

Compaq Secure Web Server V1.2 and V1.1-1 for OpenVMS Alpha (based on Apache)

Release Notes

Version 1.2, based on Apache 1.3.20
CPQ-AXPVMS-CSWS-V0102--1.PCSI

Contents:

[What's New](#)

[Compaq Secure Web Server Documentation](#)

[Apache Server Documentation](#)

[New Features](#)

[Corrected Problems and Workarounds](#)

[Additional Compaq Secure Web Server Notes](#)

[SSL Release Notes](#)

[Known Problems and Restrictions](#)

Compaq is pleased to provide you with Version 1.2, a new Compaq-supported, customer release version of *Compaq Secure Web Server for OpenVMS Alpha*. The Compaq Secure Web Server includes Secure Sockets Layer (SSL) through mod_ssl and OpenSSL.

What's New

Compaq Secure Web Server Version 1.2 is based on Apache 1.3.20 and contains several [new features](#), including mod_proxy, mod_rewrite, and a clusterwide SSL session cache, and corrections to several problems. With Version 1.2, you can install a new optional kit, [CSWS_PHP](#). PHP is a server-side, cross-platform, HTML embedded scripting language that lets you create dynamic web pages.

Important: See the [Software Patch Kits](#) page for information about **required security patch kits** for the *Compaq Secure Web Server*.

Version 1.1-1 is based on Apache 1.3.14, and [corrects problems](#) found in previous releases.

See the [Supported Configurations Page](#) for the prerequisites and supported products with version requirements for each release of Compaq Secure Web Server and its associated kits.

Version 1.1 included support for [multiple servers running on the same system](#), performance enhancements through the use of detached processes instead of subprocesses, a memory-based SSL session cache, loadable MOD_SSL, the ability to flush and re-open error and access logs without restart, and corrections to several problems.

Compaq Secure Web Server Documentation

See the [Documentation Page](#) for links to the *Installation and Configuration Guide*, *SSL User Guide*, as well as CSWS_PERL, CSWS_JAVA, and CSWS_PHP documentation.

Apache Server Documentation

Refer to the [Apache HTTP Server documentation](#) for information about the Apache server after you have completed the installation.

You can also view the online Apache server documentation on your web site at

`http://your.domain/manual`

Note: To view some of the Apache server documentation on your web site, you must enable MultiViews under `<Directory "/apache$common/htdocs">`.

New Features

Version 1.2

- Based on Apache 1.3.20 from the [Apache Software Foundation](#).

Previous versions of *Compaq Secure Web Server* were based on Apache 1.3.14 and 1.3.12.

- New optional kit: [CSWS_PHP](#)

PHP is a server-side, cross-platform, HTML embedded scripting language that lets you create dynamic web pages. PHP-enabled web pages are treated the same as regular HTML pages, and you can create and edit them the way you normally create regular HTML pages.

- New module: `mod_proxy`

[mod_proxy](#) provides for an HTTP 1.0 caching proxy server.

- New module: `mod_rewrite`

[mod_rewrite](#) provides a rule-based rewriting engine to rewrite requested URLs on the fly.

- Clusterwide SSL session cache

The SSL session cache holds state information across multiple SSL (HTTPS) connections. A session is the virtual connection between a particular browser and server. The first HTTPS request from the browser creates the session information. Subsequent HTTPS requests re-use that information to avoid the CPU overhead of setting up a new SSL connection.

Normal memory-based SSL session caches are server-specific. That is, the cache exists only in the server's local memory. This means that, in a cluster environment, the advantage of an SSL session cache is limited to HTTPS

connections directed to the same server. With a Galaxy cluster, on the other hand, you can define a memory partition as shared memory and place the SSL session cache there. That allows any server in the cluster to process an HTTPS request using a common, clusterwide SSL session cache. The cluster becomes the "server".

For non-Galaxy clusters, a clusterwide SSL session cache can be set up using a file-based session cache. This will provide the same functionality as a shared-memory cache, but at a significant cost in performance (file I/O versus memory access).

On OpenVMS Galaxy system clusters: When using clusterwide shared memory, each server in the cluster must specify the same values for the `SSLSessionCache` directive. For example:

```
SSLSessionCache      cshm:logs/ssl_scache(512000)
```

Clusterwide semaphores are available for OpenVMS system clusters. When using clusterwide semaphores, each server in the cluster must specify the same values for the **SSLMutex** directive. For example:

```
SSLMutex             csem:ClusterWideSem
```

Note: To enable clusterwide sharing of SSL session cache information in a non-Galaxy OpenVMS cluster, you can use a DBM file base session cache on a cluster mounted disk along with clusterwide semaphores.

