InterSystems™
IRIS Data Platform

Web Gateway Guide

Version 2023.1
2024-07-11

InterSystems Corporation 1 Memorial Drive Cambridge MA 02142 www.intersystems.com

Web Gateway Guide
InterSystems IRIS Data Platform Version 2023.1 2024-07-11

For Support questions about any InterSystems products, contact:

InterSystems Worldwide Response Center (WRC)
Tel: +1-617-621-0700
Tel: +44 (0) 844 854 2917
Email: support@InterSystems.com

Table of Contents

1

Introduction to the Web Gateway

Any InterSystems IRIS® web application relies on three software components that collectively handle HTTP requests and responses for that application:

- The web server is a software utility that receives incoming HTTP requests and determines how to handle them. InterSystems supports several different web servers, varying by operating system.
- The *InterSystems Web Gateway* is a software utility that receives HTTP requests from the web server and then sends them to the appropriate InterSystems IRIS server.

  Depending on the platform, the Web Gateway is a shared library, a `.dll` file, or a CGI script. You can install the Web Gateway as part of InterSystems IRIS or as a standalone utility, depending on your architecture needs.

  The Web Gateway includes management pages via which you can perform global configuration, configure servers, configure web applications, and monitor the Web Gateway.
- The *CSP server* receives HTTP requests from the Web Gateway and then handles them by calling code within InterSystems IRIS. The CSP server is actually a process running within an InterSystems IRIS server. This process is dedicated to serving requests received from the Web Gateway.

1.1 Purposes of the Web Server, Web Gateway, and CSP Server

The web server does the following:

- Accepts incoming HTTP requests, usually from browsers.
- Checks permissions.
- Possibly serves static content (see the next item).
- Sends requests for specific forms of URLs to the Web Gateway. This is determined by the configuration of the web server. In all cases, URLs with the following endings should be processed by the Web Gateway:

  ```
  .csp .cls .zen .cxw
  ```

  The first three extensions refer to kinds of code that can run in InterSystems IRIS. The last one (.cxw) is a reserved extension used by the Web Gateway management pages. Within the context of the Web Gateway, these extensions represent the *InterSystems file types*.

The web server can be configured to send additional types of files to the Web Gateway.

The Web Gateway does the following:

- Determines which InterSystems IRIS server to send a request to.
- Sends requests to the correct InterSystems IRIS server, specifically to the CSP server in that InterSystems IRIS server. (For URLs that end with .cxw, the Web Gateway instead invokes the separate Web Gateway management module.)
- Maintains connections to the InterSystems IRIS server, to avoid having to continually open new connections.

Within InterSystems IRIS, the CSP server does the following:

- Receives an HTTP request for an application.
- Checks the configuration settings for the web application, which are specified in the InterSystems Management Portal and saved in the CPF file for InterSystems IRIS.
- Executes a callback in the requested code, which sends HTML back to the Web Gateway, which sends it to the web server and back to the browser.

1.2 How the Web Server, Web Gateway, and CSP Server Work Together

When a client requests information from a web application, the flow of information is as follows:

1. An HTTP client, typically a web browser, requests a page from a web server using the standard HTTP (or HTTPS) protocol.
2. The web server recognizes this as a CSP request and forwards it to the Web Gateway using a fast server API.
3. The Web Gateway determines the InterSystems IRIS server to talk to and forwards requests to the CSP server on that target system.
4. The CSP server running in InterSystems IRIS processes the request and returns a page to the Web Gateway, which passes it back to the web server.
5. The web server sends it to the browser for display.

1.3 Structure of an InterSystems Web Application URL

The Web Gateway allows you to serve InterSystems IRIS web applications at URLs which follow the following format:

```
[protocol]://[hostname]:[port]/[instancePrefix]/[appPath]/[fileOrQuery]
```

An InterSystems IRIS instance usually hosts multiple web applications. The URLs for all the web applications which are hosted by an instance share in common the *base URL* for the instance. The base URL is the following portion of an InterSystems web application URL:

```
[protocol]://[hostname]:[port]/[instancePrefix]
```

The following table describes each part of these URLs.

URL Part	Description
[protocol]	`http` or `https`, depending on whether or not you have configured your web server to use TLS. InterSystems strongly recommends the use of TLS.
[hostname]	The IP address or DNS name where your web server is available. When the web server is on the same machine as the client, this is usually `localhost`.
[port]	The port number over which your web server is listening for requests. If you are serving your instance's web applications using the web server installed with your instance, this is the port at which that private web server is listening. By default, the installer assigns port 52773 to the web server for the first instance on a given system. For each additional instance you install, the default port for the private web server increments: 52774, 52775, and so on. If you are serving your instance using an external web server and your web server is listening over the standard ports for HTTP (80) or HTTPS (443), you generally do not need to specify the port number when constructing a URL.
[instancePrefix]	An optional string which uniquely identifies one of your InterSystems IRIS application servers. If you serve multiple InterSystems IRIS instances using a single web server and you must access a web application unique to one instance (such as an instance's Management Portal), you must configure your web server and Web Gateway to route requests to the instance using this identifier as a prefix to the application path. This identifier is the *CSPConfigName* for the instance. By default, an instance's *CSPConfigName* is its instance name, in all lowercase characters. However, it can be configured. If the application in the preceding example were exclusively available on the InterSystems IRIS application server named `iris2`, then you could configure the /iris2 path within the web server and Web Gateway configurations to serve the application at the following URL: https://yourhost.com/iris2/criticalapp/MainDashboard.csp
[appPath]	The relative path unique to the application within each InterSystems IRIS application server. When you define an application within an InterSystems IRIS application server, this is the application's **Name**.
[fileOrQuery]	*Optional*. Any combination of subordinate path, file name, and query parameters which the application may use to process the request.

2

Supported Configurations for the Web Gateway

This page provides an overview of the supported configurations for the InterSystems Web Gateway, which is responsible for communications between your web server and InterSystems IRIS® — specifically for communications with web applications and with the Management Portal.

Note: InterSystems IRIS provides the private web server (a minimal build of Apache), for the sole purpose of running the Management Portal. For any production system, you must instead install and configure one of the supported web servers, any of which is more secure and more robust.

2.1 Supported Configurations

During development, a web server, a Web Gateway, and an InterSystems IRIS instance may be on a single machine. In a large scale deployment, there may be multiple web servers and multiple InterSystems IRIS instances, all on different machines. Each web server requires a dedicated Web Gateway. A single Web Gateway can communicate with multiple InterSystems IRIS instances.

The InterSystems IRIS Gateway Registry enables you to monitor and manage the connections. All web server installations and Web Gateway installations are registered with InterSystems IRIS as they connect. The registry contains the infrastructure to allow InterSystems IRIS code to interact with connected Web Gateway installations for the purpose of reading and writing the configuration and monitoring the system status and Web Gateway Event Log.

2.2 Supported Web Servers

You can use the following web servers with the InterSystems Web Gateway in a production environment:

- Microsoft Windows:
  - Microsoft IIS. The connectivity options include the default IIS option and several alternative options for IIS 7 or later.
  - Apache. The connectivity options include the default Apache option and several alternative options for Windows Apache.

  - Nginx

- Linux, UNIX®, and macOS:
  - Apache. Several connectivity options are available: the default Apache option, the NSD, locked-down Apache web server, and several other atypical Apache configurations.
  - Nginx

Note: This list does not include the private web server, which is not intended for use in a production system.

More detailed information on supported web servers can be found in Supported Web Servers in *InterSystems Supported Platforms*.

The Web Gateway provides high-performance connectivity solutions for Microsoft, Apache, and Nginx web servers. In addition to these solutions, connectivity to InterSystems IRIS through the CGI is available for all supported operating systems.

Microsoft web servers support a multi-threaded API which allows extensions, in the form of dynamically bound libraries, to be made to the web server's core functionality. Current versions of the Web Gateway make full use of these APIs in order to bring high-performance web connectivity to the InterSystems IRIS system. The Windows version of Apache also operates in an exclusively multi-threaded mode and, as such, can also take advantage of the Web Gateway implemented as a dynamically bound library.

The UNIX® versions of Apache are architecturally different from the Microsoft Windows based web servers in that they are not exclusively multi-threaded. Apache version 2.4 is implemented using a hybrid model made up of threads and multiple processes. In this model, each UNIX® process is effectively a multi-threaded server in its own right.

The Apache web server publishes a proprietary API in addition to supporting extensions implemented as CGI modules. Extra functionality can be added to Apache by means of user-defined modules (compiled C programs). In fact, a large part of Apache's core functionality is implemented as a set of modules. You can add modules to Apache by one of two methods. First, the source to the module can be compiled directly into the Apache core. This option arguably offers the best performance but, unfortunately, involves reconfiguring and rebuilding the web server. As an alternative to building the module source directly into the Apache core, Apache versions 1.3 onwards support extensions implemented as dynamically linked libraries. This facility allows you to take advantage of the high performance of Apache modules without the need to physically build the module into the core of Apache. The CSP module is distributed as a Windows *Dynamic Link Library* (DLL), and as a UNIX® *Dynamic Shared Object* (DSO). UNIX® Shared Objects are conceptually similar to a Windows Dynamic Link Library (DLL) and are linked at run time. The overhead involved in linking to a library at run time is very low on modern operating systems.

Unlike other web servers, Nginx is based on an asynchronous event-driven architecture. With the event-driven architecture, notifications or signals are used to mark the initiation and completion of each individual operation. A consequence of this design is that while web requests are being processed, resources can be temporarily released and used by other operations. Resources can be allocated and released dynamically and are only associated with the processing of a web request while they actually required. This leads to a highly optimized use of memory and CPU. The asynchronous nature of this architecture results in threads executing concurrently without blocking each other, thus further enhancing the sharing of resources that might otherwise be associated with a thread waiting on a blocking operation. Nginx is supplied with an API to allow extensions, such as CSP, to be added to its core functionality. However, unlike other web servers, extension modules must be built into the web server core at compilation time. Nginx does not support dynamically loaded extension modules.

An alternative architecture is also provided in which the functionality of the Web Gateway is implemented as a stand-alone executable, operating in its own process and not directly connected to a web server. This version of the Web Gateway is known as the NSD. In this context, the NSD is responsible for providing the Web Gateway's core functionality and maintaining persistent connections to InterSystems IRIS. The web server communicates with the NSD via small modules of which there are two types: modules that work to the hosting web server's proprietary API and modules implemented as CGI executables. The NSD-based architecture is therefore used in cases where there is a requirement to extend the web server by means of the CGI standard, or in cases where it is desirable to disengage the functionality of the Web Gateway from that of the hosting web server.

3

Installing the Web Gateway

You can install the Web Gateway as part of an InterSystems IRIS installation or standalone, depending on your configuration needs.

3.1 Default Web Gateway Installation and Configuration

When you install InterSystems IRIS and choose a setup type of **Development**, **Server**, **Custom** (selecting the **Web Server Gateway** component), or **Web Server** (Windows platforms only), the installer configures your existing production web servers (if the default types: IIS on Windows and Apache on Linux/UNIX®/macOS) and then installs the Web Gateway. If this is suitable for your use case, installing the web server and then installing InterSystems IRIS provides a working Web Gateway configuration for InterSystems IRIS instances without requiring any adjustment.

However, if you are using a different combination of web server and platform, have an atypical web server architecture, or are an advanced user who wants to customize your environment, be sure to review this document, which provides procedures for configuring a web server and the Web Gateway to connect to InterSystems IRIS and for using the services that the Web Gateway provides.

In addition to the InterSystems file types (.csp, .cls, .zen, and .cxw), a web server hosting the Management Portal must also route requests for the following additional file types through the Web Gateway: .jpg, .gif, .png, .svg, .css, and .js. See Configuring Apache to Pass Additional File Types to CSP (for Apache) or Using the NSD with Nginx (for Nginx).

3.2 Web Gateway Management Modules

Web Gateway architectures that work directly with a hosting web server's API typically consist of two modules (whose filenames vary by operating system):

- A management module (for example, CSPmsSys.dll), which provides the Web Gateway management pages.
- The runtime module (for example, CSPms.dll), which is responsible for processing requests for CSP files and for loading and routing management requests to the management module.

These files must be in the same directory.

3.3 Web Gateway Components and Physical Installation Paths

Later sections in this guide describe how the Web Gateway components should be configured with all supported web servers. Also, the InterSystems installers create and maintain separate Web Gateway installations for the Private Web Server and any third-party web server that might be present on the same host. In this context, *third-party web server* refers to a web server that is not part of the software installed by InterSystems.

The precise installation location of Web Gateway components is not particularly critical provided:

- The physical installation paths match those given in the hosting web server configuration where appropriate.
- The security settings, in relation to required access for individual components, are adjusted appropriately. This is particularly important for Web Gateway components that are accessed directly by the web server because web servers are usually locked down to the extent that the files they are able to access (and executables that can be run) are carefully controlled. You should bear in mind that security considerations are also important for any Web Gateway configuration (and log) files that are accessed by Web Gateway binaries that are themselves bound to the web server core executable.
- The security policy of the hosting web server is respected. Some web servers – notably those shipped with Secure Linux (SELinux) – are configured such that it is not possible for them to access files that lie outside their own file system. This restriction has an impact on *where* certain web-server-facing Web Gateway components can be installed.

There are four types of Web Gateway component to consider.

1. Binaries to be loaded by the web server (API based extensions).

   This includes Windows DLLs, and UNIX® Shared Objects:

   ```
   CSPms[Sys].dll
   CSPn3[Sys].(dll|so|exe)
   CSPa*[Sys].(dll|so)
   CSPx[Sys].(dll|so)
   mod_csp*.(dll|so)
   ```

   The physical location where these are installed should match the corresponding configuration directives in the hosting web server configuration. This includes directives indicating which third-party modules should be loaded. The web server requires permission to read and load these modules. Modules named `CSP*` require permission to read and write to the Web Gateway configuration and log files. These are usually created in the same location as the Web Gateway binaries.

   When considering access control for these modules, bear in mind that it is the web server *worker processes* that need to be able to access the modules together with any dependent configuration and log files. For example, in the case of Apache, the server is usually started with superuser permissions but the worker processes that actually serve web requests run with a much lower level of authority (as indicated by the User and Group directives in the Apache configuration file). It is the User and Group specified for the worker processes that should be granted permission to load the Web Gateway modules and (where appropriate) the ability to read and write to the configuration and log files.

2. Executables to be called by the web server (*Common Gateway Interface* (CGI) modules). Not all configurations require these executables.

   ```
   [nph-]CSPcgi[Sys][.exe]
   ```

   The physical location where these are installed should match the corresponding configuration directives in the hosting web server configuration. This includes directives indicating which web requests should be processed by these CGI modules.

The worker processes of the hosting web server require execute permission for these modules. There are no further dependencies.

3. Static files to be returned by the web server.

   Note: With current Web Gateway configurations, CSP is often configured to serve static files directly from InterSystems IRIS as opposed to having the web server return them. This page does not apply to such configurations.

   JavaScript modules (such as CSPBroker.js, CSPxmlhttp.js, and so on)

   Java applets (such as CSPBroker.class, CSPBroker.jar, and so on)

   Images (such as created-with-csp.gif, and so on)

   The worker processes of the hosting web server require Read permissions for these files.

4. The CSP *network service daemon* (NSD).

   Note: Not all configurations require this facility.

   ```
   CSPnsd[Sv][.exe]
   ```

   The NSD can be installed anywhere and the web server does not need to be aware of its physical location since communication between these two points is over TCP, usually port 7038.

   The NSD requires permission to read and write to the Web Gateway configuration and log files, which are usually created in the same location.

