CSWS_PHP for HP Secure Web Server for OpenVMS (based on Apache)

Installation Guide and Release Notes

November 2005

CSWS_PHP V1.3 for OpenVMS Alpha, based on PHP 4.3.10
CPQ-AXPVMS-CSWS_PHP-V0103--1.PCSI

CSWS_PHP V1.3 for OpenVMS I64, based on PHP 4.3.10
HP-I64VMS-CSWS_PHP-V0103--1.PCSI

Contents

What's New

Two new kits for CSWS_PHP V1.3 are now available – one for OpenVMS I64 Version 8.2 and higher, and one for OpenVMS Alpha 7.3-2 and higher. For more information about PHP, see http://www.php.net/.

New features included in CSWS_PHP **V1.3** are as follows:

- CSWS_PHP V1.3 is intended to work with the **Secure Web Server Version 2.1 and 1.3-1**. It does not work with SWS V2.0.

- **Based on PHP 4.3.10**. (CSWS_PHP V1.2-1 was based on PHP 4.3.2.)

PHP 4.3.10 includes over 30 non-critical bug fixes and addresses several serious security issues. See http://www.php.net/release_4_3_10.php for more information.

- **PHP_GD extension**, which allows you to create and manipulate image files in a variety of different image formats, including gif, png, jpg, wbmp, and xpm. PHP can also output image streams directly to a browser. See http://us2.php.net/manual/en/ref.image.php for more information.

New features included in CSWS_PHP **V1.2-1** are as follows:

- CSWS_PHP V1.2-1 works with the **Secure Web Server Version V1.3-1.**
- Based on **PHP 4.3.2**.
- Includes **MySQL extension**, which allows you to access MySQL database servers. More information about MySQL can be found at http://www.mysql.com/.
- Includes **OpenVMS extension**, which performs several functions such as converting filenames and showing uptime. See the OpenVMS sample script for more information.
- Includes **updated OpenSSL extension**, based on HP SSL Version 1.2 for OpenVMS (OpenSSL 0.9.7D).

Documentation

For more information about PHP, see http://www.php.net/.

Software Prerequisites

The **CSWS_PHP V1.3** kit requires that the following software is installed *before* you install CSWS_PHP:

- OpenVMS Alpha Version 7.3-2 or higher
  - or -
  OpenVMS I64 Version 8.2 or higher
- HP Secure Web Server Version 1.3-1 for OpenVMS
  - or -
  HP Secure Web Server Version 2.1 (or higher) for OpenVMS

Installing CSWS_PHP for OpenVMS

Important: If you are upgrading from an earlier version of CSWS_PHP and you modified PHP.INI, copy the file to another directory before you begin the installation. After the installation is complete, compare your old PHP.INI with the newly installed version and modify the new file as necessary.

Before you install CSWS_PHP (or any optional module), **shut down the Secure Web Server**. You can restart the server when the installation is complete.

To install the CSWS_PHP for HP Secure Web Server for OpenVMS kit, enter the following command.

Note: You must install the CSWS_PHP kit into the same device and directory where you installed the HP Secure Web Server for OpenVMS.

For example:

```
$ SHOW LOGICAL APACHE$ROOT
  "APACHE$ROOT" = "DISK1:[WEB_SERVER.APACHE.SPECIFIC.hostname.]
        = "APACHE$COMMON:"
1   "APACHE$COMMON" = "DISK1:[WEB_SERVER.APACHE.]

$ PRODUCT INSTALL CSWS_PHP/DESTINATION=DISK1:[WEB_SERVER]
```

For a description of PRODUCT INSTALL commands, see the *POLYCENTER Software Installation Utility User's Guide*.

