGI` will support Python 3.10 and/or later.

Before Beginning the Installation

As noted previously, there have been a number of changes in `APACHEWS` 2.4 that may cause any existing configuration files (`httpd.conf` and `ssl.conf` for example) to be incompatible with the new version of the web server. If you are installing `APACHEWS` 2.4 on a system that does not currently have a previous version of `APACHEWS` installed then no additional action is required before installing the software; however, if you are upgrading from a previous version then it is strongly recommended that the following steps are taken:

1. Shut down `APACHEWS` if running

```
$ @SYS$STARTUP:APACHE$SHUTDOWN
```

2. Take a backup of any site-specific files that are in `APACHE$ROOT`: [000000...].

3. Uninstall any earlier version of `APACHEWS`:

```
$ PRODUCT REMOVE CSWS
```

```
The following product has been selected:
```

```
VSI I64VMS CSWS V2.4-54                               Layered Product
```

```
Do you want to continue? [YES]
```

```
The following product will be removed from destination:
```

```
VSI I64VMS CSWS V2.4-54                               DISK$I64SYS:[VMS$COMMON.]
```

```
Portion done: 0%
```

```
Deleting the Apache Htdocs & Icons directory trees will remove ALL user  
data stored within.
```

```
Delete the Apache Htdocs& Icons directory trees ? [NO]: YES
```

```
...10%...20%...30%...40%...50%...60%...70%...80%...90%...100%
```

```
The following product has been removed:
```

```
VSI I64VMS CSWS V2.4-54                               Layered Product
```

When prompted with the question "Delete the Apache Htdocs & Icons directory trees" be sure to answer "yes" in order to fully remove old versions of all documentation. Note that this process will not remove any site-specific files or customized configuration files.

4. Rename your existing (customized) web server configuration files.

Renaming the configuration files will allow the `APACHEWS` 2.4 installation process to create new initial versions of these files, to which you can then apply any customizations from your old configuration files (taking into consideration any of the differences between web server versions).

Note that after shutting down and uninstalling the web server, no Apache logical names such as `apache$root` or `apache$common` will be defined. In order to find and rename your customized configuration files you will instead need to take note of where the web server was installed (in the case shown above this would be `DISK$I64SYS:[VMS$COMMON.]`) and set default to the configuration directory accordingly.

For example:

```
$ SET DEFAULT DISK$I64SYS:[VMS$COMMON.APACHE.CONF]
```

Installing APACHEWS

VSI requires that you install APACHEWS 2.4 on an ODS-5 enabled disk.

Verify that the destination device is an ODS-5 volume by entering a command similar to the following (assuming `BORIS$DKA200` is the disk where you want to install APACHEWS):

```
$ show dev BORIS$DKA200:/full
```

```
Disk BORIS$DKA200:, device type HP EH0146FCBVB, is online, mounted, file-
oriented device, shareable, available to cluster, error logging is enabled.
.
.
.
Volume Status: ODS-5, subject to mount verification, protected subsystems
enabled, file high-water marking, write-through XFC caching enabled,
write-through XQP caching enabled, special files enabled.
```

Install the APACHEWS kit by entering the following command, where `BORIS$DKA200` is the name of the ODS-5 enabled disk where you want to install APACHEWS. Be sure that you manually removed any earlier version of APACHEWS before proceeding.

```
$ PRODUCT INSTALL APACHEWS/DEST=BORIS$DKA200:[000000]
```

For a detailed description of the features, you can request with the `PRODUCT INSTALL` command when starting an installation, see the POLYCENTER Software Installation Utility User's Guide.

As the installation procedure progresses, the system displays the information similar to the following output:

```
$ PRODUCT INSTALL APACHEWS/DEST=BORIS$DKA200:[000000]
```

The following product has been selected:

```
VSI I64VMS APACHEWS V2.4-58                      Layered Product [Installed]
```

Do you want to continue? [YES]

Configuration phase starting ...

You will be asked to choose options, if any, for each selected product and for any products that may be installed to satisfy software dependency requirements.

Configuring VSI I64VMS APACHEWS V2.4-58

VMS Software Inc. & The Apache Software Foundation.

* This product does not have any configuration options.

Execution phase starting ...

The following product will be installed to destination:

VSI I64VMS APACHEWS V2.4-58 DISK\$I64SYS:[VMS\$COMMON.]

Portion done: 0%...10%...20%...30%...50%...60%...70%...80%...90%...100%

The following product has been installed:

VSI I64VMS APACHEWS V2.4-58 Layered Product

VSI I64VMS APACHEWS V2.4-58

Post-installation tasks are required.

The OpenVMS Installation and Configuration Guide gives detailed directions. This information is a brief checklist.

