

CSWS_JAVA for Compaq Secure Web Server for OpenVMS Alpha

Installation Guide and Release Notes

CSWS_JAVA Version 1.1
CPQ-AXPVMS-CSWS_JAVA-V0101--1.PCSI

Overview

What's New in Version 1.1

Software Prerequisites

Documentation

Before Beginning the Installation

Installing CSWS_JAVA

 Installing CSWS_JAVA on an ODS-5 Enabled Disk

Configuring CSWS_JAVA

Configuring the JServ Servlet Engine with CSWS_JAVA

Building the Sample Web Application on OpenVMS

Running Tomcat

Release Notes (Tomcat)

 Installation restriction removed

 Java SecurityManager included

 Security vulnerabilities fixed in Tomcat 3.2.1

 Direct HTTP/SSL support not available

 Jakarta START -SECURITY problem corrected

 Setting up Tomcat to use Fast VM

 Using the latest (or beta) version of Compaq Java Development Kit with CSWS_JAVA

 Password for admin example

 Slow access the first time Tomcat server is invoked

 Pause before serving first page after restart

 Configuration dialog question about updating configuration data file

Release Notes (MOD_JSERV and JServ Servlet Engine)

 JHTML files must be in STREAM_LF format

 Delete or move JDK11*_CLASSES.ZIP

 EXAMPLE directory filenames are case-sensitive

 Faster updates to the JServ log file

 Using the latest (or beta) version of Compaq Java Development Kit or Fast VM with JServ

 Error when attempting to start JServ

 Sample Hello.java

 Enabling JSSI support

 "Internal Server Error" returned under heavy user load

Overview

CSWS_JAVA includes the following projects:

- Tomcat
- Ant
- Apache JServ

See the [Jakarta Apache Project](#) for more information about Tomcat and other projects.

Tomcat

Tomcat is an extension to the Apache server, but it runs independently of Apache, in a separate process. You can configure your system so that the Apache server serves HTML pages, while Tomcat serves the JSP pages and runs the servlets.

CSWS_JAVA includes the following new [Apache Tomcat](#) technologies:

- JavaServer Pages 1.1
- Java Servlet 2.2
- MOD_JK
- MOD_JSERV

[Tomcat](#) is the release following Apache JServ, and is the reference implementation for the Java Servlet 2.2 and JavaServer Pages 1.1 technologies. (Apache JServ was a Servlet API 2.0-compliant container.) CSWS_JAVA includes the final Tomcat Version 3.2.4.

Tomcat is a servlet container with a JSP environment. A servlet container is a runtime shell that manages and invokes servlets on behalf of users. Servlet containers can be standalone, in-process, or out-of-process. CSWS_JAVA includes support for standalone servlet containers and out-of-process servlet containers. Support for in-process servlet containers (JSSI) will be included in a future version of Tomcat.

If more than one version of Java is installed on your system, Tomcat first searches for Java 1.3.0, then Java 1.2.2, then Java 1.1.8. It uses the version it finds first.

See [Tomcat - A Minimalistic User's Guide](#) for more information.

Ant

Also included in CSWS_JAVA is [Ant](#). Ant is a partial implementation of the Jakarta Ant subproject and its use is limited to building the included sample web applications and simple user-written web applications for Tomcat.

Apache JServ

CSWS_JAVA also contains the older CSWS_JSERV kit, which includes the following [Apache JServ](#) projects:

- MOD_JSERV
- Java Servlet 2.0
- JSSI 1.1.1

Note: [Apache JServ](#) is currently in maintenance-only mode. There will be no new official releases and only well tested patches are being committed. No new features are being added.

CSWS_JAVA includes the Apache JServ components from the CSWS_JSERV kit released in October 2000. In-process servlet containers (JSSI) support does not currently exist in Tomcat. It can only be provided by Apache JServ. However, your servlets should work without any problem using the new Tomcat product in CSWS_JAVA.

What's New in Version 1.1

CSWS_JAVA Version 1.1 contains support for **Tomcat 3.2.4**.

Enhancements and Bug Fixes

Tomcat 3.2.4 contains the following enhancements and bug fixes. More information can be found in the [Tomcat 3.2.4 Release Notes](#).

- Cookie name expires is a reserved token.
- Thread initialization problem in thread pool.
- AJP12 returned invalid HTTP headers when redirecting to very long URLs.
- Fixed casting problem in JspFactoryImpl.getPageContext().
- Setting session-timeout in web.xml did not prevent sessions from timing out.
- Fixed race condition in ServerSocketFactory.getDefault().
- Removed the restrictions on encoded special characters in URLs that was added as a security precaution in 3.2.3. The encoded special characters are not decoded and remain the URL and path info returned to servlets.
- Jk_nt_service now supports the ability to be restarted automatically by the Windows 2000 service control manager if Tomcat terminates abnormally.
- Fixed invalid servlet mapping in web.xml generated by JspC.
- Added findResource() and findResources() to AdaptiveClassLoader12.
- A Date: HTTP header is now sent in responses when running stand alone.
- Simple held on to a reference to removed objects preventing garbage collection.
- Tomcat 3.2.4 now ships with JAXP 1.1. Prior releases used JAXP 1.0.1. Tomcat 3.2.4 remains completely compatible with the older version of JAXP and there is no requirement for users to upgrade to JAXP 1.1 unless their applications require the new version.
- Fixed NullPointerException in HttpConnectionHandler.

Security Fixes

Tomcat 3.2.4 contains the following security-related fixes. More information can be found in the [Tomcat 3.2.4 Release Notes](#).

- The randomness of generated session ids has been enhanced to prevent the generation of guessable ids.
- Non-normalized URIs, for example /examples/jsp/security//protected/index.jsp or /examples/jsp/./jsp/security/protected/index.jsp would bypass the security constraints specified in web.xml.
- URIs are now normalized prior to use by the container. Servlets will receive the normalized path from calls to HttpServletRequest.getRequestURI(). This is expected to become the behaviour defined in the Servlet 2.3 specification.
- URL encodings for special characters are now forbidden in request URIs. If a request URI contains %25, %2E, %2F or %5c a 404 error will be returned. This prevents the use of URL encodings to bypass the URI normalization process.
- An HTTP request with no protocol specified would return an unprocessed source for a JSP file. For example
- GET /examples/jsp/num/numguess.jsp would return the source for the numguess.jsp file.
- Tomcat 3.2.2 beta releases prior to beta 3 had allowed URI components to be decoded twice. This problem only appears when using JDK 1.3.0 or later. The double decode problem caused URLs that reveal a directory listing outside the web application. Other versions of this same double decode attack could reveal the contents of files outside the web application.

