

VSI Apache Web Server V2.4-71 (based on Apache HTTP Server)

Release Notes and Installation Guide

Publication Date: November 2026

Operating Systems: VSI OpenVMS x86-64 Version 9.2-3 or higher
VSI OpenVMS IA-64 Version 8.4-2L1 or higher
VSI OpenVMS Alpha Version 8.4-2L2 or higher

Software Version: VSI Apache Web Server V2.4-71

Kit Names: VSI-X86VMS-APACHEWS-V0204-71-1.PCSI
VSI-I64VMS-APACHEWS-V0204-71-1.PCSI
VSI-AXPVMS-APACHEWS-V0204-71-1.PCSI

VMS Software, Inc. (VSI)
Boston, Massachusetts, USA

Table of Contents

1. Introduction 3
    1.1. Requirements 3
    1.2. Apache HTTP Server Documentation 3
2. Release Notes 3
    2.1. What's New in VSI Apache Web Server V2.4-71 3
        2.1.1. Security Updates 5
    2.2. Features Introduced in Previous Releases 5
    2.3. Modules Provided With APACHEWS 5
    2.4. Known Issues and Restrictions 7
3. Pre-Installation Tasks 9
4. Installing APACHEWS 9
5. Post-Installation Tasks 11
    5.1. Configuring the Web Server 11
    5.2. Enabling and Disabling SSL 13
6. Developer Information 13
    6.1. Building Dynamically Loadable Modules 14

Appendix A. Migrating From Apache HTTP Server Version 2.2 to Version 2.4 15

1. Introduction

VSI is pleased to provide VSI Apache Web Server (APACHEWS) V2.4-71 for OpenVMS. This release is based on Apache HTTP Server Version 2.4.71 from the Apache Software Foundation.

Throughout this document, APACHEWS refers to the VSI-supported OpenVMS product, while Apache HTTP Server refers to the underlying open-source software on which APACHEWS is based.

1.1. 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 or products listed, we cannot guarantee functionality if your system is running older versions.

- VSI OpenVMS Alpha Version 8.4-2L2 or higher, VSI OpenVMS IA-64 Version 8.4-2L1 or higher, VSI OpenVMS x86-64 Version 9.2-3 or higher
- VSI TCP/IP, HPE TCP/IP Services for OpenVMS, or MultiNet TCP/IP stack
- VSI SSL3 V3.0-18 or later

  Note

  Note that MOD_SSL is dynamically linked with SSL3 (OpenSSL), and SSL3 must therefore be installed in order to use the module MOD_SSL. 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.

1.2. 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/. For a description of changes and enhancements specific to the 2.4.71 release, see https://www.apachelounge.com/changelog-2.4.html.

After installing 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.

2. Release Notes

VSI Apache Web Server (APACHEWS) V2.4-71 for OpenVMS is based on Apache HTTP Server Version 2.4.71 from the Apache Software Foundation and includes Secure Sockets Layer (SSL) MOD_SSL and OpenSSL based on OpenSSL 3.0.18.

The following sections summarize changes introduced in this release, features introduced in previous APACHEWS Version 2.4 releases, and known issues and restrictions that apply to this release.

2.1. What's New in VSI Apache Web Server V2.4-71

The following changes are introduced with the V2.4-71 release:

- Corrected an issue affecting OpenVMS authentication whereby certain authentication failures, such as invalid credentials or expired accounts, could prevent the login dialog from being redisplayed. Authentication failures are now handled correctly, allowing users to retry authentication.
- Corrected an issue that could cause an ACCVIO during LDAP authentication processing following a failed authentication attempt. LDAP authentication failures are now handled correctly without causing the server process to terminate.
- Enhanced the APACHE$CONFIG procedure to allow files and directory trees to be excluded when updating ownership information on Apache Web Server files and directories. This enhancement provides greater flexibility for sites that maintain locally managed files or customized configurations within the Apache directory structure.
- Updated APACHE$CONFIG to create the required APACHE$SPECIFIC directory structure while continuing to use the default configuration files supplied in APACHE$COMMON.
- Enhanced the installation procedure to verify that the required SSL3 software version is installed before proceeding. If the minimum supported SSL3 version is not present, the installation terminates with an error message similar to the following:

```
Minimum SSL3 software version not found on this system, abort installation
This kit requires a minimum SSL3 software version of V3.0-18.
Portion done: 0%
Recovery pass starting ...
Recovery pass concluded
%PCSI-E-S_OPFAIL, operation failed
%PCSIUI-E-ABORT, operation terminated due to an unrecoverable error condition
```

- Updated the installation information provided during product installation to include details about the APACHE$MENU configuration utility.
- Updated the definitions of the APACHE$COMMON and APACHE$SPECIFIC logical names to improve consistency for common installations. When Apache Web Server is installed in SYS$SYSDEVICE:[VMS$COMMON], these logical names now point to SYS$SYSDEVICE:[VMS$COMMON.APACHE.] and SYS$SYSDEVICE:[VMS$COMMON.APACHE.SPECIFIC.], respectively.
- Enhanced the APACHEWS product removal procedure to detect running APACHEWS instances in a CCSD environment. If APACHEWS is running on another node, the user is informed and the removal procedure terminates.
- Enhanced the APACHEWS product removal procedure to perform additional cleanup of APACHEWS files during deinstallation.
- Enhanced the APACHEWS product removal procedure to provide additional deinstallation options. Administrators can now choose whether to retain site-specific files, including configuration files and certificates, and can optionally remove the APACHE$WWW account from the SYSUAF database. For example:

```
Delete local user files (configs, certificates, etc.)? [NO]: YES
Delete the APACHE$WWW user from the UAF database? [NO]: YES
...10%...20%...30%...40%...50%...60%...70%...80%...90%...100%
The following product has been removed:
    VSI X86VMS APACHEWS V2.4-71            Layered Product
```

- Enhanced APACHEWS instance management procedures. Startup, shutdown, and removal operations now perform additional validation checks and provide informative status messages. Administrators must stop Apache instances created using APACHE$MENU before removing them and are responsible for cleaning up any associated logical names.