   Note: For security reasons, do *not* install this module in a location that is accessible by the web server. This module should not share a location with the modules listed in steps 1, 2 or 3. Many web server configurations described in this document explicitly exclude this module from the list of accessible files that can be accessed by the web server. However, it is much safer to physically install the NSD elsewhere in the file system.

5. The Web Gateway cache, kept in persistent storage.

   The Web Gateway, where possible, locates the content of large files (such as large JavaScript files) together with the accompanying HTTP response headers in permanent storage. The essential control information (expiry time and so on) and the index of all cached files is in the shared memory sector. With this architecture, each cache entry consumes a small amount of cache space (in terms of memory usage) — quite possibly no more than one cache block per entry.

   Cached content is stored in files of type .dat in the Web Gateway's temp directory, placed by the install script directly beneath the Web Gateway's installation directory. For example, in a typical IIS installation, this is in: C:\Inetpub\CSPGateway\temp. The location needs full read/write/delete permissions for the hosting web server worker processes.

3.4 Deploying the Web Gateway as a Stand-alone Component

In some cases, such as when you want to locate the Web Gateway on the system hosting the web server for use by one or more remote InterSystems IRIS instances, you can deploy the Web Gateway as a stand-alone component in the following ways:

- As described in Setup Type in the Preparing to Install InterSystems IRIS chapter of the *Installation Guide*, you can use the InterSystems IRIS installer to install the Web Gateway separately by selecting a **Web Server** setup on Microsoft Windows systems, or by selecting a **Custom** setup on any platform and including the **Web Server Gateway** component.
- You can also install the Web Gateway independently using a stand-alone installer. To get the installer, use the following WRC download page: https://wrc.intersystems.com/wrc/coDistGen.csp. To show only the Web Gateway kits, type `web gateway` in the **Name** column.

  As with the InterSystems IRIS installers, the Web Gateway stand-alone installer is provided as an executable and the UNIX® stand-alone installer is provided as a command line script.

  The defaults for the Windows stand-alone installer assume that you are using the IIS web server, and the defaults for the UNIX® stand-alone installer assume you are using the Apache web server. If you have an atypical web server architecture, or want to customize your environment, you should read this document carefully to understand the configuration options.
- The **`webgateway`** container image available from InterSystems includes both the Web Gateway and an Apache or Nginx web server, thereby providing a containerized web server for InterSystems IRIS-based applications. For more information about deploying Web Gateway containers from the **`webgateway`** image, see Using the InterSystems Web Gateway Container in *Running InterSystems Products in Containers*, Role WS: Web Server in the *InterSystems Cloud Manager Guide*, and webgateway: Define web server nodes in *Using the InterSystems Kubernetes Operator*.

Note that you need a separate installation of the Web Gateway for each web server. For more information on configuring a remote web server, see Using Web Applications with a Remote Web Server.

4

Using or Replacing the Private Web Server

InterSystems IRIS® provides the *Private Web Server*, a minimal build of Apache, for the sole purpose of running the Management Portal.

Important: For deployments of HTTP-based applications, including REST, CSP, Zen, and SOAP over HTTP or HTTPS, you should not use the private web server; instead, you must install and configure one of the supported web servers.

4.1 Purpose of the Private Web Server

The purpose of the *Private Web Server* (PWS) is only to connect to InterSystems IRIS and to meet its management needs. This private version of Apache is installed to ensure that:

- The Management Portal runs out of the box.
- An out-of-the-box testing capability is provided for development environments.

The Private Web Server is self contained and configured to listen on a TCP port other than the usual, well known, HTTP server port of 80. It does not interfere with any other web server installation operating on the same host.

The entry point for the Management Portal is normally via the following CSP path and file: /csp/sys/UtilHome.csp. For example: http://127.0.0.1:52773/csp/sys/UtilHome.csp

4.2 Limitations of the Private Web Server

Regarding the private web server:

- It is not suitable for production use.
- It uses minimal security, and is built and configured exclusively for use within a secured environment.
- It should only be used in environments where there is no public access to the Management Portal and in which all users are both trusted and connecting to the server through secured connections.

Additionally, the configuration deployed is generally unsuitable for applications for which a high volume of HTTP requests is anticipated. InterSystems tests the private web server only for use with the Management Portal.

This section discusses the differences between the configuration of the private web server and that of a typical production-grade Apache installation.

4.2.1 Limitations on Windows

Windows-based Apache installations use a special multi-threaded form of the Apache Multi-Processing Module (MPM) which is better suited to the way the operating system is optimized. Therefore, the behavior of the private web server under Windows is similar to that of a production grade Apache build as far as the ability to handle concurrent load is concerned.

If high availability and production-grade security is a requirement, or there is a need to integrate with other sources of web information, or a need for a high degree of control over the web server, a separate production-grade build of Apache is recommended — ideally operating on its own server. If, on the other hand, low volumes of HTTP traffic are expected, and there are limited demands for high availability and security, then the private web server may be suitable for deployment under these circumstances.

4.2.2 Building the Private Web Server

The (default) full Apache server is usually created with the following sequence of commands:

```
./configure --prefix=<install-dir>
make
make install
```

The minimal Apache build is typically created as follows:

```
./configure --prefix=/usr/iris/httpd --with-port=57773
            --with-pcre=$srcdir/pcre
            --enable-mods-static="log_config mime alias unixd authz_core"
            --disable-ssl
            --enable-so --without-gdbm --without-ndbm
            --without-berkeley-db --with-included-apr --with-expat=builtin
            --with-mpm=prefork --disable-shared
make
make install
```

Notice that many of the services that are normally required for a production grade installation are excluded.

While this server can be used to host other web applications it is strongly recommended that a full, independent web server installation is used for this purpose. It should be remembered that any changes made to the configuration of the Management Portal Apache installation are overwritten when the hosting InterSystems IRIS installation is upgraded.

4.2.3 Limitations on UNIX®

The private web server defaults to using the Apache Group’s prefork Multi-Processing Module (MPM). This is a non-threaded server model: the number of requests that can be concurrently served is directly related to the number of Apache worker processes in the pool.

The private web server is configured to occupy the smallest possible footprint by allowing a maximum of two worker processes to be created for the pool. The following settings are found in the Apache configuration (httpd.conf) for the private web server:

```
MinSpareServers 1
MaxSpareServers 2
```

By contrast, the default Apache configuration for a production grade build is usually as follows:

```
StartServers       5
MinSpareServers    2
MaxSpareServers   20
ServerLimit      256
MaxRequestWorkers       256
```

This configuration allows Apache to create 5 worker processes at start-up, increasing to a maximum of 256 as the concurrent load increases. Because of these differences in configuration, the performance of the private web server is noticeably inferior to that of a production grade Apache build. This performance deficit becomes more noticeable as the concurrent load increases. However, it is possible to change the configuration of the private web server to match that of a full Apache installation (shown above). Apache must be completely restarted after changing these parameters.

4.2.4 Note

The Management Portal Apache installation uses the following Web Gateway modules for communicating with InterSystems IRIS:

- *Windows*: CSPa24.dll and CSPa24Sys.dll
- *UNIX®*: CSPa24.so and CSPa24Sys.so

4.3 Using a Different Web Server to Run the Management Portal

You *can* use a different web server to support the Management Portal.

- If you are using a web server other than the private web server to manage an instance of InterSystems IRIS, you must configure the web server so that links to the documentation continue to work. To do this, configure the web server so that it includes a redirection from /csp/docbook/ to the correct URL for the documentation. You can find this information in the file *install_dir*/httpd/conf/httpd-doc.conf, which Apache uses to redirect /csp/docbook/. For information on creating a redirection, consult the documentation for the web server that you are using.
- To disable the private web server, set the WebServer CPF parameter equal to `0`.

If you want to serve the Management Portal with a standalone Apache, you need to manually configure additional file types. See Configuring Apache to Pass Additional File Types.

4.4 Managing the Private Web Server

Under normal operational conditions, the private web server for a particular instance of InterSystems IRIS is started when InterSystems IRIS is started and closed down when InterSystems IRIS is closed down. Occasionally it may be necessary to restart the private web server without disrupting the corresponding InterSystems IRIS server, for example, if you make a configuration change to the web server.

4.4.1 Making Configuration Changes

The configuration file for the private web server is located at: *install_dir*/httpd/conf/httpd.conf. However, edits made to this file do not persist through a software upgrade. The configuration file *install_dir*/httpd/conf/httpd-local.conf is provided for configuration changes you want to retain on upgrade. It is created on a new install or on upgrade if it does not already exist. If it does exist, upgrade makes no changes to its content.

4.4.2 Managing the Private Web Server on Windows

Start the private web server:

```
<install-dir>\httpd\bin\httpd -k start -n <instname>httpd
  -c "Listen port"
```

Stop the private web server:

```
<install-dir>\httpd\bin\httpd -k stop -n <instname>httpd
```

For example, suppose that InterSystems IRIS is installed in: C:\iris

InterSystems IRIS instance name: `IRIS`

TCP port for Apache: `52773`

Start:

```
C:\iris\httpd\bin\httpd -k start -n IRIShttpd -c "Listen 52773"
```

And you can stop it as follows:

```
C:\iris\httpd\bin\httpd -k stop -n IRIShttpd
```

4.4.3 Managing the Private Web Server on UNIX®

Start the private web server:

```
<install-dir>/httpd/bin/httpd -d <install-dir>/httpd
  -c "Listen port"
```

Stop the private web server:

```
kill `cat <install-dir>/httpd/logs/httpd.pid`
```

For example, suppose that InterSystems IRIS installed in /usr/iris, and the TCP port for Apache is `8972`. In this case, you can start the Private Web Server as follows:

```
/usr/iris/httpd/bin/httpd -d /usr/iris/httpd -c "Listen 8972"
```

And you can stop it as follows:

```
kill `cat /usr/iris/httpd/logs/httpd.pid`
```

Note: On AIX, *LD_LIBRARY_PATH* must include the *install-dir*/bin directory in order to manually run httpd in this way.

5

Configuring IIS to Work With the Web Gateway (Windows)

This page describes how to configure an existing Microsoft IIS web server for use with the InterSystems Web Gateway. (On Windows, the other options are Apache and Nginx; options are different on other operating systems.)

When you install the Web Gateway using the Web Gateway standalone installer, you can select the option to automatically configure an existing Microsoft IIS web server. If you do not choose this option, you can configure your web server manually as described in this page.

5.1 Common Steps (All Versions)

If you are using IIS as your web server, follow these steps to configure your web server.

1. Set permissions for the Web Gateway components.
2. Configure the web application path.
3. Enable URLs with /bin.

Then if you are using IIS 7 or later, see Microsoft IIS 7. If you have an atypical configuration, see Alternative Options for IIS 7 or Later.

5.1.1 Setting Permissions for the Web Gateway Components

Regardless of which Web Gateway configuration option you choose, you need to assign appropriate permissions to web resources held outside the standard IIS documents root (for example, C:\inetpub\wwwroot).

IIS 7 does not, by default, allow the user of a web application to access anything outside the scope of the pre-configured documents root unless you assign Read & Execute and Write permissions for those external resources to the following user or groups:

[machine_name]\IIS_IUSRS

And:

[machine_name]\Users

Note that **IIS_IUSRS** represents the user (group) under which IIS worker processes operate. It replaces the **IUSR_[machine_name]** user group found in IIS versions earlier than version 7. Applications controlled through IIS (such as the Web Gateway) operate with the level of privilege assigned to **IIS_IUSRS**.

For CSP, resources external the web server's root usually include the following:

Web Gateway binary components:

C:\Inetpub\CSPGateway

Static file components:

install-dir\CSP\

Permissions can be manually assigned to these folders via Windows Explorer as follows:

1. Right select the folder name and select **Properties**.
2. Select the **Security** tab.
3. Select **Edit**.
4. Select **Add**.
5. In the **Enter the object names to select** text box enter:

   ```
   [machine_name]\IIS_IUSRS
   ```
6. Select **Check Names** and **OK**.
7. Select **[machine_name]\IIS_IUSRS** in the **Group or Usernames** window, then:
8. Assign **Read & Execute** and **Write** permissions in the **Permissions** window.
9. Select **Apply** and **OK**.
10. Repeat the above process for the **[machine_name]\Users** user group.

The IIS user group requires full read and write permissions for the Web Gateway configuration and log files. For example, at the Windows command prompt, enter:

```
cacls CSP.ini /E /G IIS_IUSRS:F
```

```
cacls CSP.log /E /G IIS_IUSRS:F
```

Of course, this can also be done via Windows Explorer.

5.1.2 Configuring the Web Application Path

This section describes the procedure for configuring the web application path (such as csp) for IIS. These procedures are common to all Web Gateway configuration options for IIS.

IIS is configured in the **Internet Information Services (IIS) Manager** control panel. Subdirectories configured under the documents root can be classed as either **Virtual** or **Applications**. **Virtual** subdirectories (or aliases) are mapped to physical equivalents (Windows directories). The same applies to subdirectories classed as **Applications** except that, in addition to defining the physical equivalent, you can associate the application with a particular application pool. The default is **DefaultAppPool**.

Since web applications are served through the Web Gateway, the hosting subdirectories (such as /csp) should be configured as **Applications**.

In a default CSP configuration, the /csp application path is mapped to the physical location *install-dir*\CSP. All the static files are located under this root (\csp\broker…).

1. Open the **Internet Information Services (IIS) Manager**.

2. In the left panel, expand the top level to reveal the **Web Sites** section, then the **Default Web Site** section. Highlight the **Default Web Site** section:

   ```
   [MACHINE_NAME] ([machine_name]\[user_name])
               Web Sites
                       Default Web Site
   ```

3. In the right panel, select **View Applications**.
4. Again in the right panel, select **Add Application**.
5. In the **Add Application** dialog, enter:

   Alias: **csp**

   Physical path: *install-dir*\CSP\

6. Select **OK**.

If you are using a Web Gateway solution based on an alternative option, set up an application called /bin under the /csp application. Map this to the physical directory holding the Web Gateway binaries. For example:

Map application /csp/bin to C:\Inetpub\CSPGateway

5.1.3 Enabling URLs with /bin

If you installed the Web Gateway using the InterSystems IRIS installer, this step was done automatically for you. If you are installing the Web Gateway manually, you need to do this step. (See this external web site for more details and alternative ways to accomplish this https://weblogs.asp.net/owscott/iis7-blocks-viewing-access-to-files-in-bin-and-other-asp-net-folders.) To enable URLs that contain /bin, add the following location tag to your applicationHost.config file:

```
<location path="sitename.com/subfolder/bin/debug">
    <system.webServer>
        <security>
           <requestFiltering>
              <hiddenSegments>
                 <remove segment="bin" />
               </hiddenSegments>
           </requestFiltering>
        </security>
     </system.webServer>
 </location>
```

5.1.4 Restarting IIS

This section describes what happens when IIS is restarted via the various control panels.

Most configuration changes can be made in real-time to an active IIS installation. However, the **Internet Information Services (IIS) Manager** control panel provides stop, start, and restart options. These are useful for the refreshing the web server configuration but does not result in an active Web Gateway installation being reinitialized (the Web Gateway DLLs are not reloaded).

To force IIS to restart, so that the Web Gateway modules are reloaded, then restart the **World Wide Web Publishing** service via the main Windows **Services** control panel.

5.1.5 Troubleshooting

This section describes problems that commonly occur in configuring third-party modules (both Native and ISAPI) to work with IIS.

The most common problem likely to be encountered is that, after reconfiguring, requests to IIS fail with the following error:

```
Service Unavailable
```

```
HTTP Error 503. The service is unavailable.
```

This usually indicates that the default **Application Pool** has terminated.

1. Open the **Internet Information Services (IIS) Manager** window.
2. In the left panel expand the top level to reveal the **Application Pools** section.