Software Prerequisites

CSWS_JAVA for the *Compaq Secure Web Server for OpenVMS Alpha* requires the following software:

- OpenVMS Alpha Version 7.2-1 (or higher)
- [Compaq Secure Web Server Version 1.1-1 \(or higher\) for OpenVMS Alpha](#)
- [Compaq Java Development Kit Version 1.2.2 \(or higher\) for OpenVMS Alpha](#)
- All [patches](#) required for Compaq Java Development Kit for OpenVMS Alpha
- For production environments, Compaq recommends that you [install CSWS_JAVA on an ODS-5 enabled disk](#). Your installation of the *Compaq Secure Web Server* can remain on an ODS-2 disk.

Documentation

For information about Tomcat, see [the Jakarta Apache Project](#) and [Tomcat - A Minimalistic User's Guide](#).

For more information on Apache JServ and Apache JSSI, see [the Java Apache Project](#) website. General information about Apache is available from the [Apache Software Foundation](#).

Before Beginning the Installation

Before you install the CSWS_JAVA kit, perform the following steps.

1. Remove CSWS_JSERV.

Before you install CSWS_JAVA, Compaq recommends that you remove CSWS_JSERV, if it was previously installed.

Perform a backup on all of your *Compaq Secure Web Server* files, then enter the following commands to remove CSWS_JSERV:

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

```
The following product has been selected:
  CPQ AXPVMS CSWS_JSERV V1.0           Layered Product
```

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

```
The following product will be removed from destination:
  CPQ AXPVMS CSWS_JSERV V1.0           DISK$ALPHASYS: [VMS$COMMON.]
```

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

```
The following product has been removed:
  CPQ AXPVMS CSWS_JSERV V1.0           Layered Product
```

2. Delete the JSERV startup command procedure.

Enter the following command:

```
$ DELETE APACHE$ROOT:[000000]START_JSERV_MANUAL.COM;*
```

3. Start the Compaq Secure Web Server.

Enter the following commands:

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

If the web server does not restart, check
APACHE\$ROOT:[000000]APACHE\$\$SERVER.LOG for errors.

Installing CSWS_JAVA

For production environments, Compaq recommends that you install CSWS_JAVA on an ODS-5 enabled disk. Your installation of the *Compaq Secure Web Server* can remain on an ODS-2 disk. **You do not need to install CSWS_JAVA into the same disk or directory as the *Compaq Secure Web Server*.**

To install the CSWS_JAVA kit, enter the following command where *DISK\$DKA0* is the name of the disk where you want to install CSWS_JAVA.

```
$ PRODUCT INSTALL CSWS_JAVA/DEST=DISK$DKA0:[000000]
```

For a description of the features you can request with the PRODUCT INSTALL command when starting an installation such as running the IVP, purging files, and configuring the installation, see the *POLYCENTER Software Installation Utility User's Guide*.

As the installation procedure progresses, the system displays the following information.

The following product has been selected:

```
CPQ AXPVMS CSWS_JAVA V1.1          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_JAVA V1.1: Java modules for CSWS for OpenVMS Alpha
```

```
    COPYRIGHT (c) 1995-1999 The Apache Group.  All rights reserved.
```

* This product does not have any configuration options.

Execution phase starting ...

The following product will be installed to destination:

```
CPQ AXPVMS CSWS_JAVA V1.1          DISK$ODS5:[000000.]
```

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

The following product has been installed:
CPQ AXPVMS CSWS_JAVA V1.1 Layered Product

CPQ AXPVMS CSWS_JAVA V1.1: Java modules for CSWS for OpenVMS Alpha

Post installation tasks required for CSWS_Java for OpenVMS Alpha

Configure OpenVMS aspects of the CSWS Java by:

```
$ @SYS$MANAGER:APACHE$JAKARTA
```

The default installation uses the SYSTEM account to run the CSWS_JAVA (Jakarta/Tomcat) engine. If you are planning to share html files with Compaq's Secure Web Server, it is recommended that you change the Jakarta directory tree's ownership to APACHE\$WWW.

Select Option 1 from the CSWS Jakarta Configuration Menu

Example:

Enter configuration option: 1

Enter the OpenVMS account name for Jakarta (Tomcat) [SYSTEM]: apache\$www

To operate successfully, the server processes must have read access to the installed files and read-write access to certain other files and directories. Compaq recommends that you use this procedure to set the owner UIC on the CSWS files and directories to match the server. If you are changing the OpenVMS account name, you might want to change the ownership of the Jakarta tree.

Set owner UIC to APACHE\$WWW on CSWS java jakarta files (Yes/No) [Yes]: Y
This could take a minute or two . . .

After configuration, start the CSWS Java (Jakarta) by entering:

```
$ @SYS$STARTUP:APACHE$JAKARTA_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 Tomcat servlet engine at system boot time by adding the following lines to SYS\$MANAGER:SYSTARTUP_VMS.COM:

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

Shutdown the Tomcat servlet engine at system shutdown time by adding the following lines to SYS\$MANAGER:SYSHUTDOWN.COM:

```
$ file := SYS$STARTUP:APACHE$JAKARTA_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 Compaq Secure Web Server just installed, configured, and started.

URL `http://host.domain:8080/` should display the standard introductory page from the Apache Software Foundation. This has the bold text "Tomcat Version 3.2 (Final)" with the Tomcat logo in the upper left hand corner.

If you do not see this page, check the CSWS Java release notes.

Thank you for using the CSWS_Java.

Installing CSWS_JAVA on an ODS-5 Enabled Disk

Compaq recommends that large production sites consider installing CSWS_JAVA on an ODS-5 enabled disk because of directory depth issues and long filename support. Using an ODS-5 enabled disk avoids servlet name space collision that will occur with the 39.39 character filename limitation on an ODS-2 file system.