Version 1.1-1

There are no new features in V1.1-1. This version includes corrected problems only.

Version 1.1

- Support for multiple servers and logical names defined processwide

Compaq Secure Web Server Version 1.1 includes support for multiple server processes running on the same system. This feature may be useful to you if your web server consistently experiences high load.

Because each server process can define its own logical names, these logical names cannot be defined systemwide.

In Version 1.1, the following logical names are defined **processwide** for the *Compaq Secure Web Server* server processes. (In previous versions, these logical names were defined systemwide.)

```
APACHE$SPECIFIC
APACHE$COMMON
APACHE$ROOT
```

See [Process Logical Names](#) in the *Compaq Secure Web Server for OpenVMS Alpha Installation and Configuration Guide* for more information

- READ function in APACHE\$CONFIG.COM

Compaq Secure Web Server Version 1.1 includes the new READ function to the APACHE\$CONFIG.COM procedure.

The READ command parses the specified configuration file and sets the APACHE\$SPECIFIC, APACHE\$COMMON, and APACHE\$ROOT processwide logical names to appropriate values for that configuration file.

See [Specifying the READ Function](#) in the *Compaq Secure Web Server for OpenVMS Alpha Installation and Configuration Guide* for more information.

- APACHE\$DCL_ENV -l command option

APACHE\$DCL_ENV now has a new command option "-l" that allows you to list the contents of the environment file. This new option is used to display the environment variables when you specify the APACHE\$DEBUG_DCL_CGI and the APACHE\$SHOW_CGI_SYMBOLS logical names.

Example output for APACHE\$DCL_ENV -l:

(CGI Environment Variables)

```
"DOCUMENT_ROOT" = "/apache$common/htdocs"  
"HTTP_ACCEPT" = "image/gif, image/x-xbitmap,  
image/jpeg, image/pjpeg, */*"  
"HTTP_ACCEPT_LANGUAGE" = "en-us"  
"HTTP_CONNECTION" = "Keep-Alive"  
"HTTP_HOST" = "node.com"  
...
```

- Flush and re-open error and access logs without restart

The Apache error log (error_log.) and transfer logs (access_log.) can now be manipulated while the server is running. The two new functions allow you to **flush** the logs to the disk and create **new** versions of the logs, closing the old log files.

You can invoke the functions through the APACHE\$CONFIG.COM command procedure, as follows:

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

Version 1.0-1

- Support for authentication using OpenVMS usernames and passwords

Compaq Secure Web Server Version 1.0-1 for OpenVMS Alpha included a new module, MOD_AUTH_OPENVMS, that supports authentication using the usernames and passwords contained in the system authorization file (SYS\$SYSTEM:SYSUAF.DAT). See MOD_AUTH_OPENVMS in the *Compaq Secure Web Server for OpenVMS Alpha Installation and Configuration Guide* for more information.

Corrected Problems and Workarounds

Version 1.2

- Keep-Alive sessions never time-out

Compaq Secure Web Server V1.2 for OpenVMS corrects a problem that prevented keep-alive sessions from timing out.

- APACHE\$CONFIG NEW/FLUSH in virtual host containers

The following commands flush and create new access logs or error logs:

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

In previous releases, these commands did not work for error log files and access log files specified inside virtual host containers. This problem has been corrected in *Compaq Secure Web Server V1.2 for OpenVMS*.

- First character of DCL command output missing

When using the server-side include **exec cmd**, the first character of DCL command output was missing. This problem occurred on systems running OpenVMS Versions 7.2-2 and earlier.

This problem has been corrected in *Compaq Secure Web Server V1.2 for OpenVMS*.

Version 1.1-1

- User directory pattern matching case-sensitivity

In previous releases, directory containers for user home directories were forced to use uppercase characters to refer to the users' device and directory components. Using lowercase characters caused the pattern matching to fail. For example, the following directive would fail to match the users' home directories on device "sys\$sysdevice":

```
UserDir public_html
<Directory /sys$sysdevice/*/public_html>
...
```

This problem is fixed in *Compaq Secure Web Server Version 1.1-1*. User's home device name and directory components are converted to lowercase (consistent with other path name directives). This requires that you use lowercase characters in path-names.