   [MACHINE_NAME] ([machine_name]\[user_name])

   Application Pools

3. Check that the Default Application Pool (`DefaultAppPool`), or whatever application pool your server is configured to use, is marked with a Status of **Started**.
4. Restart the application pool if necessary (using the options in the right panel).
5. If problems persist, look for clues in the main Windows Event Log (in the **Applications** section). In particular, check for the following error message:

   ```
   Failed to find the RegisterModule entrypoint in the module DLL
   C:\Inetpub\CSPGateway\CSPms.dll. The data is the error.
   ```

   This, for example, indicates that the version of Web Gateway DLLs that you are using do not implement the Native Modules interface. Either obtain later DLLs from InterSystems or configure the Web Gateway to work through the conventional ISAPI interface.

As with all software, restarting often clears transient problems: To completely restart IIS, restart the **World Wide Web Publishing** service via the main Windows Services control panel.

Do not use the **Add Wildcard Script Map** utility to map file extensions. If you do, you may see this error: `The specified module required by the handler is not in the modules list. If you are adding a script map handler mapping, the IsapiModule or the CgiModule must in the modules list.` Instead use **Add Module Mapping for *** to map file extensions using a wildcard.

If URLs with /bin in them do not work, see Manual Step for Enabling URLs with /bin

5.2 Additional Steps: Microsoft IIS 7 or Later

The Microsoft ISAPI extensions (CSPms.dll, CSPmsSys.dll and CSPcms.dll) have been adapted such that they can work directly to the Native Modules interface in IIS 7 and later.

The Web Gateway modules supplied with InterSystems IRIS can work with the Native Modules in IIS 7. They can also be used with ISAPI extensions. There are additional configuration options for customers who are using the NSD. This section describes how to configure your IIS 7 web server to work with the Native Modules. For other configurations, see Alternative Options for IIS 7 or Later.

5.2.1 Install Locations for IIS 7

Install the Web Gateway components and the CSP static files as follows:

1. The default location for the Native Modules
   - CSPms.dll (Runtime module)
   - CSPmsSys.dll (Systems Management module)

The default location for these modules is:

C:\Inetpub\CSPGateway

The configuration and log files are written in this directory for non NSD-based connectivity options.

2. HyperEvents Components

   - CSPBroker.js
   - CSPxmlhttp.js

   The default location for these files is:

   install-dir\csp\broker

3. Miscellaneous static resources used by the Management Portal

   A number of static web resources (such as image files) are required by the Management Portal. The default location for these files is:

   install-dir\csp\sys

5.2.2 Recommended Option: Using Native Modules (CSPms*.dll)

This is the recommended and most-used configuration option. It uses the Native Modules interface introduced with IIS 7. This option provides the best performance.

For other configuration options, see Alternative Options for IIS 7 or Later (Windows) and Using the NSD (Windows).

Register the Native Modules and configure the web server so that it recognizes InterSystems file types and passes them to the Web Gateway for processing. Include any additional files that might be required for your installation (such as, for example, special CSP resources needed for analytics).

5.2.2.1 Registering the Native Modules

DLLs: CSPms.dll and CSPmsSys.dll

Before these modules can be used they must be registered with IIS. This is done in the **Internet Information Services (IIS) Manager** control panel.

1. Open the **Internet Information Services (IIS) Manager** window.
2. In the left panel, highlight: **[MACHINE_NAME] ([machine_name]\[user_name])**
3. In the middle panel, double-click the **Modules** icon.
4. In the right panel, select **Add Native Module** (or **Configure Native Modules**). The precise wording depends on the build of IIS in use.
5. Select **Register** and enter the following in the **Register Native Module** dialog:

   Name: **CSPms**

   Path: C:\Inetpub\CSPGateway\CSPms.dll

   Select **OK**.

6. In the left panel, expand the top level and expand **Web Sites**, and **Default Web Site**. Highlight **Default Web Site**:

```
[MACHINE_NAME] ([machine_name]\[user_name])
            Web Sites
                    Default Web Site
```

7. In the right panel, select **Add Native Module**.
8. Select **CSPms** and select **OK**.

5.2.2.2 Mapping InterSystems IRIS File Extensions

Note: Do not use **Add Wildcard Script Mapping** utility for this file extension mapping process; it gives an error. Instead, use the utility called **Add Module Mapping for ***.

Map the InterSystems file types to the Web Gateway native modules as follows:

Extension	Native Module	Binary
*.csp	CSPms	C:\Inetpub\CSPGateway\CSPms.dll
*.cls	CSPms	C:\Inetpub\CSPGateway\CSPms.dll
*.zen	CSPms	C:\Inetpub\CSPGateway\CSPms.dll
*.cxw	CSPms	C:\Inetpub\CSPGateway\CSPms.dll

These mappings do not provide support for REST; if you want to use REST, see Mapping Additional File Types.

1. Open the **Internet Information Services (IIS) Manager** window.
2. In the left panel, expand the top level and expand **Web Sites**, then the **Default Web Site** section. Highlight **Default Web Site**:

   ```
   [MACHINE_NAME] ([machine_name]\[user_name])
               Web Sites
                       Default Web Site
   ```

   Note: This activates CSP for the whole web site. To restrict the use of CSP to specific virtual sub-directories (such as /csp/) focus control on the appropriate subdirectory (under **Default Web Site**) before creating the mappings. Repeat the process for each virtual subdirectory from which CSP content is to be served.

3. In the middle panel, double-click the **Handler Mappings** icon.
4. In the right panel, select **Add Module Mapping**.
5. In the **Add Module Mappings** dialog, enter the following details:

   Request Path: *.csp

   Module: (select **CSPms** from the dropdown)

   Name: WebGateway_csp

6. Select **Request Restrictions** and ensure that the box is *not* checked next to **Invoke handler only if request is mapped to**. This action sets the value of **Path Type** to **Unspecified**, as shown in the **Handler Mappings** table.
7. Select **OK** to return to the **Add Module Mappings** dialog and select **OK** again.
8. Repeat the above process to add the following Module Mappings:

   Request Path: *.cls

   Module: (select CSPms from the list)

   Name: WebGateway_cls

   Request Path: *.zen

   Module: (select CSPms from the list)

Name: WebGateway_zen

Request Path: *.cxw

Module: (select CSPms from the list)

Name: WebGatewayManagement

5.2.2.3 Mapping Additional File Types

To configure the web server to send additional file types to the Web Gateway, add configuration entries similar to the default ones for the additional file extensions. For example:

Extension	Native Module	Binary
*.xxx	CSPms	C:\Inetpub\CSPGateway\CSPms.dll

iIf you need to serve other static files through the Web Gateway, add mappings for the needed file types so that those files can be loaded. In particular, if you need to access the Management Portal through this web server, add mappings for file types .jpg, .gif, .png, .css, and .js. If you do not do so, the Management Portal will be missing its images, its JavaScript, and its stylesheets.

To configure the web server to send all requests for a given path, set up the following wildcard entry for that path:

Extension	Native Module	Binary
*	CSPms	C:\Inetpub\CSPGateway\CSPms.dll

Important: If you want to use REST, you must use this mapping.

5.2.2.4 Operating and Managing the Web Gateway

To access the Web Gateway’s systems management suite, point your browser at the following location:

http://<ip_address>/csp/bin/Systems/Module.cxw

If you see an unauthorized user error message, see the security notes in Web Gateway and Security.

5.2.3 Configuring IIS to Return SOAP Fault Details

An InterSystems IRIS web service that encounters an error may return an `HTTP 500` error without the associated SOAP fault details. By default, IIS returns extended error information only to local clients. However, you can modify this behavior in the `<httpErrors>` element within the configuration file web.config. To do so, add the following section to instruct IIS to dispatch detailed error information to all clients.

```
<configuration>
    <system.webServer>
       <httpErrors errorMode="Detailed" />
    </system.webServer>
</configuration>
```

Use caution with this approach as sensitive information about the hosting environment may be revealed to clients. An alternative approach that avoids the security concerns of using `errorMode="Detailed”` is to instead use the `existingResponse="PassThrough"` directive.

```
<configuration>
    <system.webServer>
       <httpErrors existingResponse="PassThrough" />
    </system.webServer>
</configuration>
```

Restart IIS after making changes to the configuration.

You can make these changes manually to the IIS web.config file. Or, for a better, less error prone, approach, use the **Configuration Editor** built into the **IIS Manager**.

1. In the IIS Manager, from the **Connections** panel on the left, select the path which corresponds to the web service. For example: **Default Web Site**, then csp.
2. In the middle panel, below the **Management** heading at the bottom, double-click on **Configuration Editor**.
3. In the **Configuration Editor** dropdown at the top labeled Section, expand system.webServer and click httpErrors.
4. Click on the value next to `existingResponse` and use the dropdown to view the options. Select `PassThrough`.
5. In the **Actions** pane on the right, click **Apply**.
6. Restart IIS after making changes to the configuration.

Further information about error handling in IIS can be found at:

https://docs.microsoft.com/en-us/iis/configuration/system.webServer/httpErrors/

6

Configuring Apache to Work With the Web Gateway (Windows)

This page describes how to configure an Apache web server for use with the InterSystems Web Gateway on Windows. (On Windows, the other options are IIS and Nginx; options are different on other operating systems.)

If you installed the Web Gateway through the InterSystems IRIS® installer or if you want to configure an Apache server to work with CSP, follow the instructions in this page.

If you are using an Apache web server, follow these directions to configure Apache using Native Modules. Native Modules are extensions implemented as dynamically-linked modules (DLLs). They enable you to utilize the Internet Server Application Programming Interface (*ISAPI*) extensions. ISAPI is a high-performance API developed for Microsoft's web servers.

Apache is supplied by the Apache Group and can be downloaded free of charge from: http://www.apache.org.

The complete source code to Apache is available from Apache for download together with clear instructions for building the server. To build Apache under Windows, you must have the Microsoft C compiler (Visual C++) version 5.0 or later. Instead of building the server yourself, you can instead download prebuilt kits for Windows. The prebuilt kits are, generally, a few builds behind the latest Apache source code.

First follow the directions in Install Locations with Apache Servers (All Configurations), then follow the directions in Recommended Option: Apache API Modules (CSPa24.dll) (or if you are installing an atypical configuration, see Alternative Options for Windows Apache).

6.1 Assumptions

This page assumes that:

- The Web Gateway components are installed in `C:\Program Files\Apache Group\Apache\WebGateway`
- The web server is installed in `C:\Program Files\Apache Group\Apache\`

If the layout is different on your system, modify the configuration directives as appropriate.

6.2 Install Locations with Apache Servers (All Configurations)

All users of the Apache server should follow the directions in this section. Install the Web Gateway components and the CSP static files as follows:

1. CGI and other dynamically-linked modules:

   The common files for all Apache versions are:

   - CSPcgi.exe (Runtime module)
   - nph-CSPcgi.exe (Copy of CSPcgi.exe)
   - CSPcgiSys.exe (Systems-Management module)
   - nph-CSPcgiSys.exe (Copy of CSPcgiSys.exe)

   Separate binaries for Apache Version 2.4.x are:

   - mod_csp24.dll (Apache built-in module as a DLL, if supplied)
   - CSPa24.dll (Runtime module, if supplied)
   - CSPa24Sys.dll (Gateway Systems Management module, if supplied)

   The default location for these binaries is C:\Program Files\Apache Group\Apache\WebGateway\bin

   The original location (*install-dir*\csp\bin) is used to hold the Web Gateway components required for serving the Management Portal for the specific instance of InterSystems IRIS.

   The configuration and log files are written in this directory for non NSD-based connectivity options.

   The modules with Sys appended are special modules for accessing the Web Gateway management pages. The runtime modules (that is, those without Sys) have no access to the systems management forms.

2. HyperEvents Components

   - CSPBroker.js
   - CSPxmlhttp.js

   The default location for these files is *install-dir*\csp\broker

3. Miscellaneous static resources used by the CSP samples

   A number of static web resources (such as image files) are required by the CSP samples. The default location for these files is *install-dir*\csp\samples

4. Miscellaneous static resources used by the Management Portal

   A number of static web resources (such as image files) are required by the Management Portal. The default location for these files is *install-dir*\csp\sys

6.3 Recommended Option: Apache API Modules (CSPa24.dll)

This is the option that is used by the Private Web Server that serves the Management Portal.

This connectivity option is relatively new and offers the best performance as well as being the easiest to configure. Apache under Windows is entirely multi-threaded and its modules persist in memory from the time Apache is started. These two essential characteristics make it possible to implement the Web Gateway’s functionality as a set of stand-alone modules.

If you are installing an atypical configuration, see Alternative Options for Apache (Windows).

The modules CSPap*.dll (Runtime) and CSPapSys*.dll (Web Gateway systems management) are dynamically-linked modules that are designed to work the same way as the corresponding Microsoft ISAPI DLLs. For Apache 2.4.x, these modules are named: CSPa24.dll and CSPa24Sys.dll.

Configure the web server so that it recognizes InterSystems file types and passes them to the Web Gateway module for processing.

The web server configuration file (httpd.conf) is in C:\Program Files\Apache Group\Apache\conf

1. Apache 2.4.x: Add the following section to the end of httpd.conf.

```
LoadModule csp_module_sa c:/iris/csp/bin/CSPa24.dll
CSPFileTypes csp cls zen cxw
Alias /csp/ c:/iris/csp/
<Directory "c:/iris/csp">
    AllowOverride None
    Options MultiViews FollowSymLinks ExecCGI
    Require all granted
    <FilesMatch "\.(log|ini|pid|exe)$">
    Require all denied
    </FilesMatch>
</Directory>
```

2. Restart Apache after making changes to httpd.conf.
3. If needed, see Configuring Apache to Pass Additional File Types.

Now you can use the Web Gateway management pages to further configure the Web Gateway.

7

Configuring Apache to Pass Additional File Types (All Platforms)

On all platforms, Apache API modules always recognize the requests for the following InterSystems file types and send them to the Web Gateway:

```
.csp .cls .zen .cxw
```

You may have requests for other files that you want to send to the Web Gateway for routing to the appropriate InterSystems IRIS instance’s CSP server. For example, if you need to access the Management Portal through this web server, you must also add mappings for the following static file types: .jpg, .gif, .png, .svg, .css, and .js.

You can configure Apache to pass requests for additional files to the Web Gateway in any of the following ways:

- By CSP location directive
- By file extension
- By MIME type

Note: The CSP directives described on this page cannot be invoked as part of <VirtualHost> configurations.

7.1 Configuring Apache by Location

Use the `CSP` directive within a `<Location>` block to pass all requests for resources under a certain directory to the Web Gateway. The following causes Apache to forward all requests for resources under the /csp directory to the Web Gateway:

```
<Location /csp>
   CSP On
</Location>
```

For example, requests made for all of the following resources would be sent to the Web Gateway for routing to the appropriate InterSystem IRIS instance:

```
/csp/
/csp/samples/menu.csp
/csp/sys/
```

7.2 Configuring Apache by File Extension (CSPFileTypes Directive)

The `CSPFileTypes` directive configures the web server to route requests for files to the Web Gateway if they are files of a particular type (or types). This directive only works for requests for files that have extensions (such as /csp/menu.csp). It does not work for resources that do not have file extensions (such as /csp/menu).