Configure OpenVMS aspects of the web server by:

```
$ @SYS$MANAGER:APACHE$CONFIG
```

If the OpenVMS username APACHE\$WWW does not exist, you will be prompted to create that username. File ownerships are set to UIC [APACHE\$WWW], etc.

After configuration, start the web server manually by entering:

```
$ @SYS$STARTUP:APACHE$STARTUP
```

Check that neither SYLOGIN.COM nor the LOGIN.COM write any output to SYS\$OUTPUT:.. Look especially for a

```
$ SET TERMINAL/INQUIRE.
```

Start the web server at system boot time by adding the following lines to SYS\$MANAGER:SYSTARTUP_VMS.COM:

```
$ file := SYS$STARTUP:APACHE$STARTUP.COM
$ if f$search("''file'") .nes. "" then @'file'
```

Shutdown the Apache server at system shutdown time by adding the following lines to SYS\$MANAGER:SYSHUTDOWN.COM:

```
$ file := SYS$STARTUP:APACHE$SHUTDOWN.COM
$ if f$search("''file'") .nes. "" then @'file'
```

Test the installation using your favorite Web browser. Replace host.domain in the following URL (Uniform Resource Locator) with the information for the web server just installed, configured, and started.

URL <http://host.domain/> should display the standard introductory page from the Apache Software Foundation. This has the bold text "It Works!" If you do not see this page, check the release notes.

If you'd like to use secure connections then you'll need to create a server certificate. We recommend that you start by creating a 30 day self-signed certificate using the following certificate tool:

```
$ @APACHE$COMMON:[OPENSSL.COM]OPENSSL_AUTO_CERT.COM
```

Once the certificate has been created you'll need to uncomment the following directive in the APACHE\$COMMON:[CONF]HTTPD.CONF file to enable SSL.

```
Include /apache$root/conf/ssl.conf
```

Thank you for using the Secure Web Server.

Configuring the web server

After you have installed the APACHEWS, you are ready to configure it. The configuration tool ensures that a user account is available to run the server and that all of the files are owned by that user. It also allows the system manager flexibility in defining options for the installation.

APACHEWS 2.4 includes a simple configuration menu that allows you to choose configuration functions. All of the functions provided by the menu can be run through the menu or independently via individual command procedures.

To run the configuration menu, enter the following command:

```
$ @APACHE$COMMON:[000000]APACHE$MENU.COM
```

Following is an example of the configuration menu:

```
Apache$Menu
1. Configure the Secure Web Server
2. Create an Apache instance
3. Delete an Apache instance
4. Manage suEXEC users
5. Run OpenSSL Certificate tool
6. Convert directory tree to Stream_LF
7. Start up an Apache instance
8. Shut down an Apache instance
9. Show status of an Apache instance
10. Add a node to APACHEWS in a cluster environment
11. Exit

Enter Menu Choice:
```

The menu choices correspond to running the following procedures or commands from the DCL command line:

1. SYS\$MANAGER:APACHE\$CONFIG.COM
2. APACHE\$COMMON:[000000]APACHE\$CREATE_ROOT.COM
3. APACHE\$COMMON:[000000]APACHE\$DELETE_ROOT.COM
4. APACHE\$COMMON:[000000]APACHE\$MANAGE_SUEXEC.COM
5. APACHE\$COMMON:[000000]APACHE\$CERT_TOOL.COM
6. APACHE\$COMMON:[000000]APACHE\$CONVERT_STREAMLF.COM
7. SYS\$STARTUP:APACHE\$STARTUP.COM
8. SYS\$STARTUP:APACHE\$SHUTDOWN.COM
9. SHOW SYSTEM/PROCESS=APACHE\$tag
10. APACHE\$COMMON:[000000]APACHE\$ADDNODE.COM

For example, to perform a basic configuration and start a single instance of APACHEWS, you could proceed as follows:

```
$ @SYS$MANAGER:APACHE$CONFIG.COM
```

```
Secure Web Server for OpenVMS
                        [based on Apache]
```

```
This procedure helps you define the operating environment
required to run the Secure Web Server on this system.
```

To operate successfully, the server processes must have read access to the installed files and read-write access to certain other files and directories. It is recommended that you use this procedure to set the owner UIC on the APACHEWS files and directories to match the server.