This release also includes the various fixes and enhancements introduced with Apache HTTP Server Version 2.4.71, as described at https://www.apachelounge.com/changelog-2.4.html.

2.1.1. Security Updates

This release includes Apache HTTP Server security corrections and vulnerability fixes. For detailed information, please visit https://httpd.apache.org/security/vulnerabilities_24.html.

2.2. Features Introduced in Previous Releases

The following features were introduced in previous APACHEWS Version 2.4 releases:

- The module MOD_PROXY was updated to support single-threaded builds.
- 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 IA-64 systems at this time.

For information about the features and modules provided by Apache HTTP Server Version 2.4, see https://httpd.apache.org/docs/2.4/new_features_2_4.html.

2.3. Modules Provided With APACHEWS

Note that not all of the Apache HTTP Server 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 [1]	

[1]OpenVMS IA-64 with Python 3.8.2F only.

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

2.4. Known Issues and Restrictions

The following known issues and restrictions apply to this release of APACHEWS Version 2.4:

- APACHEWS will fail to start correctly if the audit server is not running. This requirement may be removed in future releases of APACHEWS.
- 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.
- Do not attempt to use APACHEWS Version 2.4 with older CSWS 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 version)

  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 user names in uppercase with the MOD_AUTHNZ_OPENVMS 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.

3. Pre-Installation Tasks

Existing configuration files (such as HTTPD.CONF and SSL.CONF) may be incompatible with the version of APACHEWS being installed. If you are installing APACHEWS on a system that does not currently have a previous version installed, no additional action is required before installation. However, if you are upgrading from an earlier version, VSI strongly recommends taking the following steps:

1. Shut down APACHEWS if it is 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. For example:

   ```
   $ PRODUCT REMOVE APACHEWS
   The following product has been selected:
       VSI I64VMS APACHEWS V2.4-54               Layered Product

   Do you want to continue? [YES]

   The following product will be removed from destination:
       VSI I64VMS APACHEWS 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 APACHEWS V2.4-54               Layered Product
   ```

   When prompted with the question to delete the Apache Htdocs and 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]
   ```

4. Installing APACHEWS

VSI requires that you install APACHEWS Version 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 DEVICE 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, refer to the *VSI OpenVMS POLYCENTER Software Installation Utility Manual* [https://docs.vmssoftware.com/vsi-openvms-polycenter-software-installation-utility-manual/].

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

```
The following product will be installed to destination:
    VSI X86VMS APACHEWS V2.4-71            DISK$X86SYS:[VMS$COMMON.]

Portion done: 0%...10%...20%...30%...40%...50%...60%...70%...80%...90%...100%

The following product has been installed:
    VSI X86VMS APACHEWS V2.4-71            Layered Product

VSI X86VMS APACHEWS V2.4-71

    Release notes are available in SYS$HELP:APACHEWS_2_4_71.release_notes.


    VMS Software Inc. highly recommends that you read these release notes.


    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.

    To create node-specific configuration directories
    when running Apache on a system featuring more than one root,
    execute the APACHE$COMMON:[000000]APACHE$ADDNODE.COM command procedure
    and follow the recommendations displayed.
```

```
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:SYSHUTDWN.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 VSI Apache Web Server.
```

5. Post-Installation Tasks

This section describes the tasks that should be performed after installing APACHEWS. These include configuring the web server environment, enabling optional features, and verifying that the installation is operating correctly.

5.1. Configuring the Web Server

After you have installed 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 Version 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

                VSI Apache Web Server for OpenVMS

        This procedure helps you define the operating environment
        required to run the Apache 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]

Do you want to exclude files or directories? [YES]


You can specify directories or files from APACHE$ROOT to exclude
from this step.

Format for files: [dir_in_apache_root_directory]filename.
          [CONF]HTTPD.CONF

Format for directory tree: [dir_in_apache_root_directory...]*.*;*
          [SPECIFIC...]*.*;*
Enter path to exclude : [CONF]HTTPD.CONF
Enter path to exclude :
list: [CONF]HTTPD.CONF

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).

5.2. Enabling and Disabling SSL

- Enabling 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
  ```

6. Developer Information

APACHEWS supports the use of dynamically loadable modules to extend web server functionality. This section provides information for developers to build new modules or rebuild existing modules for use with APACHEWS Version 2.4.

6.1. 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.

A. Migrating From Apache HTTP Server Version 2.2 to Version 2.4

Apache HTTP Server Version 2.4 introduced a number of architectural, configuration, and compatibility changes relative to Version 2.2. The following notes describe those changes as they apply to APACHEWS for OpenVMS and highlight areas that may require updates to existing configurations, custom modules, or operational procedures.

- 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. To load other modules, see the file HTTPD-VMS.CONF.

- 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 (for details, refer to https://httpd.apache.org/docs/2.4/mod/core.html#mutex).

  With previous versions of 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 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

  Causes the web server to use semaphores for coordination of access to shared resources.

  flock:*/path/to/lockfile*

  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:

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

where *myhost.mydomain.com* is the value that the web server has determined for the server's fully qualified domain name)

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).

The above points are largely specific to APACHEWS for OpenVMS. For additional information about changes introduced with Apache HTTP Server Version 2.4, see http://httpd.apache.org/docs/2.4/upgrading.html.