The following directive causes Apache to pass requests for files of types xxx and yyy to the Web Gateway.

```
CSPFileTypes xxx yyy
```

You can issue this directive globally for the entire server or restrict it to a `<Location>` or `<Directory>` block. For example, the following causes Apache to pass requests for files of type xxx and yyy to the Web Gateway, but *only* when requests are sent to locations under the /csp directory (including subdirectories such as /csp/samples):

```
<Location /csp/>
   CSPFileTypes xxx yyy
</Location>
```

Using the wildcard character (`*`) with the `CSPFileTypes` directive as follows causes Apache to pass requests for files of all type under the /csp directory (and /csp/samples, and so on) to the Web Gateway.

```
<Location /csp/>
   CSPFileTypes *
</Location>
```

7.3 Configuring Apache by MIME Type

In addition to recognizing the file extensions listed above, the CSP engine can also recognize files for the following MIME types:

```
application/x-csp
```

and

```
text/csp
```

For example, to add the file extension xxx to the list of files processed by InterSystems IRIS (specifically by the CSP engine), use:

```
LoadModule csp_module_sa /iris/csp/bin/CSPa24.dll
AddType application/x-csp csp cls zen xxx
```

One of the problems with using MIME types to associate types of file with CSP is that Apache checks to ensure that the path to the resource (that is, the hosting directory) physically exists, and returns a `file not found` error if it does not. It does not, however, check to ensure that the file requested physically exists – which is appropriate for resources served by the CSP engine, which can be virtual as far as the web server is concerned. The “By MIME type” approach is therefore only suitable for cases where the application’s path structure can be replicated on the web server.

8

Building and Configuring Nginx to Work With the Web Gateway (Windows)

This page describes how to build and configure an Nginx web server for use with the InterSystems Web Gateway on Windows. (On Windows, the other options are Apache and IIS; options are different on other operating systems.)

Nginx is an Open Source product. The source code can be downloaded free of charge from: http://nginx.org/

Some prebuilt kits are available for Windows which are, generally, a few builds behind the latest Nginx build. However, given that extensions must be compiled into the Nginx core, it is necessary to build the web server locally from the source code in order to include support for CSP.

After the steps in this page, you can use the Web Gateway management pages to further configure the Web Gateway.

8.1 Assumptions

This page assumes that

- The CSP/Web Gateway components are installed in *install-dir*`\csp\`
- The web server is installed in `C:\nginx\`

If the layout is different on your system, modify the configuration directives as appropriate.

8.2 Installation

The Web Gateway components and the CSP static files should be installed as follows:

1. Web Gateway Network Service Daemon (NSD)
   - `CSPnsd.exe` (main binary)
   - `CSPnsdSv.exe` (Windows service)

   The default location for these files is *install-dir*`\bin`

   The configuration and log files are written in this directory.

2. HyperEvents Components

   - CSPBroker.js
   - CSPxmlhttp.js

   The default location for these files is *install-dir*\csp\broker

   If these files are to be served as static components directly by the web server, copy them to C:\nginx\html\csp\broker

3. Miscellaneous static resources used by the Management Portal

   A number of static web resources (such as image files) are required by the Management Portal. The default location for these files is *install-dir*\csp\sys

   If these files are to be served directly by the web server, copy these to C:\nginx\html\csp\sys

8.3 Building the Nginx Web Server for CSP

Most of the Web Gateway functionality is provided by the NSD (CSPnsd[Sv].exe). For CSP access, Nginx can be built and configured to communicate with the NSD through a small compiled-in module: ngx_http_csp_module.c. For convenience, all Web Gateway installations include this source file.

Prerequisites for building Nginx:

- Microsoft Visual Studio (version 10 or higher): http://www.microsoft.com
- MSYS2 (from MinGW): https://www.msys2.org/
- Perl (preferably ActivePerl): https://www.activestate.com/products/perl/
- Mercurial source control client: https://www.mercurial-scm.org/

The build instructions given here are based on the official documentation for building Nginx under Windows:

http://nginx.org/en/docs/howto_build_on_win32.html

The Nginx documentation stipulates that the following third party add-ons are also required:

- PCRE: http://www.pcre.org/
- OpenSSL (for SSL/TLS) https://www.openssl.org/
- Zlib: http://zlib.net/

However, it is possible to create a fully functional server without these components, provided the final installation doesn't require the functionality that would otherwise be provided by them.

The default configuration script for building Nginx, including all optional modules listed above, is as follows:

```
auto/configure \
 --with-cc=cl \
 --with-debug \
 --prefix= \
 --conf-path=conf/nginx.conf \
 --pid-path=logs/nginx.pid \
 --http-log-path=logs/access.log \
 --error-log-path=logs/error.log \
 --sbin-path=nginx.exe \
 --http-client-body-temp-path=temp/client_body_temp \
 --http-proxy-temp-path=temp/proxy_temp \
 --http-fastcgi-temp-path=temp/fastcgi_temp \
 --http-scgi-temp-path=temp/scgi_temp \
 --http-uwsgi-temp-path=temp/uwsgi_temp \
 --with-cc-opt=-DFD_SETSIZE=1024 \
 --with-pcre=objs/lib/pcre-8.44 \
 --with-zlib=objs/lib/zlib-1.2.12 \
 --with-openssl=objs/lib/openssl-1.1.1k \
 --with-openssl-opt=no-asm \
 --with-http_ssl_module
```

The build process can be modified to exclude optional modules:

- OpenSSL - Remove SSL/TLS capability:

  Remove directive: `--with-http_ssl_module`

- Zlib - Remove GZIP capability:

  Add directive: `--without-http_gzip_module`

- PCRE - Remove HTTP rewrite capability:

  Add directive: `--without-http_rewrite_module`

8.3.1 Procedure for Building Nginx for CSP

1. Working in a MSYS2 shell, create the working directory structure suggested in the Nginx documentation:

   /opt/

2. Working in /opt, check-out the Nginx source code using the following command:

   ```
   hg clone http://hg.nginx.org/nginx
   ```

   This places the Nginx source code under: /opt/nginx/

3. Create a directory for the CSP extension:

   ```
   mkdir /opt/nginx/objs/lib/csp/
   ```

4. Copy the module source code (ngx_http_csp_module.c) to the directory created in the previous step.

5. In the same directory, create a configuration file called config. This file should contain the following lines:

   ```
   ngx_addon_name=ngx_http_csp_module
   HTTP_MODULES="$HTTP_MODULES ngx_http_csp_module"
   NGX_ADDON_SRCS="$NGX_ADDON_SRCS $ngx_addon_dir/ngx_http_csp_module.c"
   ```

6. Working in /opt/nginx/, configure the Nginx build environment:

   ```
   auto/configure --with-cc=cl --builddir=objs --prefix=
                  --conf-path=conf/nginx.conf --pid-path=logs/nginx.pid
                  --http-log-path=logs/access.log --error-log-path=logs/error.log
                  --sbin-path=nginx.exe
                  --http-client-body-temp-path=temp/client_body_temp
                  --http-proxy-temp-path=temp/proxy_temp
                  --http-fastcgi-temp-path=temp/fastcgi_temp
                  --with-cc-opt=-DFD_SETSIZE=1024 --without-http_rewrite_module
                  --without-http_gzip_module
                  --with-select_module --with-ipv6
                  --add-module=objs/lib/csp
   ```

   Note the final line containing the instructions to include the CSP module.

7. Compile Nginx. This can be done in either the current MSYS2 shell or a Visual Studio developer command prompt.

To use the MSYS2 shell, locate the vcvarsall.bat script corresponding to your desired Visual Studio build environment and compile Nginx.

```
cd /c/path/to/vcvarsall
vcvarsall.bat
cd -
nmake -f objs/Makefile
```

Alternatively, if you do not know where to find vcvarsall.bat, you can open a Visual Studio developer command prompt, which will set up the build environment for you. First, convert the MSYS2 path to an equivalent Windows path in the current MSYS2 shell.

```
cygpath –m $(pwd)
```

Then, open a Visual Studio command prompt for your desired build environment and navigate to that Windows path. Compile Nginx.

```
nmake -f objs/Makefile
```

If successful, you should find the server (nginx.exe) in: /opt/nginx/objs/

8. Install Nginx: The easiest way to do this is to first download and install a pre-built version of Nginx for Windows to obtain the directory structure (usually under C:\nginx\) then replace the nginx.exe file in that installation with the one created locally.

   The typical directory structure for an operational Nginx installation is as follows:

```
Directory of C:\nginx

03/07/2017  09:09    <DIR>          .
03/07/2017  09:09    <DIR>          ..
26/06/2017  10:14    <DIR>          conf
26/06/2017  10:14    <DIR>          contrib
10/05/2018  12:53    <DIR>          csp
26/06/2017  10:14    <DIR>          docs
26/06/2017  10:14    <DIR>          html
10/05/2018  15:57    <DIR>          logs
04/07/2017  15:52           715,264 nginx.exe
26/06/2017  10:17    <DIR>          scgi_temp
26/06/2017  10:17    <DIR>          temp
26/06/2017  10:17    <DIR>          uwsgi_temp
```

   Replace the copy of *nginx.exe* in this directory with the version created by the build procedure.

8.4 Using the NSD with Nginx

You must configure the web server so that it recognizes requests for InterSystems file types and passes those requests (as well as requests for any other static files your InterSystems IRIS application serves) to the NSD for processing.

To do so, edit the web server configuration file (nginx.conf), which is found in C:\nginx\conf

This section describes the server configuration directives which the CSP extension module provides for configuring the web server. Issuing any of these directives within the context of a `location` block applies the directive to traffic at the path specified.

`CSPNSD_pass` *`hostname`*`:`*`portNum`*`;`

(Required.) Specifies the address (*hostname* and *port*) where the NSD is listening.

If you do not specify an NSD address for a particular path, the NSD listens at the address `127.0.0.1:7038` by default.

`CSP on;` and `CSP off;`

Enables or disables routing to CSP servers through the Web Gateway for all requests.

If you do not issue a CSP directive which applies to a particular path, no requests sent to that path are routed through the `CSP off`, and the web server does not route any requests sent to that path through the Web Gateway.

`CSPFileTypes` *`filetype1`*`[` *`filetype2`*`...];`

Enables the routing of requests for particular file types (*filetype1*, *filetype2*, and so on) to CSP servers through the Web Gateway.

For example, if you want the Web Gateway to route requests sent to the /demo/app path if (and only if) they are requesting .csp or .cls files, issue the following directive block:

```
location /demo/app {
  CSPFileTypes csp cls;
}
```

Issuing the directive `CSPFileTypes *` enables the routing of requests for all file types. It has the same effect as `CSP on.`

`CSPNSD_response_headers_maxsize` *`size`*`;`

Specifies the maximum size of headers for an HTTP response. If response headers exceed this size, the web client receives an error.

By default, the CSP extension module applies the directive `CSPNSD_response_headers_maxsize 8k`.

`CSPNSD_connect_timeout` *`time`*`;`

Specifies the timeout for connecting to the NSD when a request is received from the web client.

By default, the CSP extension module applies the directive `CSPNSD_connect_timeout 300s`.

`CSPNSD_send_timeout` *`time`*`;`

Specifies the timeout for a single send operation request (such as **POST** or **PUT**). The timeout is applied only between successive send operations; it does not apply to the completion of a single transmission once it has begun.

By default, the CSP extension module applies the directive `CSP_send_timeout 300s`.

`CSPNSD_read_timeout` *`time`*`;`

Specifies the timeout for a single read operation (such as **GET**) upon delivery of a response. The timeout is applied only between successive read operations; it does not apply to the completion of a transmission once it has begun.

By default, the CSP extension module applies the directive `CSP_read_timeout 300s`.

8.4.1 Example: Enable CSP Routing for All Traffic at a Particular Path

Place the following section within the appropriate **server** configuration block to route all traffic sent to the /csp path to the Web Gateway:

```
location /csp {
CSP On;
CSPNSD_pass localhost:7038;
}
```

8.4.2 Example: Route Requests for InterSystems IRIS File Types to the Web Gateway

Place the following section within the appropriate **server** configuration block to enable CSP routing for requests sent to the /csp path for the InterSystems IRIS file types (.csp, .cls, .zen, and .cxw):

```
location /csp {
CSPFileTypes  csp cls zen cxw;
CSPNSD_pass localhost:7038;
}
```

8.4.3 Start and Stop Nginx and the NSD

To start Nginx:

```
C:\nginx\nginx
```

To stop Nginx:

```
C:\nginx\nginx –s stop
```

See Operating the NSD for instructions on how to operate the NSD.

8.5 Deprecated: Building Nginx to Work with the Universal Modules

Important: Use of the Universal Modules with Nginx has been deprecated due to stability issues. Deployments of the Web Gateway which connect to Nginx using the NSD fully support all features, including WebSockets.

If you are currently using the Universal Modules with Nginx, InterSystems recommends upgrading to the most recent version of the Web Gateway and rebuilding your Nginx server to work with the NSD. Be sure to remove the `CSPModulePath` directive from your server configuration when you edit the server configuration file.

The following instructions serve as a reference for existing installations only.

Nginx can be built to work with the dynamically-linked Universal Modules CSPx.dll (runtime) and CSPxSys.dll (Web Gateway systems management) instead of the NSD. The procedure for building and configuring Nginx to work with the Universal Modules varies from the NSD-based deployment as follows:

- In step 4, copy the module source code ngx_http_csp_module_sa.c and ngx_http_csp_common.h to the specified directory, instead of ngx_http_csp_module.c.
- In step 5, the configuration file for CSP (/opt/nginx/objs/lib/csp/config) reads as follows:

  ```
  ngx_addon_name=ngx_http_csp_module_sa
  HTTP_MODULES="$HTTP_MODULES ngx_http_csp_module_sa"
  NGX_ADDON_SRCS="$NGX_ADDON_SRCS $ngx_addon_dir/ngx_http_csp_module_sa.c"
  ```

- Add the **CSPModulePath** directive to the **http** configuration block to specify the path to the Universal Gateway Modules.

  ```
  CSPModulePath install-dir/bin;
  ```

- For Windows, the thread stack size must be increased to 2MB. Add the following directive to the top of the Nginx configuration file (before the **http** section).

  ```
  thread_stack_size 2000000;
  ```

- The following directives are not supported:
  - **CSPNSD_pass**
  - **CSPNSD_response_headers_maxsize**
  - **CSPNSD_connect_timeout**
  - **CSPNSD_send_timeout**
  - **CSPNSD_read_timeout**
- The following directives are supported:
  - **CSP**
  - **CSPFileTypes**

9

Configuring Apache to Work With the Web Gateway (UNIX®/Linux/macOS)

This page describes how to configure an Apache web server for use with the InterSystems Web Gateway on UNIX®, Linux, or macOS. (On these operating systems, the other option is Nginx; options are different on other operating systems.)

Several connectivity options are available for Apache. Apache is supplied by the Apache Group and can be downloaded free of charge from http://www.apache.org. The Apache Group provides support for extensions implemented as dynamically linked modules (DSOs). Extensions, written as Apache modules, can be built directly into the Apache core. This is the recommended option; other options are the NSD, locked-down Apache web server, and several other atypical Apache configurations.

Pre-built kits are available for some UNIX® systems which are, generally, a few builds behind the latest version. The complete source code to Apache is available for download together with clear instructions for building the Apache server. The freely available GNU C compiler (gcc) can be obtained for this purpose, though the Apache build procedure attempts to use the indigenous C compiler.

Many systems are shipped with Apache preinstalled, configured and ready to go. Most distributions of Linux include Apache. IBM distributes Apache with their UNIX® implementation, AIX.

Note: In certain cases, the macOS and AIX preinstalled versions of Apache may not be suitable for use with InterSystems IRIS: see Using the NSD for UNIX®, Linux, and macOS for an alternative deployment option.

This section describes the recommended option for installing the Web Gateway.

1. For all configurations, follow the directions in Install Locations for Apache on UNIX®, Linux, and macOS.
2. Then follow the directions in Recommended Option: Apache API Module without NSD (CSPa24.so).