For example, the URL `http://hostname/examples/jsp/num/numguess.jsp` builds a temporary file in the directory `[APACHE.JAKARTA.WORK.localhost_8080_2Fexamples]` called

```
_0002fjsp_0002fnum_0002fnumguess_0002ejjspnumguess_jsp_0.java
```

On an ODS-2 file system, the filename is truncated to

```
_0002FJSP_0002FNUM_0002FNUMGUESS_0002EJ.JAVA
```

If your URL grows to `http://hostname/examples/jsp/num/numguess.jsp/new_numguess.jsp` the uniqueness of the filename on an ODS-2 file system comes into question, and you will start to see internal servlet errors (wrong name errors).

Configuring CSWS_JAVA

After the installation is complete, perform the following steps.

1. **Change the Jakarta directory tree's ownership to APACHE\$WWW and set the owner UIC.**

The default installation uses the SYSTEM account to run the CSWS_JAVA Tomcat engine. If you are planning to share HTML files with the *Compaq Secure Web Server*, change the Jakarta directory tree's ownership to APACHE\$WWW by running the CSWS_JAVA configuration utility and selecting the first option.

For example:

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

```
CSWS Jakarta Configuration Menu
```

```
Configuration Options:
```

- 1 - Change Username
- 2 - Add ACL to Jakarta (Tomcat) directories
- 3 - Configure Apache's httpd.conf for Jakarta Adapters

- 6 - View current configuration
- 7 - Start CSWS Jakarta (Tomcat) for OpenVMS
- 8 - Stop CSWS Jakarta (Tomcat) for OpenVMS

- S - Save Configuration
- [E]- Exit Configuration procedure

Enter configuration option: 1

Enter the OpenVMS account name for Jakarta (Tomcat) [SYSTEM]: APACHE\$WWW

To operate successfully, the server processes must have read access to the installed files and read-write access to certain other files and directories. Compaq recommends that you use this procedure to set the owner UIC on the CSWS files and directories to match the server. If you are changing the OpenVMS account name, you might want to change the ownership of the Jakarta tree.

Set owner UIC to APACHE\$WWW on CSWS java jakarta files (Yes/No) Yes

This could take a minute or two . . .

Would you update the Jakarta configuration data file (Yes/No) [Yes]

Configuration file [SYS\$COMMON:[SYSMGR]APACHE\$JAKARTA_CONFIG.DAT]:

Press RETURN to continue

Important: Check quota requirements for servlet engines

When you select the user account for the Jakarta (Tomcat) or JServ servlet engines, consider Java quota requirements to ensure best performance of your Java applications.

The default quota values for the APACHE\$WWW account that are set by the *Compaq Secure Web Server* installation might not be optimized for Java. In particular, you might need to increase FILLM (and the related CHANNELCNT SYSGEN parameter), PGFLQUO, and BYTLM. These are pooled quotas. If you are configuring the JServ servlet engine, which is a subprocess, you need to be aware of the impact on these quotas from other Apache child processes in the same job tree. The Jakarta (Tomcat) servlet engine is a detached process and is not affected by Apache child processes.

For more information on Java quota requirements, see the section on *Setting Process Quotas for Better Performance on OpenVMS* in the *Compaq Fast Virtual Machine (Fast VM) 1.3.0-1 for the Java 2 Platform on OpenVMS Alpha Release Notes*.

2. Configure Compaq Secure Web Server to work with Tomcat.

Enter the following command, and select Option 3, then Option 1. This edits the HTTPD.CONF with the proper include line.

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

CSWS Jakarta Configuration Menu

Configuration Options:

- 1 - Change Username
- 2 - Add ACL to Jakarta (Tomcat) directories

3 - Configure Apache's httpd.conf for Jakarta Adapters

6 - View current configuration

7 - Start CSWS Jakarta (Tomcat) for OpenVMS

8 - Stop CSWS Jakarta (Tomcat) for OpenVMS

S - Save Configuration

[E]- Exit Configuration procedure

Enter configuration option: 3

Jakarta Configuration: (Mod_jserv Mod_jk adapters)

Configuration Options:

1 - Enable mod_jk httpd.conf

2 - Disable mod_jk httpd.conf

3 - Enable mod_jserv httpd.conf

4 - Disable mod_jserv httpd.conf

7 - restart CSWS (Apache) for OpenVMS

[E]- Exit Configuration procedure

Enter configuration option: 1

Location of httpd.conf [APACHE\$COMMON:[CONF]HTTPD.CONF]

Mod_jk configuration file [VARMIT\$DKA200:[APACHE.JAKARTA.CONF]

MOD_JK.CONF-AUTO]

3. Add the Jakarta command to the *Compaq Secure Web Server* LOGIN.COM file.

To set up the Tomcat environment after you install CSWS_JAVA, add the following line to the *Compaq Secure Web Server* LOGIN.COM before the exit at the top of the file:

```
$ @SYS$STARTUP:APACHE$JAKARTA ENV
```

If you do not add this line, the MOD_JK module is not loaded during CSWS startup.

4. If the *Compaq Secure Web Server* is currently running, stop it and then restart it so that these configuration changes take effect.

5. If you installed CSWS_JAVA on an ODS-5 file system, do the following:

- o Create a .TOMCATRC file in the account's SYS\$LOGIN directory that will be running Tomcat.

For example, if you are running Tomcat from APACHE\$WWW, enter the following commands:

```
$ CREATE APACHE$ROOT:[000000].TOMCATRC
$ ! add special DEC C logicals to enable ODS
$ ! DECC$EFS* are only available with OpenVMS V7.2-1
$ ! and VMS721_ACRTL V2.0
$ DEFINE/JOB DECC$EFS_CASE_SPECIAL "ENABLE"
$ DEFINE/JOB DECC$EFS_CASE_PRESERVE "ENABLE"
$ DEFINE/JOB APACHE$JAKARTA_ENABLE_ODS5 1
^Z
```

- o Start Tomcat.