As the installation procedure for CSWS_PHP 1.3 progresses, the system displays the following information.

```
The following product has been selected:

CPQ AXPVMS CSWS_PHP V1.3             Layered Product

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.


CPQ AXPVMS CSWS_PHP V1.3

    Hewlett-Packard Company & The Apache Software Foundation.

* This product does not have any configuration options.

Execution phase starting ...

The following product will be installed to destination:
    CPQ AXPVMS CSWS_PHP V1.3         DISK1:[WEB_SERVER.]

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

The following product has been installed:
    CPQ AXPVMS CSWS_PHP V1.3             Layered Product

CPQ AXPVMS CSWS_PHP V1.3

Post-installation tasks are required for PHP for OpenVMS.

The release notes give detailed directions.
This information is a brief checklist.

This installation modifies APACHE$ROOT:[CONF]HTTPD.CONF to
enable MOD_PHP.  Check HTTPD.CONF for accuracy.  The line
"Include /apache$root/conf/mod_php.conf" should be the only
difference.  Also study the MOD_PHP configuration file
(APACHE$ROOT:[CONF]MOD_PHP.CONF) for options required for your
site.
```

```
The Apache server must be shut down and restarted to make these
changes to HTTPD.CONF file take place.  Test that MOD_PHP
is working by accessing the sample script from a browser:

    http://<your web server host>/php/php_rules.php

Thank you for using Apache for OpenVMS.
```

After the installation is complete, start the *Secure Web Server* by entering the following command:

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

Removing CSWS_PHP for HP Secure Web Server for OpenVMS

You can remove the CSWS_PHP kit by using the PRODUCE REMOVE command. The HP Secure Web Server must be shut down before you remove CSWS_PHP.

To remove CSWS_PHP, enter the following commands:

```
    $ @SYS$STARTUP:APACHE$SHUTDOWN
    $ PRODUCT REMOVE CSWS_PHP
```

The PRODUCT REMOVE command removes all files created by this installation. It also attempts to remove the directories defined by this installation.

Using Extensions

CSWS_PHP 1.3 includes the extensions listed in the PHP.INI file shown below. The PHP.INI file provided with the CSWS_PHP kit resides in the PHP_ROOT:[000000] directory (APACHE$COMMON:[PHP]).

```
PHP.INI
;
; Uncomment for the automatic loading of extensions
;
;extension=php_bcmath.exe
;extension=php_bzip2.exe
;extension=php_calendar.exe
;extension=php_ctype.exe
;extension=php_dba.exe
;extension=php_exif.exe
;extension=php_ftp.exe
;extension=php_gd.exe
;extension=php_iconv.exe
;extension=php_ldap.exe
;extension=php_mhash.exe
;extension=php_mysql.exe
;extension=php_oci8.exe
;extension=php_odbc.exe
;extension=php_openssl.exe
;extension=php_openvms.exe
;extension=php_oracle.exe
;extension=php_pcre.exe
;extension=php_posix.exe
;extension=php_session.exe
;extension=php_sockets.exe
```

```
        ;extension=php_xml.exe
        ;extension=php_zip.exe
        ;extension=php_zlib.exe
```

There are two ways to load a PHP extension: using the dl() function and using the PHP.INI file. These methods are as follows:

- The dl() function allows the loading of extensions within a PHP script if the extension resides in the default extension_dir. Extension_dir is defined as PHP_ROOT:[EXTENSIONS] directory (APACHE$COMMON:[PHP.EXTENSIONS]).

- PHP.INI contains the extension statement to automatically load the extension for every PHP script executed. To enable the loading of the extension for every PHP script, uncomment the "extension=" statement as shown in the following example, and restart the Secure Web Server.

ODBC Extension

The ODBC extension works with any ODBC V2.5 capable server. The ODBC.PHP script works with Attunity Connect "On Platform" Package for OpenVMS Alpha using the RMS demo.

Add the following lines to the beginning of the APACHE$COMMON:[000000]LOGIN.COM file to enable the script to work with Attunity Connect:

```
$ !
$ ! Run the Attunity login if we find it
$ !
$ IF F$SEARCH ("NAVROOT:[BIN]NAV_LOGIN.COM") .NES. ""
$ THEN
$     @NAVROOT:[BIN]NAV_LOGIN.COM
$     DEFINE APACHE$ODBC_SHR ODNAVSHR
$     DEFINE APACHE$ODBC_PFX NV
$ ENDIF
```

The two logicals required to make the ODBC extension functional are APACHE$ODBC_SHR and APACHE$ODBC_PFX. These logicals are defined as follows:

- APACHE$ODBC_SHR
  This logical defines the ODBC shareable image to be used for the ODBC access.