   Or for less common scenarios, see Locked-Down Apache Web Server, and other Atypical Apache Configurations.

9.1 Assumptions

This page assumes that:

- The Web Gateway components are installed in /opt/webgateway/bin/
- Apache is installed in /usr/apache/

If the layout is different on your system, modify the configuration directives as appropriate.

9.2 Install Locations for Apache for UNIX®, Linux, macOS (Recommended Option)

This section describes directory locations for Web Gateway files and CSP static files. The installation directory is /iris.

1. Dynamically-linked modules CSPa24.so for Apache Version 2.4.x.

   In order to avoid disrupting existing Web Gateway installations on upgrading InterSystems IRIS, the installation procedures place these modules in the following common location. This location is not related to a particular InterSystems IRIS instance.

   ```
   /opt/webgateway/bin
   ```

   The original location (/iris/csp/bin) is used to hold the Web Gateway components required for serving the Management Portal for the specific instance of InterSystems IRIS.

   The modules with Sys appended access the Web Gateway management pages. The runtime modules (that is, those without Sys) have no access to the Web Gateway management pages.

2. HyperEvents Components

   - CSPBroker.js
   - CSPxmlhttp.js

   The default location for these files is:

   ```
   /iris/csp/broker
   ```

3. Miscellaneous static resources used by the CSP samples

   A number of static web resources (such as image files) are required by the CSP samples. The default location for these files is:

   ```
   /iris/csp/samples
   ```

4. Miscellaneous static resources used by the Management Portal.

   A number of static web resources (such as image files) are required by the Management Portal. The default location for these files is:

   ```
   /iris/csp/sys
   ```

9.2.1 Requirements for using Apache API Modules (Recommended Option and Alternative Option 1)

Before following instructions for either Recommended Option and Alternative Option 1, check that your build of Apache includes the built-in module for managing shared objects (mod_so). To perform this check on Red Hat Linux, issue the following command:

```
httpd -l
```

To perform this check on Ubuntu or SUSE, issue the following command:

```
apache2 -l
```

These commands display the list of modules currently available within Apache. The shared object module (mod_so) should appear on this list. The following shows a typical module listing (with mod_so included):

```
Compiled in modules:
  core.c
  mod_access.c
  mod_auth.c
  mod_include.c
  mod_log_config.c
  mod_env.c
  mod_setenvif.c
  prefork.c
  http_core.c
  mod_mime.c
  mod_status.c
  mod_autoindex.c
  mod_asis.c
  mod_cgi.c
  mod_negotiation.c
  mod_dir.c
  mod_imap.c
  mod_actions.c
  mod_userdir.c
  mod_alias.c
  mod_so.c
```

If mod_so is not included in the list for your Apache installation, see your Apache documentation and follow the procedure for rebuilding Apache to include this module.

9.3 Recommended Option: Apache API Module without NSD (CSPa24.so)

This option is used in the configuration of the Private Web Server used by the Management Portal.

This connectivity option offers the best performance as well as being the easiest to configure.

Before using this option you should bear in mind that Apache v2.4 is partially multi-threaded, implemented as a hybrid multi-process and multi-threaded server. In practice, this means that there is one instance of the Web Gateway per Apache child process. This is not a problem in itself but this architecture makes it difficult to control the number of connections to InterSystems IRIS (and InterSystems IRIS processes) used by the Web Gateway because each instance of the Web Gateway independently manages its own pool of InterSystems IRIS connections.

State-aware connectivity (preserve mode 1) should not be used with these modules.

The modules CSPa24.so (Runtime) and CSPa24Sys.so (Web Gateway systems management) are dynamically linked modules (DSOs).

Configure the web server to recognize InterSystems file types and pass them to the Web Gateway module for processing. Apache 2.4.x: Use modules CSPa24.so and CSPa24Sys.so.

The web server configuration file (httpd.conf) is in the following directory:

```
/usr/apache/conf
```

For Red Hat Linux, the runtime version of httpd.conf is found in:

```
/etc/httpd/conf
```

For Ubuntu or SUSE, the runtime version of httpd.conf is found in:

```
/etc/apache2/conf
```

1. Apache 2.4.x: Add the following to the end of httpd.conf.

   ```
   LoadModule csp_module_sa /opt/webgateway/bin/CSPa24.so
   CSPModulePath /opt/webgateway/bin/
   CSPFileTypes csp cls zen cxw
   Alias /csp/ /opt/webgateway/bin/
   <Directory "/opt/webgateway/bin">
            AllowOverride None
            Options MultiViews FollowSymLinks ExecCGI
            Require all granted
            <FilesMatch "\.(log|ini|pid|exe)$">
            Require all denied
            </FilesMatch>
   </Directory>
   ```

2. Restart Apache after making changes to httpd.conf.
3. See Configuring Apache to Pass Additional File Types.

Now you can use the Web Gateway management pages to further configure the Web Gateway.

10

Building and Configuring Nginx (UNIX®/Linux/macOS)

This page describes how to build and configure an Nginx web server for use with the InterSystems Web Gateway on UNIX®, Linux, or macOS. (On these operating systems, the other option is Apache; options are different on Windows.)

Nginx is an Open Source product and the source code can be downloaded free of charge from: http://nginx.org/

Some prebuilt kits are available for Linux which are, generally, a few builds behind the latest Nginx build. However, given that extensions must be compiled into the Nginx core, it is necessary to build the web server locally from the source code in order to include support for CSP.

After the steps in this page, you can use the Web Gateway management pages to further configure the Web Gateway.

10.1 Assumptions

This page assumes that:

- CSP/Web Gateway web server components are installed in /opt/webgateway/bin/
- InterSystems IRIS, if installed locally, is in /opt/iris/
- The web server is installed under /opt/nginx/

If the layout is different on your system, modify the configuration directives as appropriate.

10.2 Installation

The Web Gateway components and the CSP static files should be installed as follows:

1. Web Gateway Network Service Daemon (NSD)
    - CSPnsd

    The default location for this binary is /opt/webgateway/bin/
2. HyperEvents Components

CSPBroker.js

CSPxmlhttp.js

The default location for these files is /opt/iris/csp/broker

If these files are to be served as static components directly by the web server, copy them to /opt/nginx/html/csp/broker

3. Miscellaneous static resources used by the Management Portal

   A number of static web resources (such as image files) are required by the Management Portal. The default location for these files is /opt/iris/csp/sys

   If these files are to be served as static components directly by the web server, copy them to /opt/nginx/html/csp/sys

10.3 Building the Nginx Web Server for CSP

Most of the Web Gateway functionality is provided by the NSD (CSPnsd). For CSP access, Nginx can be built and configured to communicate with the NSD through a small compiled-in module, ngx_http_csp_module.c. For convenience, all Web Gateway installations include this source file.

The build instructions given here are based on the official documentation for building Nginx under UNIX® systems:

http://nginx.org/en/docs/configure.html

The Nginx documentation stipulates that the following third party add-ons are also required:

- PCRE

  http://www.pcre.org/

- OpenSSL (for SSL/TLS)

  https://www.openssl.org/

- Zlib

  http://zlib.net/

However, it is possible to create a fully functional server without these components, provided the final installation doesn't require the functionality that would otherwise be provided by them.

A typical configuration script for building Nginx, including all optional modules listed above, is as follows:

```
./configure --prefix=/opt/nginx --with-http_ssl_module
```

This results in a default Nginx build installed under: /opt/nginx

The build process can be modified to exclude optional modules:

- OpenSSL - Remove SSL/TLS capability:

  Remove directive: `--with-http_ssl_module`

- Zlib - Remove GZIP capability:

  Add directive: `--without-http_gzip_module`

- PCRE - Remove HTTP rewrite capability:

  Add directive: `--without-http_rewrite_module`

10.3.1 Procedure for building Nginx for CSP

1. Unpack the source distribution under a location of your choice. For example:

   /opt/

   After unpacking, if you specify /opt/, the source code distribution is under:

   /opt/nginx-n.n.n/

2. Create a directory for the CSP extension:

   /opt/nginx-n.n.n/csp/

3. Copy the module source code (ngx_http_csp_module.c) to the directory created above.
4. In that same directory, create a configuration file called config. This file should contain the following lines:

   ```
   ngx_addon_name=ngx_http_csp_module
   HTTP_MODULES="$HTTP_MODULES ngx_http_csp_module"
   NGX_ADDON_SRCS="$NGX_ADDON_SRCS $ngx_addon_dir/ngx_http_csp_module.c"
   CORE_LIBS="$CORE_LIBS -ldl"
   ```

5. Working in /opt/nginx-n.n.n/, configure the Nginx build environment:

   ```
   ./configure --prefix=/opt/nginx
               --with-http_ssl_module
               --add-module=/opt/nginx-n.n.n/csp
   ```

   Alternatively, without the optional functionality provided by OpenSSL, ZLIB and PCRE:

   ```
    ./configure --prefix=/opt/nginx
                --without-http_rewrite_module
                --without-http_gzip_module
                --add-module=/opt/nginx-n.n.n/csp
   ```

   Note the final line containing the instructions to include the CSP module.

6. Compile Nginx:

   ```
   make
   ```

7. Install Nginx:

   ```
   make install
   ```

   If successful, you should find the complete server installation under:

   /opt/nginx/

10.4 Using the NSD with Nginx

You must configure the web server so that it recognizes requests for InterSystems file types (and any other requests your InterSystems application must serve) and then passes those requests to the NSD.

To do so, edit the web server configuration file (nginx.conf) which is found in /opt/nginx/conf

This section describes the server configuration directives which the CSP extension module provides for configuring the web server. Issuing any of these directives within the context of a `location` block applies the directive to traffic at the path specified.

`CSPNSD_pass` *`hostname`*`:`*`portNum`*`;`

(Required.) Specifies the address (*hostname* and *port*) where the NSD is listening.

If you do not specify an NSD address for a particular path, the NSD listens at the address `127.0.0.1:7038` by default.

`CSP on;` and `CSP off;`

Enables or disables routing to CSP servers through the Web Gateway for all requests.

If you do not issue a CSP directive which applies to a particular path, no requests sent to that path are routed through the `CSP off`, and the web server does not route any requests sent to that path through the Web Gateway.

`CSPFileTypes` *`filetype1`*`[` *`filetype2`*`...];`

Enables the routing of requests for particular file types (*filetype1*, *filetype2*, and so on) to CSP servers through the Web Gateway.

For example, if you want the Web Gateway to route requests sent to the /demo/app path if (and only if) they are requesting .csp or .cls files, issue the following directive block:

```
location /demo/app {
  CSPFileTypes csp cls;
}
```

Issuing the directive `CSPFileTypes *` enables the routing of requests for all file types. It has the same effect as `CSP on.`

`CSPNSD_response_headers_maxsize` *`size`*`;`

Specifies the maximum size of headers for an HTTP response. If response headers exceed this size, the web client receives an error.

By default, the CSP extension module applies the directive `CSPNSD_response_headers_maxsize 8k`.

`CSPNSD_connect_timeout` *`time`*`;`

Specifies the timeout for connecting to the NSD when a request is received from the web client.

By default, the CSP extension module applies the directive `CSPNSD_connect_timeout 300s`.

`CSPNSD_send_timeout` *`time`*`;`

Specifies the timeout for a single send operation request (such as **POST** or **PUT**). The timeout is applied only between successive send operations; it does not apply to the completion of a single transmission once it has begun.

By default, the CSP extension module applies the directive `CSP_send_timeout 300s`.

`CSPNSD_read_timeout` *`time`*`;`

Specifies the timeout for a single read operation (such as **GET**) upon delivery of a response. The timeout is applied only between successive read operations; it does not apply to the completion of a transmission once it has begun.

By default, the CSP extension module applies the directive `CSP_read_timeout 300s`.

10.4.1 Example: Enable CSP Routing for All Traffic at a Particular Path

Place the following section within the appropriate **server** configuration block to route all traffic sent to the /csp path to the Web Gateway:

```
location /csp {
CSP On;
CSPNSD_pass localhost:7038;
}
```

10.4.2 Example: Route Requests for InterSystems IRIS File Types to the Web Gateway

Place the following section within the appropriate **server** configuration block to enable CSP routing for requests sent to the /csp path for the InterSystems IRIS file types (.csp, .cls, .zen, and .cxw):

```
location /csp {
CSPFileTypes  csp cls zen cxw;
CSPNSD_pass localhost:7038;
}
```

10.4.3 Start and Stop Nginx and the NSD

To start Nginx:

```
/opt/nginx/sbin/nginx
```

To stop Nginx:

```
/opt/nginx/sbin/nginx –s stop
```

See Operating the NSD for instructions on how to operate the NSD.

10.5 Deprecated: Building Nginx to Work with the Universal Modules

Important: Use of the Universal Modules with Nginx has been deprecated due to stability issues. Deployments of the Web Gateway which connect to Nginx using the NSD fully support all features, including WebSockets.

If you are currently using the Universal Modules with Nginx, InterSystems recommends upgrading to the most recent version of the Web Gateway and rebuilding your Nginx server to work with the NSD. Be sure to remove the `CSPModulePath` directive from your server configuration when you edit the server configuration file.

The following instructions serve as a reference for existing installations only.

Nginx can be built to work with the dynamically-linked Universal Modules CSPx.so (runtime) and CSPxSys.so (Web Gateway systems management). The procedure for building and configuring Nginx to work with the Universal Modules varies from the NSD-based deployment as follows:

- In step 3, copy the module source code ngx_http_csp_module_sa.c, cspapi.h, and ngx_http_csp_common.h to the specified directory, instead of ngx_http_csp_module.c.

- In step 4, the configuration file for CSP (/opt/nginx-n.n.n/csp/config) should read as follows:

  ```
  ngx_addon_name=ngx_http_csp_module_sa
  HTTP_MODULES="$HTTP_MODULES ngx_http_csp_module_sa"
  NGX_ADDON_SRCS="$NGX_ADDON_SRCS $ngx_addon_dir/ngx_http_csp_module_sa.c"
  ```

- Add the **CSPModulePath** directive from the **http** configuration block to specify the path to the Universal Gateway Modules.

  ```
  CSPModulePath /opt/webgateway/bin;
  ```

- The following directives are not supported:
  - **CSPNSD_pass**
  - **CSPNSD_response_headers_maxsize**
  - **CSPNSD_connect_timeout**
  - **CSPNSD_send_timeout**
  - **CSPNSD_read_timeout**
- The following directives are supported:
  - **CSP**
  - **CSPFileTypes**

11

Configuration Basics for the Web Gateway

After configuring the web server to work with the Web Gateway, configure the Web Gateway as needed. You can use the management pages to configure default parameters, individual servers, and web applications.

This page provides additional information on some common configuration topics.

11.1 Web Gateway Configuration and Log Files

The Web Gateway creates and uses a configuration file and a log file, respectively named CSP.ini and CSP.log. For convenience, the management pages indicate the locations of these files.

Tip: It is rarely necessary to edit the configuration file directly, because you normally perform configuration via the management pages. Where necessary, the documentation describes the directives in the configuration. If you need to edit the configuration file directly, see the class reference for %CSP.Mgr.GatewayMgr. The methods **SetApplicationParams**, **SetDefaultParms**, and **SetServerParams** provide short descriptions of the parameters.

11.2 File Types Automatically Routed to InterSystems IRIS

InterSystems file types are processed in InterSystems IRIS (specifically by the CSP engine). All other files (static files) can be served by the web server or by the CSP engine. The CSP engine can serve any type of file that is placed in the path of the web application (including static files). Setting up the CSP engine to serve static files simplifies the web server configuration for web applications because you, thus, do not need to create aliases in the web server configuration to represent the locations where an application's static files are held. Setting up the CSP engine to serve static files resolves issues of contention when a single (that is, common) web server serves two different versions of InterSystems IRIS, each requiring different versions of certain static files (for example, hyperevent broker components).