Enter the following command:

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

CSWS Jakarta Configuration Menu

Configuration Options:

    1 - Change Username
    2 - Add ACL to Jakarta (Tomcat) directories
    3 - Configure Apache's httpd.conf for Jakarta Adapters

    6 - View current configuration
    7 - Start CSWS Jakarta (Tomcat) for OpenVMS
    8 - Stop CSWS Jakarta (Tomcat) for OpenVMS

    S - Save Configuration
    [E]- Exit Configuration procedure

Enter configuration option: 7
-> Tomcat Directory /VARMIT$DKA200/APACHE/JAKARTA/
Using Java 1.3.0 setup
Starting Tomcat...
Starting TOMCAT_8007 as a detached network process
%APACHE-S-PROC_ID, identification of created process is 000012DE
Tomcat Logicals and Classpaths are cleared
```

If you started Tomcat without configuring it for ODS-5, you can force Tomcat to reconfigure the installation for ODS-5 by following these steps:

1. Delete all the files in the work directory (default is [apache.jakarta.work])
 2. Delete all subdirectories of WebApps (default is [apache.jakarta.webapps])
 3. Do *not* delete any of the .war files in the WebApps directory. Delete only the subdirectories.
 4. Restart Tomcat to recreate the WebApps subdirectories and [.work] files.
- o Starting Tomcat using a different configuration file.

By default, Tomcat uses TOMCAT_HOME/conf/server.xml for configuration. The default configuration uses TOMCAT_HOME as its base for the contexts.

You can change this by using the -f /path/to/server.xml option, with a different server configuration file and setting the home property of the context manager. See [Tomcat - A Minimalistic User's Guide](#) for more information.

Note: On OpenVMS, these commands are case-sensitive. Put quotes around the UNIX portion of the command to retain lowercase.

To change the startup directory, enter the following:

```
$ @sys$startup:apache$jakarta start "-f"
"/path/to/server.xml"
```

6. Verify the current Tomcat configuration.

Enter the following command and select Option 6. If the Tomcat Servlet engine is running, you will see a **TOMCAT_8007** process.

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

```
CSWS Jakarta Configuration Menu
```

```
Configuration Options:
```

- 1 - Change Username
- 2 - Add ACL to Jakarta (Tomcat) directories
- 3 - Configure Apache's httpd.conf for Jakarta Adapters

- 6 - View current configuration
- 7 - Start CSWS Jakarta (Tomcat) for OpenVMS
- 8 - Stop CSWS Jakarta (Tomcat) for OpenVMS

```
[E]- Exit Configuration procedure
```

```
Enter configuration option: 6
```

```
-> Tomcat Directory /VARMIT$DKA200/APACHE/JAKARTA/  
%DCL-I-SUPERSEDE, previous value of APACHE$MOD_JSERV has been superseded  
%DCL-I-SUPERSEDE, previous value of APACHE$MOD_JK has been superseded  
Using Java 1.3.0 setup  
%DCL-W-EXPSYN, invalid expression syntax - check operators and operands  
Tomcat environment Initialized
```

```
Jakarta Configuration:
```

```
Configuration file: SYS$COMMON:[SYSMGR]APACHE$JAKARTA_CONFIG.DAT
```

```
OpenVMS Account Name:    APACHE$WWW  
Tomcat home:            /VARMIT$DKA200/APACHE/JAKARTA/
```

```
Java Version information:
```

```
java version "1.3.0"  
Java(TM) 2 Runtime Environment, Standard Edition  
Classic VM (build 1.3.0-085, 02/01/2001-02:59, native threads, jit)  
Java$classpath:
```

```
JAVA$CLASSPATH" = "SYS$COMMON:[JAVA$130.LIB]TOOLS.JAR"  
(LNM$PROCESS_TABLE)  
= "VARMIT$DKA200:[APACHE.JAKARTA.LIB]JAKARTA_VMS_CLASSES.JAR"  
= "VARMIT$DKA200:[APACHE.JAKARTA.LIB]WEBSERVER.JAR"  
= "VARMIT$DKA200:[APACHE.JAKARTA.LIB]JASPER.JAR"  
= "VARMIT$DKA200:[APACHE.JAKARTA.LIB]JAXP.JAR"  
= "VARMIT$DKA200:[APACHE.JAKARTA.LIB]PARSER.JAR"  
= "VARMIT$DKA200:[APACHE.JAKARTA.LIB]SERVLET.JAR"  
= "VARMIT$DKA200:[APACHE.JAKARTA.LIB]ANT.JAR"  
= "[]"
```

```
OpenVMS V7.2-1 on node VARMIT 1-MAR-2001 08:51:30.02 Up 10 00:29:34  
  Pid  Process Name  State Pri  I/O    CPU    Page flts Pages  
000012DE TOMCAT_8007 HIB    6 19586  0 00:00:10.93 4845 2909 M
```

Note: The first invocation of Tomcat completes the installation of the environment, so there is a delay before Tomcat is ready to serve JSP pages. Subsequent invocations of Tomcat will be faster.

7. **If the *Compaq Secure Web Server Jakarta Tomcat Servlet engine* does not start, check the log files in the default directory of the account.**

Enter the following commands:

```
$ DIR APACHE$ROOT:[000000]APACHE$JAKARTA*.LOG

Directory APACHE$ROOT:[000000]

APACHE$JAKARTA_SERVER_OUTPUT.LOG;1

Total of 1 file.

