CHAPTER 8

Using Web Servers and Servlet Engines

Web servers and servlet engines provide basic access to all of the components that make up CiscoWorks Common Services (CWCS). They are critical CWCS components, with special capabilities, limitations, and functions.

The following topics describe how to use CWCS Web servers and servlet engines with your CWCS-enabled applications:

- Understanding the CWCS Web Server and Servlet Engine
- Using CWCS Web Servers and Servlet Engines
- Servlet Engines and Runtime Directory
- Implications of HTML Based Login

For basic information on CWCS Web servers and servlet engines, see the "About Web Server and Servlet Engine Components" section on page 6-6.

For more information about the CWCS Web Server and servlet engines, see:

- *Mjollnir - CMF 2.3 System Functional Specification: (EDCS-283137)*
- *Mjollnir - CMF 2.3 PRD (EDCS-263430) l*

Understanding the CWCS Web Server and Servlet Engine

CiscoWorks Common Services provides a single CWCS Web Server, on which the Tomcat servlet engine runs. The CWCS Web Server uses the Apache Web Server (version 1.3.31.x or later), on both UNIX and Windows platforms, to provide the infrastructure for client/server communication.

The CWCS Web Server services HTTP and HTTPS requests from clients, and is also used to invoke CGI scripts/programs, applets, and servlets. It incorporates a customized access control module (mod_access) that performs session-based access control on every HTTP request. A web request must have a valid Servlet session in order to be processed, with certain exceptions allowed.

Previous versions of CWCS, such as CMF 2.2, supported components (such as the CMF Desktop and the Security system) that required the JRun servlet engine. CMF 2.2 also provided the Tomcat servlet engine for applications such as PIX MC and Kilner. All support for JRun has been dropped in this release of CWCS, and all CWCS components now run under Tomcat only.

The MICE component used for transferring session information is still available to share single session information across multiple servlet engines.

The PSU component uses standard third- party products that are also used by other applications and CWCS service components. This includes the Apache Web Server, Tomcat servlet engine, and the Struts Framework. PSU has a dependency on VDS and the UII, and VDS is dependent on the CWCS Security service.

About the CWCS Web Server and SSL

The CWCS Web Server supports SSL operation, and it is enabled by default. There is an option in CWCS administration to disable or re-enable SSL mode. Note that, whenever SSL is enabled, all of the applications installed on the CWCS Server must work in SSL mode. If any applications installed on the server cannot work in SSL mode, then SSL must be disabled.

SSL is mandatory for the following applications:

- User Login
- MICE (CMFLiasonServlet)
- VMS Bundle applications

Development teams should be aware that other applications using SSL will tend to enable this mode when installed on the same server with your application. The VMS bundle, for example, automatically enables SSL for the CWCS Web Server during installation. The VMS bundle will also disable the user option to enable or disable SSL. Hence, as long as VMS is installed on a server, the server will work in SSL mode only.

Other applications can work either with or without SSL, based on configuration. In these cases, the user will have the option to enable or disable SSL. Note that a limitation of the Tomcat servlet engine and the Servlet Specification will cause any system in non-SSL mode that exposes northbound APIs (carrying user credentials) to also be in HTTP mode.

Using CWCS Web Servers and Servlet Engines

All CWCS components run under the Tomcat Servlet Engine.

About the JRE Version

This release of CWCS supports JRE versions 1.4.1_10 and 1.3.1_06.

Components running under Tomcat are compiled under JRE 1.3.1_06. One package (CSCOjre14) has been added to allow new application code to take advantage of JRE 1.4.1_10.

JRE 1.4.1_10 is used to execute the Tomcat servlet engine.

Applications compiling for JRE 1.3.1_06 must verify compatibility with JRE 1.4.1_10. If you are compiling in 1.4.1_10, then verification is not required.

About Apache Version and Access Control

This release of CWCS supports the same version of Web Server for Tomcat, as was done in CMF 2.2. However, the versions of Apache/ModSSL/OpenSSL must be upgraded to include recent security fixes.

CWCS provides custom enhancements to Apache's access control module (mod_access) to allow session-based access control. These are servlet sessions created when a user logs into CiscoWorks desktop. Except for certain resources, such as the login page, the modified access control module prevents service of HTTP requests that do not have an authenticated session cookie.

Ordinarily, the access-control check will throw a "Forbidden" error page if an attempt is made to access protected resources without logging in. However this was changed in CWCS based on a request from the Okena team. Instead of throwing a forbidden error, the user will be redirected to the login page, if a valid session is not present.

Servlet Engines and Runtime Directory

The runtime directory structure for CWCS components. The same structure can also be extended to CMF 3.0 based applications (e.g. RME 4.0, CM, etc).