To have the CSP engine serve static files for a particular web application, place the static files in the web application's file system in the correct location relative to the CSP files that make up the application (not in the web server's own documents file system).

Note: To run Zen-based applications, you must enable the **Serve Files** option and properly configure your web server.

11.3 Serving Static Files from InterSystems IRIS

You can configure web servers and Web Gateway installations so that InterSystems IRIS assumes responsibility for serving static files. The Management Portal is configured for InterSystems IRIS to serve all components in the application. However, it is still possible to configure the web server so that it retains responsibility for serving static files.

The InterSystems IRIS database server serves all of CSP. It can also serve any kind of static file for a web application via the Web Gateway. In standard web applications, web servers (not database servers) typically serve static content.

11.3.1 Indicating Character Encoding

The CSP engine, via the stream server, handles static files in a manner consistent with major web servers, with respect to determining the character encoding of JavaScript files. The modern convention is for all JavaScript files to marked as Content-Type of `application/javascript`, and you make sure this is the case with all JavaScript files used on your pages.

With JavaScript files marked this way:

- If a file contains a BOM (byte-order mark), the browser automatically detects this and uses the correct character set to read it.
- If the file does not contain a BOM, then the browser assumes the file is UTF-8.

If you need to override this behavior to specify a character set for JavaScript files, set the global *^%SYS("CSP","MimeFileClassify","JS")* to the list value **$listbuild(contenttype, binary, charset)**. For example,

```
SET ^%SYS("CSP", "MimeFileClassify", "JS") = $listbuild("text/javascript", 0 ,"ISO-8859-1")
```

This sets the older content-type and uses the ISO-8859-1 character set. Also, if the Caché translate table is defined to be something other than an empty string or if the global node *^%SYS("CSP","DefaultFileCharset")* is set to a null value, Caché will use this character set for all JavaScript and other text files. By default, neither of these global nodes are set.

11.3.2 Enabling the Serve Files Option

The **Serve Files** option for a web application has the following values:

- **Always** — **Serve Files** is on.
- **No — Serve Files** is off.
- **Always and cached** — The Web Gateway can cache static files on the web server. This setting improves efficiency as the system can serve a cached static page without going back to the InterSystems IRIS server.
- **Use InterSystems Security** — If the user has the appropriate authorization to access dynamic .csp or .cls pages which an InterSystems IRIS instance serves as part of a web application, then the CSP engine for that instance serves requests for static files at that web application location. If the user does not have appropriate authorization, then the CSP server returns an HTTP 404 response.

11.3.3 Configuring the Web Server to Allow InterSystems IRIS to Serve Static Files

By default, the installation script provided by InterSystems does not look for or configure an external web server. If you run a custom install, you can choose the option to configure any previously-installed IIS or Apache web server to enable support for the CSP engine. The installation script creates the /csp virtual directory and creates mappings for the InterSystems file types to be handled by the Web Gateway. This is sufficient for the CSP engine to function normally via that web server. It does *not* enable support for the **Serve Files** feature. You must manually configure your web server to map specific file extensions to be handled via the Web Gateway. This is by design because opening the database server up to exposure via this mechanism is a security risk and should not be done invisibly.

See Mapping Additional File Types for instructions for your web server.

11.3.4 Serving Static Files from the Web Server

You can use a traditional configuration of serving static pages from the web server. In this case, the setting of the **Static Files** option is irrelevant. This eliminates contention when a common web server serves two different versions of InterSystems IRIS, each requiring different versions of certain static files, for example, hyperevent broker components.

If you have configured the web server itself to serve static files, be sure that the static content is present on every single web server.

11.4 Enable Sticky Sessions on Hardware Load Balancer on High Availability Solutions

For High Availability solutions running over CSP, InterSystems recommends that you use a hardware load balancer for load balancing and failover. InterSystems requires that you enable sticky session support in the load balancer; this guarantees that — once a session has been established between a given instance of the Web Gateway and a given application server — all subsequent requests from that user run on the same pair. This configuration assures that the session ID and server-side session context are always in sync; otherwise, it is possible that a session is created on one server but the next request from that user runs on a different system where the session is not present, which results in runtime errors (especially with hyperevents, which require the session key to decrypt the request). See your load balancer documentation for directions on how to enable sticky session support.

Note: It is possible to configure a system to work without sticky sessions but this requires that the web session global be mapped across all systems in the enterprise and can result in significant lock contention so it is not recommended.

11.5 Enable Script to Reactivate Web Gateway Configuration

You can enable a script that is external to InterSystems IRIS to reactivate the Web Gateway’s configuration.

Scripts should add the following line (case-sensitive) to the SYSTEM section of the Web Gateway configuration file:

```
[SYSTEM]
RELOAD=1
```

The Web Gateway caretaker daemon checks the RELOAD flag approximately every minute and, if correctly set, reloads and reactivates its configuration and removes the flag from the file. The following message is written to the Web Gateway Event Log after a successful reload operation:

```
Gateway Management
Gateway Configuration Reloaded and Reactivated
```

11.6 Hybrid Multi-Process/Multi-Threaded Web Server Architecture

The Web Gateway contains enhanced support for the hybrid multi-process/multi-threaded web server architecture. Apache version 2.4 under UNIX® is an example of a web server implemented according to this architecture.

The core Web Gateway resources are held in the shared memory sector. All web server worker processes hare a common running configuration, connection table and form cache. The Web Gateway System Status form shows the status for the whole web server installation instead of just that of a single worker process. The status form’s connection table includes an extra column with the web server process ID with respect to each connection to InterSystems IRIS.

Finally, state-aware sessions are supported in the multi-process architecture. Although the connection pool (to InterSystems IRIS) is distributed amongst several web server processes, the Web Gateway uses an InterProcess Communications (IPC) protocol to route requests for state-aware sessions to the correct hosting process in the web server environment.

12

Overview of the Web Gateway Management Pages

The Web Gateway provides a set of management pages that you can use to configure and monitor the Web Gateway. This page describes how to access these pages and how to localize them, and provides an overview of the options in them.

12.1 Accessing the Web Gateway Management Pages

By default, only clients local to the Web Gateway's hosting computer can access the Web Gateway management pages, so that the browser you use to access these pages must be running on the same machine as the web server and the Web Gateway. For example:

http://localhost:<*port_no*>/csp/bin/Systems/Module.cxw

Be sure to include the cxw file extension. (Also, if you are using Apache, remember that URL paths and files names are case-sensitive under Apache.)

For more information on default InterSystems IRIS web service server port numbers, see the WebServerPort entry of the *Configuration Parameter File Reference*.

When you try to access these pages, you are asked for a username and password. Look for the username in the configuration file. The password is the one that you entered during the InterSystems IRIS installation. If you forget the password, see Security.

Tip: Note that you can also access these pages from the InterSystems Management Portal. Navigate to **System Administration** > **Configuration** > **Web Gateway Management**. The same considerations apply with respect to client addresses.

12.2 Enabling Access from Additional Client Addresses

You can add additional clients to the list of authorized administrators by adding the client IP addresses to the `System_Manager` parameter in the SYSTEM section in the configuration file. This parameter represents a comma- or plus-separated list of clients (by IP address) who may access the Web Gateway management pages. The directive shown below grants access to three remote clients in addition to the default local access.

```
[SYSTEM]
System_Manager=190.8.7.6, 190.8.7.5, 190.8.7.4
```

For new Gateway installations, for which there is no local browser available, manually edit the configuration file and add the *System_Manager* parameter, which is equivalent to the **Systems Manager Machines** setting, found under the **Default Parameters** section of the Web Gateway management pages. You can specify wildcard and numeric ranges in the entries for this parameter.

Note: If you attempt to load the Web Gateway management pages, and the browser fails to load the page, giving an error **You are not authorized to use this facility**, this is likely due to the *System_Manager* setting blocking access to your IP address.

The following example indicates that the last part of the IP address can take the value of a number between 4 and 6 inclusive.

```
[SYSTEM]
System_Manager=190.8.7.4-6
```

The previous example is a more convenient way of writing:

```
[SYSTEM]
System_Manager=190.8.7.6, 190.8.7.5, 190.8.7.4
```

You can also use wildcards, such as, in this example:

```
[SYSTEM]
System_Manager=190.8.7.*
```

The following directive grants access to all clients:

```
[SYSTEM]
System_Manager=*.*.*.*
```

However, it is not recommended to use such a directive on operational systems; this approach does not provide strong security, because client IP addresses can be spoofed.

The use of a proxy between the client and the web server/Gateway installation effectively translates all client IP addresses to that of the proxy. In this scenario, you would have to either specify the proxy's IP address as a Gateway Systems Manager (which would effectively grant access to all web users coming in through the proxy) or, preferably, enable the designated systems managers to bypass the proxy layer altogether.

The IP-based scheme, while useful as a first line of defense, should not be relied upon as the sole means through which access to the Web Gateway management pages is controlled – certainly not for CSP installations that are available over the internet. For production systems, it is recommended that you use the hosting web server configuration to control access to the Web Gateway systems management modules.

12.3 Available Options

The following table shows the options available on the Web Gateway Management Main Menu page.

Menu Item	Action
About Web Gateway	Shows information about the Web Gateway, including the version of the InterSystems IRIS distribution, the Web Gateway build number, the version of OpenSSL that is loaded, the version of the hosting web server, the active interface, the name and location of the Web Gateway configuration file (CSP.ini), and the name and location of the event log (CSP.log). The Web Gateway build number is made up of two numeric components. The first number indicates the version of InterSystems IRIS. The second number is the internal Web Gateway build number.

Menu Item	Action
System Status	Displays the status of active server connections. Also allows you to close connections and clear the Web Gateway cache.
Test Server Connection	Tests the connection to an InterSystems IRIS server by opening a stateless session.
View Event Log	Allows you to view information in the Web Gateway Event Log, as well as clear its contents. This log is a file maintained on the web server host.
View HTTP Trace	Provides an interactive view of the HTTP requests and responses processed by the Web Gateway.
Default Parameters	Allows you to configure the Web Gateway on a specific web server. Also, it allows you to customize CSP responses to errors and other conditions.
Server Access	Configures Web Gateway access to a specific InterSystems IRIS server.
Application Access	Configures the access to an application according to the application path. Path, in this context, refers to the path contained within the application URLs.
Back to Management Portal	Returns to the InterSystems IRIS Management Portal page.

These pages include a **Help** button.

12.4 Localization

Localization of the Web Gateway management pages is based on the contents of the file CSPres.xml, if it is installed. If no localization file is present, the Web Gateway management pages use the embedded English text. The language settings of the browser have no influence on this mechanism.

You can support alternative languages by installing the appropriate text resource file as a file named CSPres.xml in the Web Gateway's home directory. When the Web Gateway starts or restarts, it loads the text resources found in CSPres.xml and the Management forms then appear in the chosen language.

To create a CSPres.xml file, rename the appropriate CSPres_xx.xml file in the InterSystems IRIS bin directory to CSPres.xml.

For example, to convert to Spanish:

1. Rename CSPres_es.xml to CSPres.xml
2. Restart the web server. You must restart because the language text affects the CSP module for the given web server.

To convert back to English:

1. Rename CSPres.xml back to CSPres_es.xml
2. Restart the web server.

13

Configuring the Default Parameters for Web Gateway

This page describes how to configure the default parameters for InterSystems IRIS® Web Gateway via the Web Gateway management pages. Other articles describe how to configure servers and applications.

The **Default Parameters** option on the Web Gateway management pages controls all the global (system-wide) configuration parameters for the Web Gateway. Note that you must be a system manager to use this option.

When you configure access to a particular InterSystems IRIS Server, any unspecified optional parameters and/or custom system forms are automatically inherited from the global configuration. For example, if you do not set a **Server Response Timeout** parameter for a specific server, that server inherits the global **Server Response Timeout** setting.

13.1 Web Gateway

This section contains parameters that are globally relevant to the whole Web Gateway installation.

Instance Host Name

This is the network host name for this particular instance of the Web Gateway. The Web Gateway generates a default value which is shown beneath the text box. The value of this parameter is transmitted to InterSystems IRIS with the request data as system variable CSPIHN. Your application can use the value to access management services provided by the Web Gateway over the network.

The format for this parameter is: `server_name:port`

Maximum Connections

Maximum number of connections to InterSystems IRIS that can be created from this Gateway instance. The default value is set to 1024. Increasing this value allows better application responsiveness if an application uses more connections but may also result in heavier server resource utilization.

Changes to the Maximum Connections parameter only take effect after a restart of the Web Gateway (or the hosting web server).

Maximum Cache Size

Maximum amount of shared memory to be reserved for the purpose of caching CSP response data.

The cache size may be specified as a number with no suffix for bytes, a number followed by **K** for kilobytes, or a number followed by **M** for megabytes.

The default value for this parameter is 256K. This value can be raised or lowered as required.

Changes to the Maximum Cache Size parameter only take effect after a Gateway (or hosting web server) restart.

Web Server ID Cookie

Suppresses the Web Server ID Cookie (*CSPWSERVERID*). It can be set to:

- Enabled (Default)
- Disabled

The Web Server ID Cookie is used to enable load balancers to implement passive cookie affinity for web applications. However, there are situations where it is desirable to suppress the automatic generation of this cookie. For example, in proxy applications where the web request is transparently passed to other servers for processing.

The Web Server ID Cookie is not dispatched when returning resources that are deemed to be static (i.e. images and JS files). In this context, static files include all responses generated by InterSystems IRIS that are not accompanied by a Web Server ID Cookie. An exception to this rule is made for cases where the application is configured to `Never` use session cookies. In this case the Web Server ID Cookie is included with all responses (as before).

13.2 Security

If a username and password are defined here, then all system managers must provide this username and password to access the Web Gateway management pages.

If you forget the password, use the following procedure to set a new password:

1. Edit the configuration file to specify a new value for the **Username** and **Password**; make the changes in the `SYSTEM` section of the file. You can specify the value of the password in plaintext.
2. Restart the web server. The web server reloads the configuration and updates the file to hold the password hash instead of the plaintext password.

You can now log in to the web server with the updated username and password.

```
[SYSTEM]
Username=cm
Password=1Bx4tt88mttAWaf7isJg3Urqc2zE
```

You can configure the following CSP security parameters:

Access to these forms

Enable or disable access to the Web Gateway management pages option. The default is **Enabled**. When access is **Disabled** you cannot re-enable access using the Web Gateway management pages. To re-enable access, manually edit the configuration file. Set the `SM_Forms` parameter to `Enabled` in the `SYSTEM` section of this file.

```
[SYSTEM]
SM_Forms=Enabled
```

Username

Username required to access the Web Gateway management pages.

Password

Password required to access the Web Gateway management pages.

Password (Confirm)

When the password is modified, confirm the new value here.

Session Timeout

The amount of idle time (in seconds) that an active Systems Management session remains logged on. After this time has expired, the management session expires and the manager is automatically logged out of the Web Gateway management pages.

System Manager Machine/s

Defines a list of client machines (by IP address) through which you can access these Systems Management options. Any client with System Manager access can add or delete access to any CSP system, change any setting in the configuration file, and close down any active sessions. The addresses are separated by either a comma or a plus sign. In this example, two clients have System Manager access:

`127.0.0.1, 45.123.231.12`

If this field is undefined, only a client operating on the same machine as the Web Gateway (that is, the web server host) can configure CSP.

This field is supplemented with a check box (**Override Username and Password**) which, if checked, allows listed client machines to be exempt from entering a username and password to gain access to the Management Forms.