If you wish to preserve the old behavior of using uppercase characters for directives referring to user home directories, set the following logical name systemwide or set it inside the SYLOGIN.COM file for the *Compaq Secure Web Server* server process:

```
DEFINE APACHE$USER_HOME_PATH_UPPERCASE YES
```

- mod_osuscript "Location:" header

The following example OSU script does not work under mod_osuscript:

```
$ write net_link "<DNETCGI>"
$ write net_link f$fa0("Location:
http://<domain.name>/!//")
$ write net_link "</DNETCGI>"
$!
$ exit
```

This script should cause the server to issue a redirect to the URL specified, but it results in a "Document contains no data" error from Netscape and a blank page from IE.

This problem is corrected in *Compaq Secure Web Server Version 1.1-1*.

- Perl scripts under mod_osuscript strips first character of parameter

A Perl script run under mod_osuscript control with a URL that contains parameters does not get the full parameter name. The first character is stripped.

For example, if the URL is as follows:

```
<server>/<location>/perl_script.pl?File=file.typ&Mode=s
```

Script:

```
use CGI;

my $q = new CGI;
$q->import_names('Q');
$p1 = $Q::File;
$p2 = $Q::Mode;
```

\$p1 will be blank because the imported name becomes "\$Q::ile". The 'F' is stripped.

This problem is corrected in *Compaq Secure Web Server Version 1.1-1*.

- CGI script output buffer not flushed

In previous releases, CGI script output, which is buffered, was not sent to the browser until the buffer was filled or the script process terminated. *Compaq Secure Web Server Version 1.1-1* corrects this problem by flushing the buffer whenever the CGI script process pauses while sending output. This behavior is only supported for HTTP connections. HTTPS (HTTP over SSL) connections cannot be flushed in this fashion because of SSL protocol requirements.

- SSL compute-bound processes

Previous versions of *Compaq Secure Web Server* contained problems with SSL cache naming and consistency in single and multiple server configurations. These problems could have caused an Apache child server process to become compute-bound in a very tight processing loop. The *Compaq Secure Web Server Version 1.1-1* corrects these problems.

Version 1.1

Σ Script name always first argument for CGI parameters in URL

In previous versions of *Compaq Secure Web Server*, the script name was the first argument for any CGI that specified parameters in the URL. Fixing this bug may cause problems if you coded around the problem in existing scripts.

If you coded to the previous behavior and want to maintain that behavior, define the following logical name systemwide or in APACHE\$WWW account's LOGIN.COM:

```
APACHE$CGI_ARG1_AS_SCRIPT_NAME
```

This behavior is not dynamic and will provide the previous behavior for the duration of active servers.

Following is an example of the previous behavior, where the script name is always the first argument:

```
http://node.com/cgi-bin/test.com;?1+2+3
```

```
Argument 1: test.com;
```

```
Argument 2: 1
```

```
Argument 3: 2
```

```
Argument 4: 3
```

```
http://node.com/cgi-bin/test.exe;?1+2+3
```

```
Argument 0: node$dka0:[sys0.syscommon.apache.][cgi-bin]test.exe;1
```

```
Argument 1: test.exe;
```

```
Argument 2: 1
```

```
Argument 3: 2
```

```
Argument 4: 3
```

In Version 1.1, the argument information is now corrected as follows:

```
http://node.com/cgi-bin/test.com;?1+2+3
```

```
Argument 1: 1
```

Argument 2: 2
Argument 3: 3
http://node.com/cgi-bin/test.exe;?1+2+3
Argument 0: node\$dkka0:[sys0.syscommon.apache.][cgi-bin]test.exe;1
Argument 1: 1
Argument 2: 2
Argument 3: 3

Version 1.0-1

Σ Initial configuration data file not created

The following problem has been corrected in *Compaq Secure Web Server Version 1.0-1 for OpenVMS Alpha*.

In Version 1.0 of *Compaq Secure Web Server*, if you did not previously install a beta kit of the *Compaq Secure Web Server* or *Apache Web Server for OpenVMS*, the initial configuration data file required to run the server was not created when you installed the kit.

Σ Large binary file transfer using CGI

The following problem has been corrected in *Compaq Secure Web Server Version 1.0-1 for OpenVMS Alpha*.

In previous versions of *Compaq Secure Web Server* and *Apache for OpenVMS*, a CGI application could not transmit binary files larger than 32K bytes because of a problem with embedded Carriage Return / Line Feed pairs, even if the application attempted to transmit the file in multiple segments containing fewer than 32K bytes.

Note: In Version 1.0-1 and higher, the maximum amount of data you can transfer *in each write operation* is approximately 32K bytes. If your binary file is larger than 32K bytes, transfer the file using multiple write operations of 32K bytes each.