For all Tomcat based components and applications, the runtime directory is as follows:

- All the jar files that are common across webapps are located under $NMSROOT/MDC/tomcat/lib/apps
- All the class files that are common across webapps are located under $NMSROOT/MDC/tomcat/lib/apps/classes

Each webapp defines a document base. The document base is a directory under which webapp-specific files are located. The document base is a path relative to the Tomcat home directory:

- Jar files specific for each webapp are placed under $NMSROOT/MDC/tomcat/$document_base/ WEB-INF/lib.
- Class files specific for each webapp will be placed under $NMSROOT/MDC/tomcat/$document_base/ WEB-INF/classes.
- JSP files using UII are located under NMSROOT/MDC/tomcat/$document_base/WEB-INF/screens
- JSP files not using UII are located under $NMSROOT/MDC/tomcat/$document_base/JSP

Runtime Structure for New Components

The fundamental runtime directory structure for this release of CWCS is the same as in CMF 2.2. All components new in this release (such as DCR, CiscoWorks Home page, Device Center, and CMIC) use the Tomcat servlet engine.

In CMF 2.2, the Tomcat root directory was $NMSROOT/MDC/tomcat. It would be ideal if the Tomcat root directory referred to a generic name instead of MDC. However, changing the Tomcat root directory to a new directory (such as $NMSROOT/NG) will involve impacts on all applications based on Tomcat, including MDCs. Therefore, it was decided to address this in a future CWCS release.

CWCS 3.0 has two new webapps based on Tomcat. One webapp includes the CiscoWorks Homepage and the user interfaces for CMIC and DCR. The other is for Device Center. All shared components are placed under $NMSROOT/MDC/tomcat/lib/apps. For example, cmic.jar is deployed under $NMSROOT/MDC/tomcat/lib/apps/. Similarly, any class files that need to be shared are placed under $NMSROOT/MDC/tomcat/lib/apps/classes directory

The new structure is explained in more detail below. The generic structure is described followed by details about individual modules. The structure is based on the Java Servlet Specification from Sun and the directories mentioned follow the Web Application structure specified by Sun. Please refer to http://java.sun.com/products/servlet/download.html to download the latest servlet spec.

Existing Tomcat Based Components/Applications

The runtime structure is unchanged.

Note It is recommended that the existing Tomcat based components also move to the new runtime structure to have uniformity. But this decision is to be made by respective components/application teams.

New Tomcat Based Components/Applications

The runtime directory structure mentioned in this document is referenced by the TOMCAT_HOME environment variable (currently it is $NMSROOT/MDC/tomcat). This document assumes that each application is developed and deployed as a separate webapp. All CWCS components are deployed under a single webapp.

Note We do not recommended that you put any class or jar files under $NMSROOT/lib/classpath or $NMSROOT/www/classpath. These directories will be phased out in future releases of CWCS .

It is not recommended to put any class/jar files under $NMSROOT/tomcat/lib as this directory is used by Tomcat servlet engine.

It is left to the component/application teams to investigate further to find whether their components will be deployed as an individual webapp or clubbed under a common webapp.

Applications deploy their files under webapps/$app_name/ directory.

WAR Files Used by Webapps

WAR files used by webapps are located in $NMSROOT/MDC/tomcat/webapps/

For example, CMF.war file will be deployed under $NMSROOT/MDC/tomcat/webapps/

Jar Files Used by a Single Webapp

Jar files used by a single webapp are located under $NMSROOT/MDC/tomcat/webapps/$app_name/WEB-INF/lib

This could be SRC components used by a particular application. For example, RME using a particular version of UII will be deployed under $NMSROOT/MDC/tomcat/webapps/rme /WEB-INF/lib/uii.jar

Class Files Used by a Single Webapp

Class files used by a single webapp are located under $NMSROOT/MDC/tomcat/webapps/$app_name/WEB-INF/classes

For example, there are some inventory APIs used by various components in RME at $NMSROOT/MDC/tomcat/webapps/rme/WEB-INF/classes/com/cisco/nm/rme/inventory

Jar Files Shared Between Multiple Webapps

These are all shared components (CMF-R) placed under $NMSROOT/MDC/tomcat/lib/apps

For example, cmic.jar, dcr.jar will be deployed under $NMSROOT/MDC/tomcat/lib/apps/

Class Files Shared Between Multiple Webapps

These are class files belonging to shared components placed under

$NMSROOT/MDC/tomcat/lib/apps/classes directory

For example, LogViewer.class will be deployed under $NMSROOT/MDC/tomcat/lib/apps/classes/com/cisco/core/maas/server

JSP Files for Webapps

JSP files for webapps are located under $NMSROOT/MDC/tomcat/webapps/$app_name/screens/

For example, JSP files for DCR are located in $NMSROOT/MDC/tomcat/webapps/cwcs/screens/dcr and for CWHP in $NMSROOT/MDC/tomcat/webapps/cwcs/screens/cwhp and for CMIC in $NMSROOT/MDC/tomcat/webapps/cwcs/screens/cmic.

Action and Form Bean Classes

All Action and Form bean classes for each webapp are placed under $NMSROOT/MDC/tomcat/webapps/$app_name/WEB-INF/classes/com/cisco/nm /$app_name/ui.

Form beans are placed under /form and action classes under /action respectively.