Custom Login Form

Defines a custom login form that controls access to the Web Gateway management pages. This parameter can be either the full path to a physical file or a link which enables the hosting web server to serve the form.

Examples:

```
C:\Inetpub\wwwroot\login.html
/login.html
```

If a physical file name is specified then the Web Gateway retrieves and dispatches the form to the client. Otherwise, it sends an 'HTTP Redirect' response header to enable the client to request the form directly from the hosting web server. The custom form must implement an HTTP POST request to login Gateway Administrators.

The essential form fields are shown below:

```
<FORM METHOD=POST ACTION="/csp/bin/Systems/Module.cxw">
<INPUT TYPE=HIDDEN NAME="CSPSYS" VALUE="17">
<INPUT TYPE=HIDDEN NAME="CSPSYSsmSection" VALUE="SYSTEM">
<INPUT TYPE=TEXT NAME="CSPUNM" SIZE='20' VALUE="">
<INPUT TYPE=PASSWORD NAME="CSPPWD" SIZE='20' VALUE="">
<INPUT TYPE=SUBMIT NAME="CSPSYSbOK" VALUE="Login">
```

Where CSPUNM is the username and CSPPWD the password. The text assigned to the Login (submit) button (shown as 'Login' above) can be changed.

A simple but complete example is shown below:

```
<html>
<head>
<title>Web Gateway Management</title>
</head>
<h2>Web Gateway Management</h2>
<FORM METHOD=POST ACTION="/csp/bin/Systems/Module.cxw">
<INPUT TYPE=HIDDEN NAME="CSPSYS" VALUE="17">
<INPUT TYPE=HIDDEN NAME="CSPSYSsmSection" VALUE="SYSTEM">
<BR>
Username:
<INPUT TYPE=TEXT NAME="CSPUNM" SIZE='20' VALUE="">
<BR>
Password:
<INPUT TYPE=PASSWORD NAME="CSPPWD" SIZE='20' VALUE="">
<BR>
<INPUT TYPE=SUBMIT NAME="CSPSYSbOK" VALUE="Login">
</form>
</html>
```

13.3 Connections to InterSystems IRIS

This section contains parameters related to maintaining connections to InterSystems IRIS.

Server Response Timeout

The maximum number of seconds allowed for the target InterSystems IRIS server to respond to a request from the web server. The timeout refers to a period of no activity, so, for example, sending a line of HTML data every second for 10 hours does not cause a timeout. The minimum allowable value for this field is 5 seconds.

The value set here is the default for the system. If an Inherited Value is specified, the value came from the Default Parameters page. You may, however, set a different value on individual server-specific configurations or within the application itself.

Note that if you have an Apache server, you can also set this value using *Timeout* in the Apache httpd.conf file. The lower of these two values is triggered first.

Queued Request Timeout

This is the maximum number of seconds that a request can remain in a queue waiting for an available connection to the appropriate InterSystems IRIS system. The minimum allowable value is 5 seconds. If an Inherited Value is specified, the value came from the Default Parameters page.

No Activity Timeout

This parameter is relevant to stateless connections only. The parameter indicates the maximum amount of time (in seconds) that a stateless connection remains open in an idle state before closing. If this timeout is exceeded, the session automatically closes. This facility prevents stateless sessions accumulating on your InterSystems IRIS server, particularly after periods of high activity where a large number of connections were opened to cope with the increased workload. If a value is not specified, stateless connections remain open until they are manually closed. If an Inherited Value is specified, the value came from the Default Parameters page.

Note that a process may remain up to a few minutes beyond the configured timeout. By design, the Web Gateway only checks for connection timeouts periodically; it is not immediately notified when a timeout occurs. Depending on the timing of this check, a process can linger for up to 420 additional seconds.

Apply timeout to all Connections

Applies the **No Activity Timeout** option to all connections (including those making up the minimal connection pool). If this option is not checked, the Web Gateway does not apply the **No Activity Timeout** to the minimal connection pool (as defined by the **Minimum Server Connections** parameter). If this option is checked the Web Gateway applies the timeout to all connections in the pool. This option is used by installations that have a very low level of CSP usage and, as a result, have a preference for all CSP processes to time out. If an Inherited Value is specified, the value came from the Default Parameters page.

Event Log Level

Controls what information is written to the Web Gateway Event Log. See Event Logging Parameters for details.

Event Log File

Specifies a location and filename for the Web Gateway Event Log. If not specified, it is written to the directory hosting the Web Gateway installation. For example:

To specify an alternative location:

/opt/logfiles/cspgateway/

To specify an alternative location and file name:

/opt/logfiles/cspgateway/event_log_01012006.log

Retain All Log Files

If **Event Log Rotation Size** is blank (the default), the Web Gateway Event Log grows until the administrator manually clears it. If the capacity of the file is specified by **Event Log Rotation Size**, InterSystems IRIS copies the log file to a file named filename.old, where *filename* is the full original filename. Subsequent log rotations overwrite filename.old with the current contents of the log. To retain all log files, select **Retain All Log Files**. Each log is named with the date and time when the copy took place.

Event Log Rotation Size

This defines the size after which log rotation should take place. The default value is blank which means that the Web Gateway maintains one log file which grows until the administrator manually clears it.

If rotation is required, the size may be specified as a number with no suffix for bytes, a number followed by **K** for kilobytes, or a number followed by **M** for megabytes.

The minimum size that can be specified is 100K. This value is automatically set if the administrator attempts to set a lower value in the maintenance suite.

Rotated copies of the log file, if retained, are named according to the date and time of rotation as follows:

```
CSP_YYYYMMDD_hhmm.log
```

where `YYYY` is year, `MM` is month, `DD` is date, `hh` is hour, and `mm` is minutes past the hour, for example:

```
CSP_20090109_1830.log (Log rotated at 18:30 on 9th January 2009)
```

If more than one log file rotation takes place in the space of one minute, a serial number is appended to the file name to prevent duplicates, for example:

```
03/12/2015  17:02         106,660 CSP_20151203_1702.log
03/12/2015  17:02         124,752 CSP_20151203_1702.log.0001
03/12/2015  17:02         124,752 CSP_20151203_1702.log.0002
```

The rotated log file that is not to be retained is named: filename.old, where *filename* is the full original filename.

In order for this facility to work, the Web Gateway must have create/write access to the directory hosting the Web Gateway binaries (i.e. the location where the main log file is kept). If the Web Gateway is unable to perform a successful rotation it continues writing to the current log file.

This field is supplemented with a check box labeled **Retain All Log Files**. If selected, this option instructs the Web Gateway to keep all log files according to the naming scheme outlined above.

Maximum Logged Request Size

If you have enabled logging for HTTP requests by specifying an Event Log Level of V9 (or a variant such as V9b), this parameter specifies how much of the HTTP request is included in the log. Any request which exceeds this maximum is truncated.

The default value for this parameter is 256K, and the minimum value is 40K. If you leave the field empty, it is set to the default (256K). The minimum is enforced; if you attempt to assign a value lower than the minimum, it is set to 40K.

SSL/TLS Library Path

Specifies the path to the OpenSSL libraries. On UNIX®, these files are libssl.so and libcrypto.so, and on Windows, the files are libcrypto-1_1-x64.dll and libssl-1_1-x64.dll. The default is for the Web Gateway to source these libraries locally in its home directory. See Overriding the Library Path If You Use SSL/TLS in Kerberos Library for more information.

Preserve Mode Exclude File Types

Allows static files to be served asynchronously in state-aware applications. In stateless applications, statics (files other than csp, cls, csr, and zen) are processed asynchronously with respect to the main session. In other words, requests for these files bypass the session lock and can be processed concurrently outside the main processing stream for the application.

This parameter allows this scheme to be extended to state-aware applications. State-aware applications are not only subject to the conventional session lock but are also subject to the connection lock in the Web Gateway. The connection lock is responsible for ensuring that all requests for the user/session are routed to the same InterSystems IRIS process. For applications that rely on static components being served from InterSystems IRIS, this leads to excessive request queuing which, in turn, can lead to browser instability (such as hangs).

Use this parameter to define a list of (space separated) file types (by extension) to process asynchronously and therefore exempt from connection/session locking in the Web Gateway and InterSystems IRIS. If the list is prefixed with `*-` (asterisk hyphen) then all files are processed asynchronously EXCEPT those defined in the following list.

Examples

```
Preserve Mode Exclude File Types=gif jpg jpeg
```

Process files of type GIF, JPG and JPEG asynchronously with respect to the state-aware session:

```
Preserve Mode Exclude File Types=*- csp cls csr zen
```

Process all files asynchronously with respect to the state-aware session EXCEPT those of type CSP, CLS, CSR and ZEN. This, incidentally, is the rule applied in the CSP engine for stateless applications.

This mechanism can be monitored using log level v4. When invoked for a request, a record similar to the one shown below is added to the log.

```
 >>> Time: Fri Oct 04 14:56:40 2017 ...GET /csp/samples/zenutils.js
     State-Aware Session (preserve == 1)
     Process this request concurrently in the pool of stateless connections (File Type=js)
```

13.4 ASP Redirect

Web Document Root

This is the full physical path to the document root directory of the web server. For example, for Microsoft IIS Web Servers, this path is usually c:\InetPub\wwwroot. This parameter is only required if you plan to use the facility within CSP to send the CSP output through the Microsoft ASP engine to render the final page.

Temp ASP Directory

This is the full physical path to a directory where the Web Gateway can temporarily store Microsoft ASP content. This parameter is only required if you plan to use the facility within CSP to send the CSP output through the Microsoft ASP engine to render the final page.

13.5 Internal HTTP Server

This section is only relevant to the NSD. This section contains the following parameters:

Service Status

The HTTP server can be Enabled or Disabled. Select either:

- Enabled
- Disabled

The default is `Enabled`.

In the interests of security, it is best to disable this facility, unless it is intended that the NSD should be able to respond to raw HTTP requests.

NSD Document Root

For cases where the NSD is intended to be used as a stand-alone web server in its own right, this parameter defines the full physical path to the web documents root. For example:

```
/opt/webgateway/home/
```

If the server is used to serve web applications, then the broker components should be installed under:

```
/opt/webgateway/home/broker/
```

The static files used to support the CSP samples:

```
/opt/webgateway/home/samples/
```

The static files used to support the Management Portal:

```
/opt/webgateway/home/sys/
```

13.6 Custom Error Pages

The **Error Pages** section of the global configuration screen allows you to customize Web Gateway error messages and system responses. These can be set on a global or per-InterSystems IRIS server basis. To customize the default CSP responses, perform the following:

1. From the Web Gateway management pages main menu, select **Default Parameters**.
2. In the **Error Pages** section, enter the name of the CSP page that you wish to replace the corresponding Gateway page with. Enter the full physical path to your CSP page, or enter a path relative to that of the Web Gateway.
3. Select **Save Configuration**.

You can customize the following Web Gateway system responses:

Server Error

Page to display when the Web Gateway encounters an internal error. For example, an error occurs if there is a problem communicating with an InterSystems IRIS server. The specific error is always recorded in the Web Gateway Event Log.

Server Busy

Page to display when all available CSP connections are in use.

Server Unavailable

Page to display when the InterSystems IRIS server (or application) has been deliberately disabled from within the configuration.

Server Timeout

Page to display when the request has timed out.

Connection Closed

Page to display when you log out of a state-aware session.

13.7 Event Logging Parameters

The **Event Log Level** field specifies the information that Web Gateway writes to the Web Gateway Event Log. Logging options are defined as a string of characters, each character representing a logging command. The value set here for the log level is the default for the system (that is, all InterSystems IRIS servers). Except where noted, you can set a different value for individual InterSystems IRIS servers.

You can view or clear the log from the CSP Web Management page menu. The logging parameters, shown below, are used mainly for troubleshooting:

Logging Option	Function
E	Record all errors. This option allows you to monitor connection failures.

Logging Option	Function
V	Verbose: Record the basic connection dialog between the Web Gateway and an InterSystems IRIS system. Use this option to record the strategic points of communication between the Web Gateway and an InterSystems IRIS server. There are 7 levels to this command (1 to 7). Each successive level records more detailed information. The levels are accumulative. For example, level V3 includes all log information specified for V1 and V2.
EV	Enter EV to turn on basic event logging. The higher log levels generate a large volume of data in the log file and should only be used for diagnosing problems. For production systems it is recommended that the log level should be set to no higher than EV.
V1	Same as V.
V2	In addition to the information specified for previous levels, this level records: • Information regarding basic connection management between the Web Gateway and InterSystems IRIS (Start and Close points for each connection). • Transmission interrupts received from the browser. • Cases where connections to InterSystems IRIS are forcefully closed (due to a lack of response from InterSystems IRIS or other errors where the connection cannot be recovered). • Access violations in state-aware (preserve mode 1) sessions (For example, Invalid Session ID).
V3	In addition to the information specified for previous levels, this level records: InterSystems IRIS headers and HTTP headers. *Note:* When this logging level is specified for an individual server, request headers are not logged, but response headers and other data will be.
V4	In addition to the information specified for previous levels, this level records: Information regarding the serialization of state-aware sessions. *Note:* When this logging level is specified for an individual server, request headers are not logged, but response headers and other data will be.
V5	In addition to the information specified for previous levels, this level records the contents of data buffers received from, and sent to, InterSystems IRIS via the WebSocket protocol. All data framing (where applicable) is also recorded. Finally, further information about the nature of the WebSocket created is also recorded at initial connection time. For example: • WebSocket Connection • WebSocket Connection Accepted by InterSystems IRIS: WSClassProtocolVersion=2; SharedConnection=0; NoDataFraming=2; BinaryData=1; *Note:* When this logging level is specified for an individual server, request headers are not logged, but response headers and other data will be.

Logging Option	Function
V6	In addition to the information specified for previous levels, this level records: • Headers to the data blocks sent to InterSystems IRIS. • Request Data from the web server (except multipart attachments). • Headers to the data blocks received from InterSystems IRIS. *Note:* When this logging level is specified for an individual server, request headers are not logged, but response headers and other data will be.
V7	In addition to the information specified for previous levels, this level records: The full content returned from InterSystems IRIS. *Note:* When this logging level is specified for an individual server, request headers are not logged, but response headers and other data will be.
V9	Record incoming HTTP request data. The full bodies of all HTTP requests are recorded. This log directive can be further extended and refined. • v9r: In addition to logging all HTTP requests, record all HTTP responses. • v9a: Record all HTTP requests to http.log in the Web Gateway home directory. • v9b: Record all HTTP requests on a per-session basis. Log files of the form http[session_id].log is created in the Web Gateway home directory, where *session_id* is the 10-Byte session ID. • v9m: Log all multi-part posts in the Web Gateway home directory. The raw incoming HTTP request are recorded together with the individual components in both their encoded and decoded form. *Note:* The forms V9, V9r, V9a, and V9b have no effect when specified for individual servers. These forms of logging can be enabled only at the default level.
s	Sessions: Record information about the management of session tokens: • The point at which new session IDs are allocated. • For existing sessions: an indication as to whether the session token was extracted from a cookie or the form/URL variable CSPCHD. • For all requests: the final session ID transmitted to InterSystems IRIS. *Note:* This logging option has no effect when specified for individual servers. This option can be enabled only at the default level.