- APACHE$ODBC_PFX
  This logical defines, if needed, any ODBC API prefix.

OCI Extension

Add the following lines to the beginning of the file APACHE$COMMON:[000000]LOGIN.COM:

For Oracle 8i:

```
$ !
$ ! Define the OCI extension logicals if we find the OCI client shareable
$ !
$ IF F$SEARCH ("ORA_ROOT:[UTIL]ORACLIENT_V817.EXE") .NES. ""
$ THEN
$     DEFINE APACHE$OCI_SHR ORA_ROOT:[UTIL]ORACLIENT_V817.EXE
$ ENDIF
```

For Oracle 9i:

```
$ !
$ ! Define the OCI extension logicals if we find the OCI client shareable
$ !
$ IF F$SEARCH ("ORA_ROOT:[LIB32]LIBCLNTSH.SO") .NES. ""
$ THEN
$     DEFINE APACHE$OCI_SHR ORA_ROOT:[LIB32]LIBCLNTSH.SO
$ ENDIF
```

The two logicals required to make the OCI extension functional are APACHE$OCI_SHR and APACHE$OCI_PFX. These logicals are defined as follows:

- APACHE$OCI_SHR
  This logical defines the OCI8 shareable image to be used for the OCI8 access.

- APACHE$OCI_PFX
  This logical defines, if needed, any OCI8 API prefix.

Sample PHP Scripts

The following PHP sample scripts are included in the CSWS_PHP 1.3 kit (calendar.php, info.php, odbc.php, and php_openvms.php). These scripts demonstrate the use of the provided extensions.

PHP_CALENDAR.PHP (Useful for Hebrew calendar in Unicode)

```
<?php

#
# Load the calendar extension if needed
#
if (! extension_loaded ("calendar"))
    dl ("php_calendar");

#
# Display the header
#
echo "    Testing the Calendar extension<br>\n";
#
# Test the calendar functions
#
$m = date("m", time());
$d = date("d", time());
$y = date("Y", time());
$jd = GregorianToJD($m,$d,$y);
echo "Gregorian month (abbr.): " . jdmonthname($jd, 0) . "<br>\n";
echo "Gregorian month: " . jdmonthname($jd, 1) . "<br>\n";
echo "Julian month (abbr.): " . jdmonthname($jd, 2) . "<br>\n";
echo "Julian month: " . jdmonthname($jd, 3) . "<br>\n";
echo "Jewish month: " . jdmonthname($jd, 4) . "<br>\n";
$y = 1800;
$jd = GregorianToJD($m,$d,$y);
echo "French month: " . jdmonthname($jd, 5) . "<br>\n";

?>
```

PHP_INFO.PHP

```
<?php

#
# Display the header
#
echo "    Testing the PHPINFO () function<br>\n";
#
# Test the PHPINFO () function
#
phpinfo (INFO_ALL);

?>
```

PHP_ODBC.PHP

```
<?php

 #
 # Load the ODBC extension if needed
 #
 if (! extension_loaded ("odbc"))
     dl ("odbc");

 #
 # Display the header
 #
 echo "    Testing the ODBC extension<br>\n";

 #
 # Test the ODBC functions
 #
 $ctx = odbc_connect ("NAVDEMO", "", "");
 $cur = odbc_exec ($ctx, "select c_custkey, c_name from customer");
 odbc_result_all ($cur, "border=1 align='center'");
 $rc = odbc_free_result ($cur);
 odbc_close ($ctx);

?>
```