Java Script/Images Files for Webapp

All Java script and images are placed under

$NMSROOT/MDC/tomcat/webapps/$app_name/js and $NMSROOT/MDC/tomcat/webapps/$app_name/images

For example, for CiscoWorks Home Page, Javascript is placed under $NMSROOT/MDC/tomcat /webapps/cwcs/js/cwhp and images under $NMSROOT/MDC/tomcat /webapps/cwcs/images/cwhp

Other Configuration, Properties and Data Files

Other configuration, properties and data files for each webapp are placed under $NMSROOT/MDC/tomcat/webapps/$app_name/etc

For example, CWHP related config files are placed in $NMSROOT/MDC/tomcat /webapps/cwcs/etc/cwhp

Other configuration, properties and data files common across webapps are placed under $NMSROOT/MDC/etc/$component_name/

For example, DCR related config files/ preferences etc placed under $NMSROOT/MDC/etc /dcr

Runtime Structure for CiscoWorks Common Services Webapps

All CiscoWorks Common Services modules providing user interfaces are grouped under the CWCS webapp under Tomcat. These include:

- CiscoWorks Homepage UI, DCR UI, CMIC UI-related classes, JSP and JS files
- CSTM files
- UII files

JSP files related to CiscoWorks Common Services modules appear under subdirectories specific to the modules. For example:

- $NMSROOT/MDC/tomcat/webapps/cwcs/screens/dcr
- $NMSROOT/MDC/tomcat/webapps/cwcs/screens/cwhp
- $NMSROOT/MDC/tomcat/webapps/cwcs/screens/cmic

All Action and Form bean classes are placed under the following directories:

- $NMSROOT/MDC/tomcat/webapps/cwcs/WEB-INF/classes/com/cisco/nm/cmf/cwhp /ui

  /action directory will contain CiscoWorks Homepage UI related action classes

  /form directory will contain CiscoWorks Homepage UI related form bean classes
- $NMSROOT/MDC/tomcat/webapps/cwcs/WEB-INF/classes/com/cisco/nm/cmf/dcr/ui

  /action directory will contain DCR UI related action classes

  /form directory will contain DCR UI related form bean classes
- $NMSROOT/MDC/tomcat/webapps/cwcs/WEB-INF/classes/com/cisco/nm/cmf/cmic/ui

  /action directory will contain CMIC UI related action classes

  /form directory will contain CMIC UI related form bean classes

UII, CTM and OGS are all per product SRC components. So UII will be deployed under $NMSROOT/MDC/tomcat/webapps/cwcs/WEB-INF/lib/uii.jar. Similarly CTM and OGS files (jar files) will be placed under this directory.

Runtime Structure for DCR

DCR (Device Credentials Repository) will be accessed by other applications including RME, PIX MC, IOS MC. DCR will run as a daemon. Therefore, it can be placed under $NMSROOT/lib/server/ or $NMSROOT/objects/server or $NMSROOT/lib/server/dcrserver/dcrserver.jar

DCR Server jar will include OGS and CTM files.

Runtime Structure for CMIC

CMIC provides access mechanism through well-defined APIs that are part of the library files. All components or applications access CMIC using library files. The CMIC registry (database) access is taken care by library files and is transparent to the component users. Therefore, CMIC.jar will be placed under $NMSROOT/MDC/tomcat/lib/apps for applications to use.

Runtime Structure for Device Center

Device Center will be a webapp under Tomcat: $NMSROOT/MDC/tomcat/webapps/devicecenter/

JSP files related to Device Center will be placed under:

- $NMSROOT/MDC/tomcat/webapps/devicecenter/screens/devicecenter

All Action and Form bean classes are placed under:

- $NMSROOT/MDC/tomcat/webapps/devicecenter/WEB-INF/classes/com/cisco/nm/cmf//devicecenter/ui

  /action directory will contain CMIC UI related action classes

  /form directory will contain CMIC UI related form bean classes

Implications of HTML Based Login

The previous applet-based login panel was a bottleneck in terms of the time it took to load the login page and for the username and password fields to appear. The applet-based login used the heavyweight JAAS GUI mechanism to render the login panel. There was a lot of customer feedback asking for a reduction in the time taken for the login panel to appear.

For these reasons, the JAAS GUI mechanism was discarded and a simple HTML-based login panel, generated via a JSP page, was substituted. A JSP page is required to render the login panel based on the login module that is selected. Eliminating the JAAS GUI mechanism and the SPS transport eliminated a bulky set of classes from the login panel.

To ensure secure transport of the login credentials, CWCS must use SSL while submitting login information. The applet-based login panel used a proprietary secure-transport mechanism called SPS (Secured Packet Stream) that did not require SSL. But for HTML login, the SSL port is open in non-SSL mode also, to accept login requests. The HTML login panel requires the SSL port to be always open. This means that an SSL certificate must be generated for the CWCS Web Server at install time.

However, asking the user for certificate information twice is not acceptable, so the Core Apache install script has been changed to generate a certificate for CWCS.