$ TYPE APACHE$ROOT:[000000]APACHE$JAKARTA_SERVER_OUTPUT.LOG

$ Set NoOn
$ VERIFY = F$VERIFY(F$TRNLNM("SYLOGIN_VERIFY"))
-> Tomcat Directory /SYS$COMMON/APACHE/JAKARTA/
Using Java 1.3.0 setup
Setting up symbols for foreign command line usage...
JAVA$FILENAME_CONTROLS now set to: -1
Running Tomcat.....
Exceeded quota, Please raise paging file quota
Requires a minimum of 200,000 free
Current available is: 100000
%SYSTEM-F-EXQUOTA, process quota exceeded
  SYSTEM          job terminated at 1-MAR-2001 09:31:53.02

Accounting information:
Buffered I/O count: 81 Peak working set size: 2016
Direct I/O count: 74 Peak virtual size: 167936
Page faults: 58 Mounted volumes: 0
Charged CPU time: 0 00:00:00.10
Elapsed time: 0 00:00:00.24
```

8. **Optional: Add new CLASSPATH entries.**

To add new CLASSPATH entries (for example, JDBC drivers), add the following line to your .TOMCATRC file:

```
$ DEFINE APACHE$JAKARTA_USER_CLASSPATH NAVROOT:[JAVA]NVJDBC1.JAR
```

9. **Optional: Supply additional JVM command line parameters.**

You may need to supply additional JVM command line parameters if, for example, you need to increase the maximum heap size to 128 MB. (Use the appropriate command line settings for the version of Java that is installed. For more information, enter `java -h`)

Create a text file with all of the JVM command line settings. Enter the following command:

```
$ CREATE TOMCAT_JVM_ARGS.DAT
-mx128m
^Z
```

Add the following line to your .TOMCATRC file:

```
$ def APACHE$JAKARTA_JAVA_PARAMETERS_FILE disk:[directory] -
_$ TOMCAT_JVM_ARGS.DAT
```

Note: Make sure that the APACHE\$WWW account can read these files.

10. **Access the included jsp and servlet examples** via `http://hostname:8080` after you have successfully configured and started Tomcat. If you have configured the *Compaq Secure Web Server* to work with Tomcat, you also can access the examples with the *Compaq Secure Web Server* via `http://hostname/examples`.

See the [Building the Sample Web Application on OpenVMS](#) for instructions on how to run the sample application.

Configuring the JServ Servlet Engine with CSWS_JAVA

The default configuration for CSWS_JAVA applies to Tomcat. However, you can still configure the JServ servlet engine with the CSWS_JAVA kit. There are two ways to do this:

- Option 1: Configuring a system that is *not* running Tomcat.

Set default to the JServ directory. The default installation is usually `SY$COMMON:[APACHE.JSERV]`. This example assumes a disk name of DKA200.

```
$ SET DEF DKA200: [APACHE.JSERV]
```

Run the JServ configuration command procedure.

```
$ @CONFIGURE_VMS
```

Note: If you have set up a symbol for EDIT, delete the symbol before running CONFIGURE_VMS.COM.

Add MOD_JSERV.CONF to HTTPD.CONF as follows and save the file.

```
$ EDIT APACHE$ROOT: [CONF] HTTPD.CONF
```

Add the following line to the end of the HTTPD.CONF. (This step was not required in the CSWS_JSERV Version 1.0 kit.)

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

If you have installed CSWS_JAVA into a directory other than where you installed *Compaq Secure Web Server*, perform the following steps. (If you installed CSWS_JAVA into the same directory, skip to Step 5.)

Edit LOGIN.COM for the APACHE\$WWW account.

Add the following lines (before the exit at the top of the file):

```
$ DEFINE/JOB APACHE$JSERV_CONFIG_FILE
APACHE$ROOT: [CONF] -
_$ JSERV.PROPERTIES
$ DEFINE/JOB APACHE$JSERV_DIRECTORY
disk: [directory]
```

where *disk:[directory]* is the location of JSERV.DIR. For example, if you installed CSWS_JAVA into DKA200:[000000], enter:

```
$ DEFINE/JOB APACHE$JSERV_DIRECTORY
DKA200: [APACHE]
```

If, for example, you installed CSWS_JAVA into DKA200:[CSWS_JAVA], enter:

```
$ DEFINE/JOB APACHE$JSERV_DIRECTORY
DKA200: [CSWS_JAVA.APACHE]
```

Update MOD_JSERV.CONF by entering:

```
$ EDIT APACHE$ROOT: [CONF] MOD_JSERV.CONF
```

Change the line that currently reads:

```
LoadModule jserv_module jserv/src/c/mod_jserv.exe
```

Add the full path name to MOD_JSERV.EXE in the preceding line, so that the line now reads:

```
LoadModule jserv_module
/dka200/apache/jserv/src/c/mod_jserv.exe
```

Stop and restart the *Compaq Secure Web Server*.

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

Test the server status page at:

```
http://localhost/jserv/
```

- Option 2: Configuring JServ and Tomcat on the same system.

Tomcat and JServ servlet engines use the same port to communicate with *Compaq Secure Web Server*. You must configure one or the other to listen on a different port. **Perform steps 1 through 3 above**, then perform the following steps:

Edit JSERV.PROPERTIES as follows and save the file.

```
$ EDIT APACHE$ROOT: [CONF] JSERV.PROPERTIES
```

Change the following:

```
# Default: 8007
port=8007
```

To the following:

```
# Default: 8007
port=8010 (any unused port)
```

Edit MOD_JSERV.CONF as follows and save the file.

```
$ EDIT APACHE$ROOT: [CONF] MOD_JSERV.CONF
```

Change the following:

```
# Default: protocol-dependant (for ajpv12 protocol this
is "8007")
ApJServDefaultPort 8007
```

To the following:

```
# Default: protocol-dependant
(for ajpv12 protocol this is "8007")
ApJServDefaultPort 8010
```

Both Tomcat and JServ use the http://nodename/servlet URL, so to avoid confusion, add or change the following lines to MOD_JSERV.CONF.

Add a default ApJServMount directive before the current ApJServMount directives as follows:

```
ApJServMount default /root
```

Then change the following line:

```
ApJServMount /servlet /root
```

To a line similar to the following:

```
ApJServMount /servletjserv /root
```

Edit the JSSI directive to reflect the change made to the ApJServMount directive. Change the following line:

```
#ApJServAction .jhtml
/servlets/org.apache.servlet.ssi.SSI
```

To a line similar to the following:

```
#ApJServAction .jhtml
/servletjserv/org.apache.servlet.ssi.SSI
```

Stop and restart the *Compaq Secure Web Server*.

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

Test the server status page at:

```
http://localhost/jserv/
```

Building the Sample Web Application on OpenVMS

To build the sample web application found in [.jakarta.doc.appdev.sample], perform the following steps.

Set your directory to the sample directory.

```
$ SET DEFAULT DKA200:[APACHE.JAKARTA.DOC.APPDEV.SAMPLE]
```

Enter the following command, where *dka200* is the disk where you installed CSWS_JAVA.