PHP_OPENVMS.PHP

```
<?php

#
# Load the OpenVMS extension if needed
#
if (! extension_loaded ("openvms"))
     dl ("php_openvms");

 #
 # Display the header
 #
 echo "    Testing the OpenVMS extension<br>\n";

 #
 # Allow only errors to be reported
 #
 error_reporting (E_ERROR);

 #
 # Test the OpenVMS convert filename function
 #
 # openvms_cvt_filename (func_code, file_name)
```

```
    #
    # func_codes:
    #       OPENVMS_CVT_VMS_TO_UNIX         Convert vms filespec to unix filespec
    #       OPENVMS_CVT_UNIX_TO_VMS         Convert unix filespec to vms filespec
    #
    $VmsFn = "PHP_ROOT:[SCRIPTS]PHP_OPENVMS.PHP";
    $UnixFn = openvms_cvt_filename (OPENVMS_CVT_VMS_TO_UNIX, $VmsFn);
    if ($UnixFn === FALSE)
        echo "openvms_cvt_filename (OPENVMS_CVT_VMS_TO_UNIX, \"$VmsFn\") = " .
openvms_message (openvms_status ()) . "<br>\n";
    else
        echo "openvms_cvt_filename (OPENVMS_CVT_VMS_TO_UNIX, \"$VmsFn\") =
$UnixFn<br>\n";

    #
    # Test the OpenVMS getdvi function
    #
    # openvms_getdvi (item_code [,device_name])
    #
    # item_codes:
    #       <item_code>                     Any Item code supported by F$GETDVI
    #       "?"                             List of supported item codes
    # device_name:                          Defaults to "TT"
    #
    $item = "DISPLAY_DEVNAM";
    $val = openvms_getdvi ($item);
    if ($val === FALSE)
        echo "openvms_getdvi (\"$item\") = " . openvms_message (openvms_status ()) .
"<br>\n";
    else
        echo "openvms_getdvi (\"$item\") = $val<br>\n";

    #
    # Test the OpenVMS getjpi function
    #
    # openvms_getjpi (item_code [,proc_name][,pid])
    #
    # item_codes:
    #       <item_code>                     Any Item code supported by F$GETJPI
    #       "?"                             List of supported item codes
    # proc_name:                            Any process name
    # pid:                                  Any process ID or -1 wild card
    #
    $item = "LAST_LOGIN_I";
    $val = openvms_getjpi ($item);
    if ($val === FALSE)
        echo "openvms_getjpi (\"$item\") = " . openvms_message (openvms_status ()) .
"<br>\n";
    else
        echo "openvms_getjpi (\"$item\") = $val<br>\n";

    #
    # Test the OpenVMS getsyi function
    #
    # openvms_getsyi (item_code [,node_name][,csid])
    #
    # item_codes:
    #       <item_code>                     Any Item code supported by F$GETSYI
    #       "?"                             List of supported item codes
    # node_name:                            Any node name
    # csid:                                 Any cluster system ID or -1 wild card
    #
    $item = "BOOTTIME";
    $val = openvms_getsyi ($item, "", 0);
    if ($val === FALSE)
        echo "openvms_getsyi (\"$item\") = " . openvms_message (openvms_status ()) .
"<br>\n";
    else
        echo "openvms_getsyi (\"$item\") = $val<br>\n";

    #
    # Test the OpenVMS time function
    #
    # openvms_time ([millisecond_time])
    #
    $val = openvms_time ();
    if ($val === FALSE)
        echo "openvms_time () = " . openvms_message (openvms_status ()) . "<br>\n";
    else
        echo "openvms_time () = $val<br>\n";

    #
    # Test the OpenVMS uptime function
    #
    # openvms_uptime ()
    #
    $uptime = openvms_uptime ();
    if ($uptime === FALSE)
        echo "openvms_uptime () = " . openvms_message (openvms_status ()) . "<br>\n";
    else
        echo "openvms_uptime () = $uptime<br>\n";

    echo "<br>\n";

    #
    # Show the cluster info
    #
    ShowCluster ();

    #
    # Show the system info
    #
    ShowSystem ();

    #
    # Show Cluster
    #
    function ShowCluster ()
    {

    $SystemId = openvms_getsyi ("SCSSYSTEMID");
    $NodeName = openvms_getsyi ("NODENAME");
    $Time = strtok (openvms_time (), ".");

    echo "<pre>\n";
    $hdr = "View of Cluster from system ID $SystemId  node: $NodeName";
    $pad = str_repeat (" ", 79 - (strlen ($hdr) + strlen ($Time)));