Compaq Secure Web Server supports the transfer of binary files larger than 32K bytes in multiple write operations if the following conditions are met:

- The device characteristics of SYS\$OUTPUT (or SYS\$INPUT) are set correctly.
- Your CGI application opens SYS\$OUTPUT (or SYS\$INPUT) using the proper parameters.

There are two ways to properly set the device characteristics of SYS\$OUTPUT (or SYS\$INPUT in the case of PUT/POST operations), as follows:

1. **Execute APACHE\$FLIP_CCL to set the device characteristics of SYS\$OUTPUT to support large binary file transfers.**

APACHE\$FLIP_CCL disables carriage control on the output device that is required for transferring binary data. APACHE\$FLIP_CCL is a symbol that executes APACHE\$ROOT:[000000]APACHE\$FLIP_CCL.EXE_ALPHA.

If your CGI application is run from within a command file, execute APACHE\$FLIP_CCL before you execute your CGI application. For example:

```
$ !CGI command file
$ APACHE$FLIP_CCL
$ run MYCGIAPPLICATION
```

If you use DCL commands in your command file to write the HTTP header prior to running your CGI application to transfer binary data, execute APACHE\$FLIP_CCL before you write the header. For example:

```
$ !CGI command file
$ APACHE$FLIP_CCL
$ write sys$output f$fao("!AS!//", "Content-type:
image/jpeg")
$ run MYCGIAPPLICATION
```

If you have not written a CGI application to transfer data, Compaq provides APACHE\$DCL_BIN as a convenient tool for transferring binary files. (You may be able to use this image instead of writing your own.) The symbol APACHE\$DCL_BIN executes the image. The command line syntax for APACHE\$DCL_BIN is as follows:

```
$ APACHE$DCL_BIN [-s bin-size] bin-file
```

where *-s bin-size* is an optional parameter that specifies the size in bytes of the binary file, and *bin-file* is the name of the binary file. If you do not specify *bin-size*, APACHE\$DCL_BIN computes the size of the binary file.

The following command file is an example of how to transfer binary files using APACHE\$DCL_BIN.

```
$ !CGI command file using APACHE$DCL_BIN
$ APACHE$FLIP_CCL
$ write sys$output f$fao("!AS!//", "Content-type:
image/jpeg")
$ APACHE$DCL_BIN myjpegfile.jpg
```

As stated previously, APACHE\$FLIP_CCL disables carriage control on the output device that is required for transferring binary data. However, carriage control is required when generating HTTP headers. To generate the headers correctly, you must provide explicit Carriage Return / Line Feed pairs after each header, as follows:

- o If you are writing the headers from a command file, use the F\$FA0 lexical function as shown in the preceding example.

o If you are writing the headers from your application, the new line escape sequence "\n" in your write statement generates the proper carriage control characters.

Regardless of which way you choose to set the device characteristics, your CGI application must open SYS\$OUTPUT in binary mode in order to successfully transfer binary files. An example of an fopen() call you can use is as follows:

```
outfile = fopen("SYS$OUTPUT", "wb", "rat=none", "rfm=stm",  
"ctx=bin");
```

2. Call APACHE\$FIXBG() from within your CGI application to set the device characteristics of SYS\$OUTPUT to support large binary file transfers.

APACHE\$FIXBG() is provided in the APACHE\$FIXBG.EXE shareable image. The following C code is provided as an example of how to call APACHE\$FIXBG().

```
#include <descrip.h>  
#include <ssdef.h>  
#include <starlet.h>  
extern int APACHE$FIXBG(short int, int);  
$DESCRIPTOR(output_file,"SYS$OUTPUT");  
unsigned short stdout_sock;  
int ret_stat;  
ret_stat = SYS$ASSIGN(&output_file,&stdout_sock, 0, 0);  
if (ret_stat == SS$_NORMAL)  
ret_stat = APACHE$FIXBG(stdout_sock,1);
```

Link your CGI application against the APACHE\$FIXBG shareable image using the following command in your linker options file:

```
APACHE$FIXBG/share
```

Σ MOD_PERL initializes the process logical name table between HTTP requests (Corrected in CSWS_PERL V1.0-1)

The following problem has been corrected in CSWS_PERL for *Compaq Secure Web Server V1.0-1 for OpenVMS Alpha*.

In previous versions of *Compaq Secure Web Server*, if installed, MOD_PERL ran clean-up code that initialized the process logical name table between HTTP requests, even if Perl scripts were not actually executed. This means that CGI scripts could not rely on logical names defined in the process logical name table by LOGIN.COM, SYLOGIN.COM, or other means.