```
$ @sys$startup:apache$jakarta ant "-buildfile" build.xml -
_ $ "dist" "-Dtomcat.home=/dka200/apache/jakarta"
```

You will then see the following output:

```
-> Tomcat Directory /VARMIT$DKA200/APACHE/JAKARTA/
Using Java 1.3.0 setup
Setting up symbols for foreign command line usage...
%DCL-I-SUPERSEDE, previous value of
JAVA$FILENAME_CONTROLS has been superseded
JAVA$FILENAME_CONTROLS now set to: -1
%DCL-I-SUPERSEDE, previous value of
JAVA$FILENAME_CONTROLS has been superseded
Run ANT in Tomcat's environment
Buildfile: BUILD.XML

prepare:
[mkdir] Created dir: /dka200/apache/jakarta/webapps/myapp
[copy] Copying 3 files to /dka200/apache/jakarta/webapps/myapp
[mkdir] Created dir: /dka200/apache/jakarta/webapps/myapp/WEB-
INF
[copy] Copying 1 file to
/dka200/apache/jakarta/webapps/myapp/WEB-INF
[mkdir] Created dir: /dka200/apache/jakarta/webapps/myapp/WEB-
INF/classes
[mkdir] Created dir:
/dka200/apache/jakarta/webapps/myapp/javadoc
```

```
compile:
  [javac] Compiling 1 source file to
  /dka200/apache/jakarta/webapps
  /myapp/WEB-INF/classes

dist:
  [jar] Building jar:
  /dka200/apache/jakarta/webapps/myapp/myapp.jar
  [jar] Building jar:
  /dka200/apache/jakarta/webapps/myapp/myapp.war

BUILD SUCCESSFUL

Total time: 9 seconds
Tomcat Logicals and Classpaths are cleared
```

Edit the file `server.xml`

Edit `/dka200/apache/jakarta/conf/server.xml` so that Tomcat can recognize your application. Add to the file a definition with the Context tag. This enables Tomcat to load your application kept under `jakarta-tomcat/webapps/myapp` upon request to 'myapp'. For example:

```
<Context path="/myapp"
docBase="webapps/myapp" debug="0" reloadable="true">
</Context>

</ContextManager>
</Server>
```

Stop Tomcat if it is running.

Start Tomcat and enter the URL:

```
http://127.0.0.1:8080/myapp/index.html
```

You should see a page with links to a JSP or servlet file. Selecting either page produces a display of the request headers.

Running Tomcat

For information about running Tomcat, see [Tomcat - A Minimalistic User's Guide](#).

Release Notes (Tomcat)

This section contains notes about the **Tomcat component** of the current release of CSWS_JAVA.

- Installation restriction removed

In the CSWS_JAVA beta kit 1, you were required you to install CSWS_JAVA into the same device and directory where you installed the *Compaq Secure Web Server for OpenVMS* if you wanted to use the JServ component in CSWS_JAVA .

This restriction has been removed.

- Java SecurityManager included

Support for the Java SecurityManager is included in Java 1.3. See [Tomcat Security](#) for more information.

- Security vulnerabilities fixed in Tomcat 3.2.1
 - Protection of resources in /WEB-INF and /META-INF directories

The servlet specification prohibits servlet containers from serving resources in the /WEB-INF and /META-INF directories of a web application archive directly to clients. In Tomcat 3.2, this means that URLs similar to:

```
http://localhost:8080/examples/WEB-INF/web.xml
```

will return an error message, rather than the contents of your deployment descriptor. However, there is a vulnerability in Tomcat 3.2 that exposes this information if the client requests a URL like this instead:

```
http://localhost:8080/examples//WEB-INF/web.xml
```

(Note the double slash before "WEB-INF"). This vulnerability has been corrected in Tomcat 3.2.1.

- Show source vulnerability

The example application delivered with Tomcat 3.2 included a mechanism to display the source code for the JSP page examples. This mechanism could be used to bypass the restrictions on displaying sensitive information in the WEB-INF and META-INF directories. This vulnerability has been removed.

- Direct HTTP/SSL support not available

The Java Secure Socket Extension (JSSE) class library is not included in this kit. JSSE is necessary for making direct HTTP/SSL connections within Tomcat using SSLSocketFactory. HTTP/SSL connections between the browser and web server function normally; only direct HTTP/SSL connections made from Java servlets are not supported in this kit.

HTTP/SSL support may be included in a future release.

- Jakarta START -SECURITY problem corrected

In CSWS_JAVA beta kit 1, there was a problem with the JVM security policy handler that was discovered while testing Tomcat on OpenVMS. Java engineering has fixed this problem in the final version of Java 1.3. The policy trace information has been disabled in CSWS_JAVA.

- Setting up Tomcat to use Fast VM

If you want to use Fast VM with Tomcat, download and install the Compaq Fast VM for Java kit from <http://www.compaq.com/java/alpha/index.html>.

Then define the following logical in the .TOMCATRC file:

```
$ define APACHE$JAKARTA_USE_FASTVM true
```

- Using the latest (or beta) version of Compaq Java Development Kit with CSWS_JAVA

You can use the latest customer release or beta version of Compaq Java Development Kit with CSWS_JAVA. To do so, you must run the appropriate Java setup file for *CSWS_JAVA for Compaq Secure Web Server*.

Note: These steps only apply to the Tomcat component in CSWS_JAVA. To set up another version of Java with the Apache JServ component, see [Using the latest \(or beta\) version of Compaq Java Development Kit or Fast VM with JServ.](#)

To use another version of Java, perform the following steps.

Install the desired version of Java. Kits are available from <http://www.compaq.com/java/alpha/index.html>.

Create the setup procedure for the product you want to run with CSWS_JAVA.

This example shows the setup for Java Development Kit Version 1.3.1 with FastVM.

Create the file `sys$startup:apache$setup_jakarta_env.com` containing the following lines:

```
$ @sys$common:[java$131.com]java$131_setup fast
$ define apache$jakarta_user_classpath
sys$common:[java$131.lib]tools.jar
```

Define `apache$jakarta_jdk_setup` to point to this file. `Apache$jakarta_jdk_setup` can be defined in your system startup procedure, `APACHE$ROOT:[000000]LOGIN.COM` (before the line containing `$ @SYS$STARTUP:APACHE$JAKARTA ENV`), or in `.TOMCATRC`.

Define the logical `APACHE$JAKARTA_JDK_SETUP` to point to your system file. You can do this from the system table or the *Compaq Secure Web Server* `login.com` file.