    echo $hdr . $pad . $Time . "\n";
    echo "+-----------------------------+\n";
    echo "|      SYSTEMS      | MEMBERS |\n";
    echo "|-------------------+---------|\n";
    echo "|  NODE  | SOFTWARE |  STATUS |\n";
    echo "|--------+----------+---------|\n";

    $ctx = -1;
    while (1)
        {
        $csid = openvms_getsyi ("NODE_CSID", "", &$ctx);
        if ($csid === FALSE)
            {
            $status = openvms_status ();
            if ($status != 2560)
                echo openvms_message (openvms_status ()) . "<br>\n";
            break;
            }
        $NodeName = str_pad (openvms_getsyi ("NODENAME", "", $csid), 6, " ",
STR_PAD_RIGHT);
        $swtype = openvms_getsyi ("NODE_SWTYPE", "", $csid);
        $swvers = openvms_getsyi ("NODE_SWVERS", "", $csid);
        $software = str_pad ($swtype . $swvers, 8, " ", STR_PAD_RIGHT);
        if (strcasecmp (openvms_getsyi ("CLUSTER_MEMBER", "", $csid), "TRUE") == 0)
            $status =  "MEMBER";
        else
            $status = "      ";
        echo "| $NodeName | $software | $status  |\n";
        }

    if (openvms_getsyi ("CLUSTER_NODES") == 0)
        {
        $NodeName = str_pad (openvms_getsyi ("NODENAME"), 6, " ", STR_PAD_RIGHT);
        $swtype = openvms_getsyi ("NODE_SWTYPE", "", $csid);
        $swvers = openvms_getsyi ("NODE_SWVERS", "", $csid);
        $software = str_pad ($swtype . $swvers, 8, " ", STR_PAD_RIGHT);
        if (strcasecmp (openvms_getsyi ("CLUSTER_MEMBER", "", $csid), "TRUE") == 0)
            $status =  "MEMBER";
        else
            $status = "      ";
        echo "| $NodeName | $software | $status  |\n";
        }

    echo "+----------------------------+\n";
    echo "</pre>\n";

    }

    #
    # Show System (Requires World Privilege)
    #
    function ShowSystem ()
    {

    $VmsVer = trim (openvms_getsyi ("VERSION"));
    $NodeName = openvms_getsyi ("NODENAME");
    $UpTime = trim (openvms_uptime ());
    $Time = openvms_time ();

    echo "<pre>\n";
    echo "OpenVMS $VmsVer  on node $NodeName  $Time  Uptime  $UpTime\n";
    echo "  Pid    Process Name    State  Pri      I/O       CPU       Page flts  Pages\n";
    $ctx = -1;
    while (1)
        {
        $pid = openvms_getjpi ("PID", "", &$ctx);
        if ($pid === FALSE)
            {
            $status = openvms_status ();
            if ($status != 2472)
                echo openvms_message (openvms_status ()) . "<br>\n";
            break;
            }
        $prcpid = str_pad ($pid, 8, " ", STR_PAD_RIGHT);
        $prcnam = str_pad (openvms_getjpi ("PRCNAM", "", $pid), 15, " ", STR_PAD_RIGHT);
        $state = str_pad (openvms_getjpi ("STATE", "", $pid), 5, " ", STR_PAD_RIGHT);
        $pri = str_pad (openvms_getjpi ("PRI", "", $pid), 3, " ", STR_PAD_LEFT);
        $io = openvms_getjpi ("DIRIO", "", $pid) + openvms_getjpi ("BUFIO", "", $pid);
        $io = str_pad ($io, 9, " ", STR_PAD_LEFT);
        $cputim = openvms_time (openvms_getjpi ("CPUTIM", "", $pid));
        $pagflts = str_pad (openvms_getjpi ("PAGEFLTS", "", $pid), 9, " ", STR_PAD_LEFT);
         $pages = openvms_getjpi ("GPGCNT", "", $pid) + openvms_getjpi ("PPGCNT", "",
        $pid);
        $pages = $pages / (openvms_getsyi ("PAGE_SIZE") / 512);
        $pages = str_pad ($pages, 6, " ", STR_PAD_LEFT);
        $multithread = openvms_getjpi ("MULTITHREAD", "", $pid);
        $owner = openvms_getjpi ("OWNER", "", $pid);
        $mode = openvms_getjpi ("MODE", "", $pid);
        if ($multithread >= 1)
            $sts = "M";
        else
            $sts = " ";
        if ($owner != 0)
            $sts .= "S";
        else
        if (strcasecmp ($mode, "NETWORK") == 0)
```

```
            $sts .= "N";
        else
        if (strcasecmp ($mode, "BATCH") == 0)
            $sts .= "B";
        else
            $sts .= " ";
        echo "$prcpid $prcnam $state  $pri$io$cputim $pagflts $pages $sts\n";
        }
    echo "</pre>\n";

  }
  ?>
```