Σ Inability to use ACLs to access CGI scripts

The following problem has been corrected in *Compaq Secure Web Server Version 1.0-1 for OpenVMS Alpha*.

In previous versions of *Compaq Secure Web Server* and *Apache for OpenVMS*, the server restricted CGI execution to script files owned by the server process (APACHE\$WWW). The server did not honor access control list (ACL) entries that granted read access to APACHE\$WWW. Any attempt to run a CGI script not owned by APACHE\$WWW resulted in this browser message:

```
"Forbidden
You don't have permission to access <script-name> on this
server."
```

or, if the URL did not contain the file extension, the error was as follows:

```
"Not Found
The requested URL <script-name> was not found on this
server."
```

Version 1.0-1 processes CGI scripts for which the server has been granted read access, either through UIC-based SOGW protection or an ACL entry. Read access applies not only to the script file, but also to any directory along the path.

To grant read access via ACL, use the following DCL command:

```
$ set security/acl=(ident=apache$www,access=read) <file-
spec>
```

Σ Extra blank line required after HTTP headers in CGI scripts

The following problem has been corrected in *Compaq Secure Web Server Version 1.0-1 for OpenVMS Alpha*.

In previous versions of *Compaq Secure Web Server* and *Apache for OpenVMS*, HTTP headers generated by CGI scripts sometimes required an extra blank line following the HTTP headers in order to be processed correctly. This was most often observed following a "Location" header, such as:

```
$ write sys$output "Location: http://new.html"
$ write sys$output ""
$ write sys$output ""
```

In Version 1.0-1, the extra line is no longer necessary.

Σ Server process falls into HIB state

The following problem has been corrected in *Compaq Secure Web Server Version 1.0-1 for OpenVMS Alpha*.

In previous versions of *Compaq Secure Web Server*, on heavily loaded systems running CGI scripts, occasionally a server process fell into HIB state and ceased processing.

Σ Workaround: Server-side include (SSI) "exec cmd"

When writing server-side include (SSI) `exec cmd` commands, the initial write to `sys$output` inserts a `<NULL><LF>` character sequence at the beginning of the data stream.

The `<NULL>` character may cause unpredictable behavior in some browsers. In particular, Netscape browsers ignore data between the `<NULL>` character and the next HTML tag or until the end of the TCP/IP data segment received from the server. The browser either displays no data at all or displays only the tail end of the data.

There are several workarounds to this problem, but each requires that the target of the `exec cmd` command be a DCL procedure.

Example SSI script:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
<HTML>
<HEAD>
<TITLE>Test server-side include</TITLE>
</HEAD>
<BODY>
<!--#exec cmd="$@apache$Root:[000000]test.com"--> <BR>
</BODY>
</HTML>
```

APACHE\$ROOT:[000000]TEST.COM:

\$! The following example eliminates the `<NULL><LF>` in the first

\$! record sent to `sys$output` by temporarily turning off CCL and then

\$! generating a line of output text which includes the `<PRE>` tag.

```
$!
$ apache$flip_ccl
$ write sys$output "<PRE>"
$ apache$flip_ccl
$ show system
$ write sys$output "</PRE>"
$ exit
$!
```

\$! This section includes the `<NULL><LF>`, but is followed by an

\$! HTML tag which allows Netscape to resume HTML processing after

\$! the `<NULL>` character.

```
$!
$ write sys$output "<PRE>"
$ show system
$ write sys$output "</PRE>"
$ exit
$!
```

\$! This section permanently disables carriage-control and \$! performs its own CR/LF processing by writing output to \$! a file and reading it back to format it with HTML tags.