You should do this each time the product is installed, but it only has to be done once for each installation on a cluster.

```
Set owner UIC on APACHEWS files? [YES] YES
```

```
Do you want to enable the impersonation features provided by suEXEC?
If so, the server will support running CGIs using specified usernames.
```

```
Enable suEXEC? [NO]
```

```
Setting ownership on files. This could take a minute or two. . . .
```

```
Disabling suEXEC configuration. This could take a minute or two. . . .
Configuration is complete. To start the server:
```

```
$ @SYS$STARTUP:APACHE$STARTUP.COM
```

Once you are satisfied that the web server is functioning correctly, you can re-apply any site-specific configuration details and restore any site-specific files from a previous installation or perform any of the functions provided by the configuration menu (or individual command procedures).

Enabling and disabling SSL

- To enable SSL

Generate a self-signed certificate, which is valid for 30 days. To do so, use the following certificate tool:

```
$ @APACHE$COMMON:[OPENSSL.COM]OPENSSL_AUTO_CERT.COM
```

Uncomment the following directive in the APACHE\$COMMON:[CONF]HTTPD.CONF file and restart the web server:

```
Include /apache$root/conf/ssl.conf
```

- Disabling SSL

To disable SSL, comment the following directive in the APACHE\$COMMON:[CONF]HTTPD.CONF file:

```
Include /apache$root/conf/ssl.conf
```

Building dynamically loadable modules

APACHEWS for OpenVMS is ported from the Apache HTTP Server and includes all of the standard Apache HTTP Server modules as well as some OpenVMS-specific functionality. The Apache HTTP Server architecture allows new modules to be added to the server at the following times:

- When the server is built
- Dynamically at run-time using the Apache HTTP Server Dynamic Shared Object (DSO) feature

On OpenVMS, the DSO function is performed by the `LIB$FIND_IMAGE_SYMBOL` run-time library routine. When the web server encounters a `LoadModule` directive in `httpd.conf`, it calls `LIB$FIND_IMAGE_SYMBOL` (via the C RTL `dlopen()` and `dlsym()` functions) to load a shareable image and to find the necessary universal symbols.

For example:

```
LoadModule mymod_module /apache$common/modules/mod_mymod.exe
```

This directive directs the web server to activate the shareable image `mod_mymod.exe` using the universal symbol "`mymod_module`" to locate the relevant module data structure describing the module's internal routine entry points.

A detailed description of the module architecture and how to develop your own custom modules (or to port existing third-party developed modules) is beyond the scope of this document; however, the following important points specific to OpenVMS should be noted:

1. Your C code must be compiled with the following compiler switches:

```
/POINTER_SIZE=32/DEFINE=( _USE_STD_STAT )/NAMES=(AS_IS,SHORTENED)
```

If the macro `_USE_STD_STAT` is not defined as illustrated above, the HTTP request structure passed by the web server into your module will not have the correct size and request structure fields will not be correctly aligned between the web server and your custom module code.

The names of universal symbols (function names and global variables) in the web server shareable images `APACHE$APR_SHR` and `APACHE$HTTPD_SHR` are case-sensitive and custom modules must therefore be compiled with `/NAMES=(AS_IS,SHORTENED)` and linked with appropriate use of the `CASE_SENSITIVE` linker option (see below).

2. With previous releases of APACHEWS it was necessary to have a copy of the APACHEWS code available in order to include the necessary C header files into your custom module project. This release includes the text library `APACHE$ROOT:[INCLUDE]APACHE$LIBRARY.TLB`, which includes all of the header files that are required when developing custom modules. This library can be used as follows when compiling your module code:

```
$ cc/pointer_size=32/define=( _USE_STD_STAT )/names=(as_is,shortened) -  
mod_mymod.c+apache$root:[include]APACHE$LIBRARY.TLB/lib
```

3. As commented above, the names of symbols in the shareable images `APACHE$APR_SHR` and `APACHE$HTTPD_SHR` are case-sensitive. It is necessary to link your custom code with these images, taking into consideration this case sensitivity, as illustrated below:

```
$ link/share mod_mymod.obj,sys$input/opt  
CASE_SENSITIVE=YES  
SYMBOL_VECTOR=(mymod_module=DATA)  
APACHE$APR_SHR/share  
APACHE$HTTPD_SHR/share
```

Note that modules are implemented as an OpenVMS shareable image and must therefore be linked with the `/SHARE` qualifier.

Your custom module shareable images must not contain any linker warnings or errors, otherwise they will not be properly loaded at run-time.

Known issues

- APACHEWS will fail to start correctly if the audit server is not running. This requirement may be removed in future releases of APACHEWS (CSWS).
- APACHEWS may fail to start correctly if a `Listen` directive is not specified in `httpd.conf`. The symptom of this problem is that the web server appears to be running but is not listening on any TCP/IP port for client requests. Inclusion of a simple `Listen` directive such as "`Listen 80`" (where 80 is the desired port number) is sufficient to negate this problem. This issue will be resolved in future releases.