Release Notes

This section contains notes on the current release of CSWS_PHP.

- PHP$ logical names changed to APACHE$

  In Version 1.2 and higher of CSWS_PHP for HP Secure Web Server, the prefix of the PHP$ logical names was changed to APACHE$. This change allows other Secure Web Server-based scripting languages (such as Perl) to use the database connectivity features. The changed logical names are as follows:

Old Logical Name (1.0 and 1.1)	New Logical Name (1.2 and higher)
PHP$ODBC_SHR	APACHE$ODBC_SHR
PHP$ODBC_PFX	APACHE$ODBC_PFX
PHP$OCI_SHR	APACHE$OCI_SHR
PHP$OCI_PFX	APACHE$OCI_PFX

- Add logical names to PHP_SETUP.COM for extended file name support

  If you are using CSWS_PHP alone or with HP Secure Web Server Version 1.3 or higher, and you have ODS-5 files with extended file names such as grab_globals.lib.php (which has multiple dots), the files will not be processed correctly.

  To solve this problem, add the following logical definitions to the end of PHP_SETUP.COM located in APACHE$COMMON:[000000]:

  ```
  $ DEFINE /NoLog DECC$EFS_CASE_PRESERVE ENABLED
  $ DEFINE /NoLog DECC$EFS_CASE_SPECIAL ENABLED
  $ DEFINE /NoLog DECC$EFS_CHARSET ENABLED
  $ DEFINE /NoLog DECC$FILE_SHARING ENABLED
  ```

  Using the commands SET PROCESS/PARSE=EXTENDED and SET PROCESS/PARSE=EXTENDED/CASE=SENSITIVE does not work in this situation.

- Configuring CSWS_PHP **not** required

  During the installation, the file `PHP_SETUP.COM` is added to the `APACHE$ROOT` directory, and an include for MOD_PHP.CONF is added to the end of HTTPD.CONF. When you start the Secure Web Server, `PHP_SETUP.COM` is run and PHP is loaded into the server. You do not need to configure CSWS_PHP.

- PHP **DNS** functions supported only with TCP/IP Services for OpenVMS

  This version of CSWS_PHP supports the **CHECKDNSRR** and **GETMXRR** functions only on systems using HP TCP/IP Services for OpenVMS. These functions may be supported with other TCP/IP products in a future CSWS_PHP kit.

- PHP **LINK** functions not supported

  This version of CSWS_PHP does not support the **LINK**, **LINKINFO**, **SYMLINK**, and **READLINK** functions. These functions may be supported in a future CSWS_PHP kit.