```
$ def/system APACHE$JAKARTA_JDK_SETUP -
_$ sys$startup:apache$setup_jakarta_env.com
```

Shut down and restart the *Compaq Secure Web Server*.

```
$ @sys$startup:apache$shutdown
```

```
$ @sys$startup:apache$startup
```

- Password for admin example

When you try to access the admin example, Tomcat prompts for a password. By default, Tomcat uses `conf/tomcat-user.xml` for its username and password. To access the admin pages, you must have added the "admin" role to any user. For example:

```
<tomcat-users>
  <user name="tomcat" password="tomcat" roles="tomcat,admin" />
```

In addition, you must change the trusted parameter on the `/admin` Context, as follows:

```
<Context path="/admin"
  docBase="webapps/admin"
  crossContext="true"
  debug="0"
  reloadable="true"
  trusted="true">
</Context>
```

- Slow access the first time Tomcat server is invoked

The first time you invoke the Tomcat server, several minutes may pass before you can access `http://hostname:8080`. The reason for this is that Tomcat deploys all of the applications (mostly examples) in the `webapps` directory. This is only done the first time the server is invoked. If you delete the subdirectories in the `webapps` directory, you can avoid the slow first time startup.

- Pause before serving first page after restart

After the Tomcat server is restarted, it pauses before serving the first page. This occurs because the first JSP causes the server to generate a set of secure random numbers, which invokes Java's GC several times in an attempt to get a truly random number. Because Java Version 1.1.8 for OpenVMS and Java Version 1.2.2 for OpenVMS allocate the maximum heap size on startup, serving the first page takes a longer time compared to other platforms. Java Version 1.3.0 for OpenVMS uses a dynamic memory heap, which will speed up this random number generator routine in a future version of `CSWS_JAVA`.

- Configuration dialog question about updating configuration data file

When you run `APACHE$JAKARTA` or `APACHE$JAKARTA_CONFIG`, you see the question "Would you update the Jakarta configuration data file (Yes/No) [Yes]".

This question is asking whether you want the new changes to be reflected in the configuration file (`APACHE$JAKARTA_CONFIG.DAT`). In the future, you might want to have a development Tomcat server and a production Tomcat server on the same system, but with different configuration information for each server.

Release Notes (MOD_JSERV and JServ Servlet Engine)

The following section contains notes about the **JServ component** of the current release of CSWS_JAVA.

- JHTML files must be in STREAM_LF format

If the file is in a format other than STREAM_LF, the JSSI processor will fail.

To convert your file to STREAM_LF, enter the following:

```
$ CONVERT/FDL=STREAM_LF.FDL infile.  
JHTML outfile.JHTML
```

The FDL file should contain the following lines:

```
$ TYPE STREAM_LF.FDL  
  
FILE  
  ALLOCATION           4  
  BEST_TRY_CONTIGUOUS yes  
  EXTENSION          0  
  ORGANIZATION       sequential  
  
RECORD  
  BLOCK_SPAN         yes  
  FORMAT             stream_LF  
  SIZE               0
```

- Delete or move JDK11*_CLASSES.ZIP

If you are installing Compaq Java Development Kit Version 1.1.8 for OpenVMS Alpha on a system that has a previous version of the JDK installed, be sure that you delete or move the previous version of the JDK11*_CLASSES.ZIP file from the SYS\$COMMON:[JAVA.LIB] directory before you begin the installation. Otherwise, problems may occur when you start the JServ process.

- EXAMPLE directory filenames are case-sensitive

The names of the examples in the EXAMPLE directory are case-sensitive. Although the .CLASS files appear in uppercase in the EXAMPLE directory, the URL must specify "Hello" and "IsItWorking" for the examples to work.

- Faster updates to the JServ log file

The JSERV.LOG file remains empty until sufficient data causes the buffer to be flushed. (If the server is shut down, the log file is empty.) If, for example, you are debugging a servlet and you need more timely updates to the JServ log file, you can change the Java mode for flushing and sharing files. To do this, modify the logical JAVA\$FILE_OPEN_MODE in the file START_JSERV_MANUAL.COM (located in APACHE\$COMMON:[000000]).

You can define JAVA\$FILE_OPEN_MODE to 0 (the default - no file sharing), 2 (sync writes, the file is in sync with disk), or 3 (the JDK monitors when an application writes to a

file and flushes writes to the disk before each read operation; but only one process can share the file reliably).

Caution: Do not define JAVA\$FILE_OPEN_MODE to a value of 2 in a production environment. Mode 2 will have an adverse effect on performance.

See the [Compaq Java 2 SDK for OpenVMS Alpha Release Notes](#) for more information.

- Using the latest (or beta) version of Compaq Java Development Kit or Fast VM with JServ

You can use other versions, including the latest customer release or beta version, of Compaq Java Development Kit with JServ. To do so, you must run the appropriate Java setup file for *CSWS_JAVA for Compaq Secure Web Server*.

Note: These steps only apply to the Jserv component in CSWS_JAVA. To set up another version of Java with CSWS_JAVA, see [Using the latest \(or beta\) version of Compaq Java Development Kit with CSWS_JAVA](#). To enable the Fast VM option for Java 1.2.2, see [Setting up Tomcat to use Fast VM](#).

To use another version of Java or Fast VM with JServ, perform the following steps.

Install the desired version of Java or Fast VM. Kits are available from <http://www.compaq.com/java/alpha/index.html>.

Create the setup procedure for the product you want to run with CSWS_JAVA.

This example shows the setup for Java Development Kit Version 1.2.2:

```
$ create/prot=w:re
sys$startup:apache$setup_jserv_env.com
$ @sys$common:[java$122.com]java$122_setup
$ exit
```

This example shows the setup for Fast VM:

```
$ create/prot=w:re
sys$startup:apache$setup_jserv_env.com
$ @sys$common:[java$122.com]java$122_setup fast
$ exit
```

Define the logical APACHE\$JSERV_JDK_SETUP to point to your system file. You can do this from the system table or the *Compaq Secure Web Server* login.com file.