```
$!
```

```

$ apache$flip_ccl
$ tmpfil = "APACHE$EXEC_CMD_" + f$getjpi("", "pid") +
".TMP"
$ define/user sys$output 'tmpfil'
$ show system
$ open x 'tmpfil'
$ read/end=end_it/error=end_it x y
$ write sys$output "<BOLD>"
$ write sys$output f$fao("!AS<BR>", y)
$ write sys$output "</BOLD>"
$ write sys$output "<PRE>"
$loop:
$ read/end=end_it/error=end_it x y
$ write sys$output f$fao("!AS!/", y)
$ goto loop
$! type apache$root:[000000]test.txt
$end_it:
$ close x
$ write sys$output "</PRE>"
$ delete 'tmpfil';*
$ exit

```

Σ Workaround: Running CGI scripts

When you use .COM scripts with the POST method, you must read from APACHE\$INPUT, not SYSS\$INPUT. See the file APACHE\$ROOT:[CGI-BIN]TEST-CGI-VMS.COM for an example.

Additional Compaq Secure Web Server Notes

- Support discontinued on OpenVMS Version 7.1-2 and earlier

Beginning with *Compaq Secure Web Server V1.2 for OpenVMS*, OpenVMS Alpha Version 7.2-1 or later is a required software prerequisite.

Note: Although *Compaq Secure Web Server* can be installed and run on OpenVMS Version 7.1-2 systems, the official end of support date for Version 7.1-2 is December 31, 2001. Customers are encouraged to upgrade to a later version of OpenVMS, such as Version 7.2-1 or Version 7.3.

- Running MOD_OSUSCRIPT with Compaq Secure Web Server Version 1.1 for OpenVMS

The *Compaq Secure Web Server for OpenVMS Alpha* provides a CGI script environment. However, it also includes MOD_OSUSCRIPT, an optional module that enables the server to run scripts that were written for the OSU http server's script environment (which is not CGI).

In Compaq Secure Web Server Version 1.1 for OpenVMS Alpha and higher, the Apache logical names must be defined **systemwide** (not processwide) in order for MOD_OSUSCRIPT to work properly.

See [Running MOD_OSUSCRIPT](#) in the *Installation and Configuration Guide* for more information.

- Location of the Listen 80 directive changed in Compaq Secure Web Server Version 1.1 for OpenVMS

In the *Compaq Secure Web Server* Version 1.1 kit, the location of the Listen 80 directive has been moved from the MOD_SSL.CONF file to the HTTPD.CONF file.

This could cause a problem in your configuration of *Compaq Secure Web Server Version 1.1* if you use one configuration file from a previous version and one new V1.1 configuration file.

If neither configuration file includes a Listen 80 directive, requests to HTTP:// will not work because the *Compaq Secure Web Server* is not listening on port 80. If both configuration files include the Listen 80 directive, the *Compaq Secure Web Server* will fail to start, and the log file will contain an entry similar to the following:

```
[Tue Mar 13 17:41:39 2001] [crit] (48)address already in
use :
make_sock: could not bind to port 80
```

If you use a configuration file (MOD_SSL.CONF or HTTPD.CONF) from a previous version of *Compaq Secure Web Server*, the workaround is to manually edit the files so that the Listen 80 directive is included in HTTPD.CONF and not in MOD_SSL.CONF.

- Apply CRTL patch to fix problem with root logical names

On OpenVMS Alpha Version 7.2, a problem in the Compaq C Run-Time Library prevents the *Compaq Secure Web Server* from correctly locating index.html files in top-level document roots with concealed logical names. An example of a top-level document root with a concealed logical name is /web_root/000000 (where *web_root* is defined as *ddcu:[directory.]*).

The C RTL patch kit (ECO) for OpenVMS Version 7.2 (VMS72_ACRTL-V0100) corrects this problem. VMS72_ACRTL-V0100 is available from the [Compaq support website](#).

- Internet Explorer forces download of script with URL ending in .COM

Microsoft Internet Explorer treats the content for a URL ending in .COM as an executable image and pops up the "File Download" dialogue box. For example, Internet Explorer treats the URL <http://hostname/cgi-bin/test-cgi-vms.com> as an image and does not display its contents, even though this CGI script is returning "text/plain" content.

To work around this problem, add a semi-colon (;) to the end of the URL or eliminate the .COM file extension entirely.

This problem does not occur with Netscape browsers.

- Access to CGI scripts

The *Compaq Secure Web Server for OpenVMS* allows access to a particular CGI script by script name (without the .COM or .EXE extension) or with a fully-specified script name.

If no extension is specified, the *Compaq Secure Web Server* searches for a CGI script in the following order: *script-name*, *script-name.COM*, *script-name.EXE*.

- Using MOD_NEGOTIATION on OpenVMS

The Apache module MOD_NEGOTIATION allows you to specify language variants of HTML files. For example, `filename.html.fr` is the French variant of `filename.html`.

To specify language variants using the *Compaq Secure Web Server for OpenVMS*, use an underscore instead of a period before the language tag, as follows:

```
filename.ext_tag
```

For example, use `INDEX.HTML_EN` instead of `INDEX.HTML.EN`.

- Error %IMGACT-F-SYMVECMIS

If you receive this error, apply the most current DEC C RTL ECO from the [Compaq support website](#).

SSL Release Notes

- Delete the temporary, self-signed certificate

After you install a real certificate, delete the temporary, self-signed certificate (`APACHE$SPECIFIC:[CONF.SSL_CERT]SERVER.CRT`) that was created during the installation of the *Compaq Secure Web Server*. This prevents the accidental use of the temporary certificate if you installed the real certificate in `APACHE$COMMON:[CONF.SSL_CERT]` using the same name, and your `MOD_SSL` .CONF directive uses `APACHE$ROOT` as part of the certificate file path. For example:

```
SSLCertificateFile /apache$root/conf/ssl_cert/server.crt
```

Because `APACHE$ROOT` is a search-listed logical name, the server first searches `APACHE$SPECIFIC:[CONF.SSL_CERT]` and then searches `APACHE$COMMON:[CONF.SSL_CERT]` for the `SERVER.CRT` file. If you used the same name as the temporary certificate file, the server will find the temporary file first.

- Problem corrected in V1.0-1: Inability to import client certificates into NetscapeÆCommunicator or MicrosoftÆInternet Explorer

The following problem has been corrected in *Compaq Secure Web Server Version 1.0-1 for OpenVMS Alpha*.

In Version 1.0, client certificates converted from PEM to PKCS12 format with the OpenSSL utility sometimes could not be imported into Netscape and Microsoft web browsers.

Netscape Communicator issued the following error message:

```
"Unable to import certificates. The file specified is
either corrupt or is not a valid file."
```

Internet Explorer issued the following error message:

```
"Invalid password error."
```

Using the OpenSSL utility to analyze the .P12 file resulted in an error message similar to the following:

```
$ openssl pkcs12 -in apache$root:[openssl.crt]test3.p12
Enter Import Password:
MAC verified OK
Error outputting keys and certificates
172:error:06065064:digital envelope
    routines:EVP_DecryptFinal:baddecrypt:EVP_ENC:248:
172:error:23077074:PKCS12
    routines:PKCS12_pbe_crypt:pkcs12
cipherfinalerror:P12_DECR:95:
172:error:2306A075:PKCS12
    routines:PKCS12_decrypt_d2i:pkcs12 pbe
crypterror:P12_DECR:121:
$
```

- Restriction on key length for 56-bit encryption web browsers

If you are using a version of Netscape Communicator or Microsoft Internet Explorer with 56-bit encryption, the key length of the RSA public-key in the client certificate should be no longer than 512 bits.

- Legal caution

SSL data transport requires encryption. Many governments, including the United States, have restrictions on the import and export of cryptographic algorithms. Please ensure that your use of SSL is in compliance with all national and international laws that apply to you.

- Understand security issues

Successfully implementing SSL requires more than an SSL-aware web server. Seek expert advice when setting up secure connections with clients.

- 30-day certificate expiration

After installing *Compaq Secure Web Server*, when you run the configuration utility (APACHE\$CONFIG.COM) and enable SSL, it creates and installs a self-signed server certificate. This is valid for 30 days only and must be replaced,

preferably with a commercial CA certificate. *Compaq Secure Web Server* will not run without a server certificate that is valid for your system.

- Changing MOD_SSL directives

Always make changes to mod_ssl directives in the MOD_SSL.CONF include file. Do not add mod_ssl directives to HTTPD.CONF. You must stop and restart the *Compaq Secure Web Server* for any change you make in MOD_SSL.CONF to take effect.

- <Location> container

The <Location> container statement (which provides for access control by URL) is not supported by mod_ssl directives (although its use in other contexts is permitted in HTTPD.CONF).

- Using the Certificate Tool

Completion of all fields is mandatory when using the OpenSSL Certificate Tool options. Required information that is omitted will prevent certificates from being generated or create invalid certificates.

- Common name usage

When creating server certificate requests, the common name must be the same as your server's DNS host name (or virtual host name, if name-based virtual hosting is used).

- Initializing command-line OpenSSL

Before using command-line OpenSSL, you must run the following command to initialize the environment:

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

- Signing client certificates

When signing a client certificate you must use the same pass phrase you used to create your certificate authority.

- PKCS12 export format

In order to serve PKCS12 client certificates correctly to Netscape users, you need to define this file type in the MOD_SSL.CONF file:

```
AddType application/octet-stream .p12
```

When distributing client certificates signed by your own CA, clients must load both the client certificate and your CA certificate in their browser. (The password used when converting the client certificate to PKCS12 format is the same one used by clients to install the certificate.)

- Use semaphore for SSLMutex

When using the **SSLMutex** `mod_ssl` directive, use the default system semaphore-caching type (`SSLMutex sem`). Mutex file locking (`SSLMutex file`) will reduce system performance or, in some cases, result in hung HTTP or HTTPS connections.

- Shared SSLMutex and SSLSessionCache resources among servers are supported

All servers must specify the same **SSLMutex** directive. If different **SSLMutex** directives are used, access to the SSL session cache will not be synchronized properly.

- Use builtin for SSLRandomSeed

When using the **SSLRandomSeed** `mod_ssl` directive, use the default builtin source (`SSLRandomSeed builtin`). Using another source will prevent your server from starting.

- Caution changing SSL directives in .HTACCESS files

When modifying your SSL configuration in .HTACCESS files, be aware that making certain changes when using indexing via a secure connection can lead to unexpected results. For example, if you change the **SSLCipherSuite** to *high* within a directory tree viewed via indexing, the system returns an error for a *low* encryption-capable browser. This occurs even when the directory containing the .HTACCESS file is not visible at the current level of the index. The reason this occurs is that during indexing, directories are processed along with any .HTACCESS files. If the .HTACCESS file includes SSL directives that may force a renegotiation of the handshake (for example, **SSLCipherSuite**), errors may occur unexpectedly.

Known Problems and Restrictions

NOTE: This list is a summary and does not contain all known problems in the *Compaq Secure Web Server*.

- CGI scripts may fail when running TCPware from Process Software

If you are running TCPware from Process Software, CGI scripts may fail with the following entry in the error log file:

```
18-Jan-2002 14:20:50 [000007C0] GENERIC_SOCKETPAIR_inet:
SYS$ASSIGN() -2312 [Thu Jan 18 14:20:50 2002]
[error] [client a.b.c.d] couldn't spawn child process:
/apache$common/cgi-bin/test.cgi
```

To avoid this problem, add the following line to your SYSTARTUP_VMS.COM procedure:

```
$ define/system tcpip$device ucx$device
```

- Problem with PROXY-CACHE file cleanup

The **CacheGcInterval** directive does not work correctly when you select the caching option under MOD_PROXY. The workaround for this problem is to periodically delete the files in the CACHE directory if the size becomes unmanageable.

This problem will be corrected in a future release of *Compaq Secure Web Server*.

- **User** directive not supported

The **User** directive cannot be used to change the user profile of the web server processes in this version of *Compaq Secure Web Server*. All web server processes run under the APACHE\$WWW user profile. Compaq may add support for the **User** directive in the future.

- **ProxyPass** directive not supported inside a virtual container

The **ProxyPass** directive in mod_proxy is not supported inside a virtual container.

- DCL SEARCH command causes fatal error in CGI scripts

The DCL SEARCH utility is not compatible with CGI scripts when the SEARCH command accesses more than one file. For example:

```
$ SEARCH *.HTML "Compaq"
```

In this case, the SEARCH utility closes and re-opens SYS\$OUTPUT on every file access. In a CGI script, this tears down the output stream and yields the following error:

```
%SEARCH-F-WRITEERR, error writing SYS$OUTPUT:.;  
-RMS-F-WER, file write error  
-SYSTEM-F-BADPARAM, bad parameter value
```

The workaround for this problem is to output the search results to a temporary file, and then type the contents of the output file. For example:

```
$ pid = f$getjpi("", "PID") !Get PID to generate unique name  
$ search *.html "Compaq"/output=SRCH_'pid'.TMP  
$ type SRCH_'pid'.TMP  
$ delete SRCH_'pid'.TMP;*
```

- DCL extended parse style not supported

Compaq Secure Web Server does not support the DCL extended parse style feature. Extended parse style allows additional DCL syntax for ODS-5 extended file specifications and case-preserves foreign command arguments.

If extended parse style is enabled, the *Compaq Secure Web Server* startup will fail with the following error:

```
%SYSTEM-F-NOLOGTAB, no logical name table name match
```

You can disable DCL extended parse style for the *Compaq Secure Web Server* by adding the following line to APACHE\$ROOT:[000000]LOGIN.COM:

```
$ SET PROCESS/PARSE_STYLE=TRADITIONAL
```

- Limited support for ODS-5 disks

Compaq Secure Web Server for OpenVMS has limited support for the OpenVMS Extended File System (EFS, ODS-5). You can install the *Compaq Secure Web Server* on an EFS disk. ODS-2 features and deeply nested directories work properly. However, EFS-specific features such as multiple periods (dots) in file names and extended characters do not work properly.

- Overwritten ERROR_LOG and ACCESS_LOG entries

Because several processes write to APACHE\$ROOT:[LOGS]ERROR_LOG. and ACCESS_LOG., entries from one process can occur in the middle of a line written from another.