```
$ def/system APACHE$JSERV_JDK_SETUP -
_ $ sys$startup:apache$setup_jserv_env.com
```

Shut down and restart the *Compaq Secure Web Server*.

```
$ @sys$startup:apache$shutdown
$ @sys$startup:apache$startup
```

- Forbidden message when attempting to connect

If you get a Forbidden message when attempting to connect to `http://servername/jserv/`, the reason is that by default, you can only connect to `jserv status Servlet` from `localhost`. If you want to grant access from other hosts, change the following in `MOD_JSERV.CONF` and restart the server:

Original text:

```
# Enable the Apache JServ status handler with the URL of
# "http://servername/jserv/" (note the trailing slash!)
# Change the "deny" directive to restrict
access to this status page.
<Location /jserv/>
    SetHandler jserv-status

    order deny,allow
    deny from all
    allow from localhost
</Location>
```

Change to:

```
# Enable the Apache JServ status handler with the URL of
# "http://servername/jserv/" (note the trailing slash!)
# Change the "deny" directive to restrict
access to this status page.
<Location /jserv/>
    SetHandler jserv-status

    order deny,allow
    deny from all
    allow from .your_domain.com
</Location>
```

- Error when attempting to start JServ

If you receive the following error when attempting to start JServ, check `APACHE$ROOT:[LOGS]` to make sure that your account has RWED access.

```
ApacheJServ/1.1.1
java.io.IOException: Directory not writable:
/apache$root/000000/logs
    at org.apache.java.io.LogWriter.(Compiled Code)
    at org.apache.java.io.LogWriter.(Compiled Code)
    at org.apache.jserv.JServLog.(Compiled Code)
    at org.apache.jserv.JServ.start(Compiled Code)
    at org.apache.jserv.JServ.main(Compiled Code)
```

- Sample Hello.java

To run the `Hello.java` sample program, perform the following steps:

Set up the proper classpath for Java by entering the following commands:

```
$ @SYS$MANAGER:JAVA$SETUP
!$ set up the java symbols for Java 1.1.8
! or use one of the following setup commands:
!$ @SYS$MANAGER:JAVA$122_SETUP
!$ set up the java symbols for Java 1.2.0
!$ @SYS$MANAGER:JAVA$130_SETUP
!$ set up the java symbols for Java 1.3.0
$ SET DEFAULT APACHE$ROOT:[JSERV]
!$ set default to the JServ directory
$ @START_JSERV_MANUAL CLASSPATH_ONLY
!$ we define the proper JAVA$CLASSPATH
```

Compile `Hello.java` by entering the following commands.

```
$ SET DEFAULT [.EXAMPLE]
$ javac Hello.java
```

Restart the server.

Try the example by entering the following URL:

```
http://servername/servlets/Hello
```

servlets is the default `ApJServMount` point if you used `CONFIGURE_VMS.COM`.

Note that the names of the examples in the `EXAMPLE` directory are case-sensitive. Although the `.CLASS` files appear in uppercase in the `EXAMPLE` directory, the URL must specify "Hello" and "IsItWorking" for the examples to work.

- Enabling JSSI support

This kit includes the jar file needed for JSSI support. (Note that support for JSSI only exists in the JServ functionality.)

To enable this support, uncomment the following line from `MOD_JSERV.CONF` and restart the server:

```
#ApJServAction .jhtml /servletsjserv/org.apache.servlet.ssi.SSI
```

- "Internal Server Error" returned under heavy user load

When you run the *Compaq Secure Web Server for OpenVMS* with `MOD_JSERV` (Apache JServ Version 1.1.1), during heavy user load the web server sometimes returns "Internal Server Error." (This may also happen when generating a large report using Java servlets that interact with a database via stored procedures.)

In both the cases the log file shows the following error:

```
java.lang.OutOfMemoryError
    at
org.apache.jserv.JServConnection.processRequest(Compiled Code)
    at org.apache.jserv.JServConnection.run(Compiled Code)
    at java.lang.Thread.run(Thread.java)
```

To correct this problem, increase the amount of memory (java heap) the JServ process uses by performing the following steps.

Enter the following in the JServ parameter file:

```
$ create sys$manager:jserv_parameters.dat
-mx64m
CTRL Z
```

Enter the following commands:

```
$ set file/owner=apache$www sys$manager:jserv_parameters.dat
$ def/system APACHE$JSERV_PARAMETERS_FILE -
_$ sys$manager:jserv_parameters.dat
```

Increase the page file quota (PGFLQUO) to an amount corresponding to the amount you increased the java heap in step 1.

To determine the appropriate value for PGFLQUO, multiply the new java heap amount by 2048. For example, if you increase the java heap to 64m as shown in step 1, increase PGFLQUO by at least 131,072 (64 * 2048). The standard installation value for PGFLQUO is 250,000 pages, so you should set the new PGFLQUO to at least 381,072 pages for the APACHE\$WWW account.

To change the PGFLQUO value, use the system manager account and run the AUTHORIZE utility. For example:

```
$ SET DEFAULT SYS$SYSTEM
$ RUN AUTHORIZE
UAF> SHOW APACHE$WWW
Username: APACHE$WWW                               Owner: APACHE WEBSERVER
...
Maxjobs:      0 Fillm:      300 Byt1m:      200000
Maxacctjobs:  0 Shrfillm:   0 Pbyt1m:      0
Prclm:        8 Dio1m:     300 Pgflquo:    250000
...
UAF> MODIFY APACHE$WWW/PGFLQUO=382000
%UAF-I-MDFYMSG, user record(s) updated
UAF> EXIT
$
```

Note that when increase PGFLQUO, you should monitor the free size of the system page and swap files because they also may need to be increased.

Shut down and then start the *Compaq Secure Web Server for OpenVMS*. (A restart will not start the JServ process.)

```
$ @sys$startup:apache$shutdown  
$ @sys$startup:apache$startup
```