Logging Option	Function
c	Connections: Record information about connections made using the Kerberos Library (IRISCONNECT). Include a log level of lowercase `c` to instruct the Web Gateway to record a complete audit of all IRISCONNECT functions called, together with the input parameters supplied and the result returned. For the sake of brevity, the content of the input and output buffers to and from InterSystems IRIS are not recorded at this level. Set a log level of uppercase `C` to record, in addition to the IRISCONNECT function calls, the contents of the input and output buffers. In addition to the logging facilities provided by the Web Gateway, it is possible to instruct the IRISCONNECT library to generate a detailed trace recording its internal processes. To additionally request that a IRISCONNECT trace be generated, add a digit to the `c` directive to indicate the type of trace required. For example, a log level of `c3`, in addition to the standard Gateway log entries, generates a level 3 IRISCONNECT trace. Valid IRISCONNECT trace levels are 1 to 6 and are defined as follows: • 6 — errors • 5 — warnings • 4 — informational message • 3 — output data • 2 — input data • 1 — normal events Unlike the Web Gateway log levels, the IRISCONNECT trace is less verbose at the higher log levels. Log level `1`, therefore, provides the most detailed trace file. The Web Gateway instructs the IRISCONNECT library to maintain its trace in a file called *irisconnect.log* located in the Web Gateway home directory. The security considerations and permissions for this file are the same as those for the Web Gateway Event Log. *Note:* An IRISCONNECT trace can only be activated on a per-process basis, so it cannot be truly isolated to a server. Once configured, trace log generation is not triggered until a new SSL connection is attempted.

Logging Option	Function
t	Transmission: Record the raw data buffers received by and dispatched by the Web Gateway. The format for this option is: `t[x][y]`. The value of `x` instructs the Web Gateway to record data buffers transmitted between the Web Gateway and InterSystems IRIS and the value of `y` instructs the Web Gateway to record data buffers transmitted between the Web Gateway and the client, via the hosting web server. `x` and `y` can take the following values: • 0: No transmission data to be recorded. • 1: Record request data only. • 2: Record response data only. • 3: Record request and response data. Using lowercase `t` results in the Web Gateway recording just the first 256 Bytes of transmitted data for each buffer. Using uppercase `T` results in the Web Gateway recording the full data buffer. All non-printable characters are recorded in their escaped form. *Note:* When this logging level is specified for an individual server, `y` options record response buffers sent to the client, but not incoming request buffers from the client.
p[n]	Performance: Instructs Gateway to capture information to assess the performance of the CSP installation. *n* is the number of seconds (total service time) below which data is not recorded for a request. For example, a directive of `p` records data for all requests, `p2` records data for requests taking longer than 2 seconds to service. The following information is recorded. • Total time to service request: The total time spent in servicing the request (from the time it reaches the Web Gateway to the time at which the last Byte of response data leaves the Web Gateway environment. • Obtain [NEW] connection to InterSystems IRIS: Time taken between the request reaching the Web Gateway and a connection to InterSystems IRIS being reserved for the purpose of servicing the request. The message recorded indicates if a new connection is created during this time (as opposed to an existing one being reused). • Send request to InterSystems IRIS: Time taken between the first and last Byte of request data being read from the web server and dispatched to InterSystems IRIS. • Processing request in InterSystems IRIS: Time taken between the last Byte of request data being dispatched to InterSystems IRIS and the first Byte of response data being received by the Web Gateway. • Receive response from InterSystems IRIS: Time taken between the first and last Byte of response data being received from InterSystems IRIS and dispatched to the web server.

Logging Option	Function
p[n]([v])	Provides the capability to conditionally activate verbose logging based on the results of the performance monitor. Useful in situations where you want to record further information about requests that take more than a certain time to process. *n* is the optional lower time-to-service threshold (in seconds) for which performance data is recorded and *v* is the verbose log level required. This mechanism applies to verbose Event Log and HTTP logging settings. A request to record error information, *e* is always applied to all requests regardless of whether or not they are recorded by the performance monitor. For example: `ep5(v9)` This option records any errors encountered while processing requests for all requests (*e*). In addition, it records the HTTP request message (*v9*) but only for requests that take longer than 5 seconds to process (*p5*). Gateway event logging is designed to have a minimal impact on performance and to occupy a small footprint in terms of system resources consumed. Therefore, the following limitations apply: • Only one verbose log level can be specified per individual setting. In other words it is not possible to specify a *v9* level for requests recorded by the performance monitor and a *v2* level for all other requests. For example, if `v2p5(v9)` is specified then only the conditionally applied *v9* level is honored. • The Web Gateway configuration allows you to specify an Event Log level both globally and on a per server basis. When verbose logging is in force, some records are written before the target InterSystems IRIS server has been identified so, for best results, it is best to specify conditional logging at the global level under Default Parameters.

Logging Option	Function
pp[n]	Provides detailed timing information as follows: • Pre-processing of request: Time taken to identify the target InterSystems IRIS server; includes the initial handover from the web server and basic request processing to identify the server. • Obtain [NEW] connection to InterSystems IRIS: Time taken to allocate a connection to the appropriate InterSystems IRIS server. Indicates whether a new connection is created (instead of an existing one reused). • Format request: Time taken to parse and format the request message for transmission to InterSystems IRIS. • Send request to InterSystems IRIS: Time taken between the first and last byte of request data read from the web server and dispatched to InterSystems IRIS. • Processing request in InterSystems IRIS: Time taken between the last byte of request data dispatched to InterSystems IRIS and the first byte of response data received by the Web Gateway. • Post-processing of response(b): When a content-length header is required, this reports the time taken for the dispatch of the response data back to the client via the web server. • Post-processing of response(c): Time taken between the dispatch of the response and the Web Gateway being ready to read the response footer data from InterSystems IRIS. The footer data is part of the internal communication protocol between the Web Gateway and InterSystems IRIS and includes control information (For example: instructions to change the preserve setting for the session). • Receive footers from InterSystems IRIS: Time taken to receive the response footer data from InterSystems IRIS. • Post-processing of footers: Time taken to process footer data and respond to instructions received. • Release connection to InterSystems IRIS: Time taken to release the active connection to InterSystems IRIS. • Cleanup: Time taken to release resources used in servicing the request and return control back to the hosting web server.

Logging Option	**Function**
W (or w)	On Windows, generates a memory dump if a crash occurs. This option is case insensitive. On AIX, generates a core file using the `gencore` utility. This option is case insensitive. On Linux or MacOS, this option is case sensitive. Specifying `w` generates a standard core dump using `gcore`. Specifying `W` dumps all memory mappings (including shared memory) into the core file by executing `gcore -a`. On Unix systems, the following preconditions must hold: • The `gcore` (Linux or MaxOS) or `gencore` (AIX) is present on the machine and is available through the PATH environment variable. On Linux and MacOS systems, the version of `gcore` must support the `-a` command line option. • Web server worker processes need write permission to the directory where the Web Gateway modules are located. In default installations this directory is `/opt/webgateway/bin`. • A non-root process needs permission to produce a core dump of another process running under the same user ID. On MacOS, System Integrity Protection must be disabled. On Linux, if the Yama security module is present (as on RHEL and Ubuntu systems), execute the following command to grant the required permission until the next reboot: `echo 0 | sudo tee /proc/sys/kernel/yama/ptrace_scope`. To permanently grant this permission, create or edit the file `/etc/sysctl.d/10–ptrace.conf`. If there is a line starting with “kernel.yama.ptrace_scope”, change it to “kernel.yama.ptrace_scope = 0”. If no such line exists, add “kernel.yama.ptrace_scope = 0”, then execute `sysctl –p`. **Note:** For security reasons, it is recommended that such permission be granted only temporarily.

14

Configuring Server Access

This page describes how to configure servers to which the InterSystems IRIS® Web Gateway connects. For these configuration tasks, you use the Web Gateway management pages. Other articles describe how to configure default settings and applications.

Each InterSystems IRIS system accessed by the Web Gateway must be defined here. Any unspecified optional parameters or custom system forms are automatically inherited from the Web Gateway default settings.

14.1 Adding a Server Configuration

To configure access to an InterSystems IRIS server:

1. From the Web Gateway management pages main menu, select **Server Access**.
2. Select **Add Server**. The second configuration screen appears. Note that many parameter fields have default settings.
3. In the **Server Name** text box, enter a unique, descriptive name for the server. This logical name is used to identify the server configuration in the CSP configuration file.
4. Enter the system parameters (described below) for this server configuration.
5. Select **Save Configuration**.

14.1.1 Server Access Parameters

The set of base server configuration parameters are as follows:

Server Configuration Parameter	Function
Server Name	Logical name to identify this server configuration in the CSP configuration file.
Service Status	Allows you to enable and disable this configuration (default is Enabled).
IP Address	The DNS host name or IP address (physical or virtual) of the InterSystems IRIS server to connect to.
Superserver TCP Port	The TCP port number on which the InterSystems IRIS server is listening for incoming connections. This is the TCP port number of the InterSystems IRIS superserver which is 1972 by default, but may be different if multiple instances are deployed on the same system..

Server Configuration Parameter	Function
Configuration is Mirror Aware	Configures a mirror primary as a server to access mirrored databases. In a failover or disaster recovery, the connection is redirected. By default, not selected. *Note*: If you have configured a mirror VIP, do not configure a mirror aware Web Gateway, which causes the Web Gateway to ignore the VIP. Instead, simply configure the Web Gateway to connect to the VIP like any other client. In general, use of a mirror aware Web Gateway is the appropriate choice only in unusual circumstances. To configure, enter the IP address of one of the failover members. From this failover member, the Web Gateway obtains a list of the failover and disaster recovery (DR) async members in the mirror and connects to the current primary based on this list (and not the VIP even if one is configured). The CSP connection fails until a primary is found. Once the connection is established, if the mirror fails over, the Web Gateway changes the connection to the new primary. If no primary can be found among the failover members, the Web Gateway attempts to find one among the DR asyncs in the list, which enables it to reestablish the connection when a DR async is promoted to primary in a disaster recovery situation. For details, see Redirecting Application Connections Following Failover or Disaster Recovery in Mirroring in the *High Availability Guide*.

14.1.2 Stateless Parameters

The set of parameters relevant to stateless connections are as follows:

Stateless Parameter	Function
Minimum Server Connections	The Web Gateway implements process affinity. This means that it always attempts to reconnect sessions to the same InterSystems IRIS process that serviced its previous request if possible. This parameter specifies the minimum number of connections that the Web Gateway should make to the InterSystems IRIS server before starting to share the connections among many clients. The higher this number, the more effective process affinity is. The default value is 3.
Maximum Server Connections	This is the absolute maximum number of connections that the Web Gateway is allowed to make to the InterSystems IRIS server. If concurrent usage exceeds this number, the Web Gateway starts to queue requests. Requests remain in the queue until an InterSystems IRIS connection becomes available to service the request or the *Queued Request Timeout* is exceeded. This is unspecified by default, indicating that the only hard maximum is the number of maximum connections for the Web Gateway, which is 1024 by default.
Maximum Connections per Session	This represents the maximum number of connections to InterSystems IRIS that can be concurrently used by an individual session. The default value is 3.

14.1.3 Connection Security Parameters

Connection Security settings are required by the Web Gateway to access the InterSystems IRIS server. These parameters are discussed in greater depth in a later section. The set of parameters relevant to connection security are as follows:

Connection Security Parameter	Function
Connection Security Level	Level of security required for connecting to the InterSystems IRIS server. Select one of the options: • Password • Kerberos • Kerberos with Packet Integrity • Kerberos with Encryption • SSL/TLS
Username	Username required by the Web Gateway for connecting to the InterSystems IRIS server.
Password	Password required by the Web Gateway for connecting to the InterSystems IRIS server.
Password (Confirm)	When you create a new password, confirm the new password by entering it again.
Product	Product being connected to (InterSystems IRIS).
Service Principal Name	Service principal name. A Generate button is provided for creating a default name with respect to the target InterSystems IRIS server.
Key Table	Full path to the Key Table file.

14.1.4 SSL/TLS Parameters

The following parameters are relevant only to installations using SSL/TLS to secure connections between the Web Gateway and InterSystems IRIS.

SSL/TLS Parameter	Function

SSL/TLS Parameter	Function
Minimum SSL/TLS Protocol Version	Minimum version of the SSL/TLS protocol to use. The following options are provided: • TLSv1.0 • TLSv1.1 • TLSv1.2 • TLSv1.3 (on platforms where it is supported) On platforms where TLSv1.3 is supported, the default value is `TLSv1.2`. Otherwise, the default value is `TLSv1.1`.
Maximum SSL/TLS Protocol Version	Maximum version of the SSL/TLS protocol to use. The following options are provided: • TLSv1.0 • TLSv1.1 • TLSv1.2 • TLSv1.3 (on platforms where it is supported) On platforms where TLSv1.3 is supported, the default value is `TLSv1.3`. Otherwise, the default value is `TLSv1.2`.
SSL/TLS Key Type	The type of SSL/TLS key file (based on the algorithm used to generate it). The following options are provided: • DSA — Digital Signature Algorithm • RSA — Rivest, Shamir, and Adelman (inventors of the algorithm) The default is RSA.
Require Peer Certificate Verification	If checked, requires peer certificate verification for this installation.
SSL/TLS Cipher Suites (TLSv1.2 and below)	Cipher suites for TLSv1.2 and below. The default is `ALL:!aNULL:!eNULL:!EXP:!SSLv2`.
SSL/TLS Cipher Suites (TLSv1.3)	Cipher suites for TLSv1.3. The default is `TLS_AES_256_GCM_SHA384:TLS_CHACHA20_POLY1305_SHA256:TLS_AES_128_GCM_SHA256`. Available only on platforms where TLSv1.3 is supported.
SSL/TLS Certificate File	The full path to the SSL/TLS certificate file for the Web Gateway. Supported file formats for certificate files are the same as those supported for InterSystems IRIS TLS Configurations. `Example: C:\InterSystems\certificates\clicert.pem`
SSL/TLS Private Key File	The full path to the private key associated with the Web Gateway's SSL/TLS certificate. Supported file formats for certificate files are the same as those supported for InterSystems IRIS TLS Configurations. `Example: C:\InterSystems\certificates\clikey.pem`

SSL/TLS Parameter	Function
SSL/TLS CA Certificate File	The full path to the certificate for Certificate Authority (CA) for the Web Gateway’s certificate. Supported file formats for certificate files are the same as those supported for InterSystems IRIS TLS Configurations. `Example: C:\InterSystems\certificates\cacert.pem`
SSL/TLS Private Key Password	The password to the SSL/TLS Private Key.

14.1.5 Optional Parameters

The descriptions of the Optional Parameters are given in Configuring Default Parameters, If any of these parameters is blank, its value is inherited from the Web Gateway global configuration described in Connections to InterSystems IRIS.

14.1.6 Error Pages

The Error Pages parameters let you customize the Web Gateway responses. If not specified, the parameters are inherited from the global configuration. For a description of each parameter, see Custom Error Pages.

14.2 Copying a Server Configuration

You can quickly configure a new server by copying the configuration entry of an existing server. Having done this, both configuration entries are identical, except for the server name. You can then edit the second configuration and make changes to it (such as changing the IP address).

This feature is also useful for fine-tuning a configuration. By creating a second (temporary) configuration for a server, you can test parameter changes without worrying about losing the original configuration.

To copy an existing server configuration:

1. From the Web Gateway management pages main menu, select **Server Access**.
2. At the **Server Access** screen, select an existing server name.
3. Select the **Copy Server** option.
4. Select **Submit**. The second configuration screen appears.
5. In the **Server Name** text box, enter a unique, descriptive name for the new server.
6. Select **Save Configuration**.

14.3 Disabling Access to a Configured Server

Use this facility to prevent users from accessing a configured InterSystems IRIS server through this Gateway installation.

To disable access to a server:

1. From the Web Gateway management pages main menu, select **Server Access**.
2. At the **Server Access** screen, select an existing server name.

3. Select the **Edit Server** option.
4. Select **Submit**. The Server configuration screen appears.
5. For the **Server Status** parameter, select **Disabled**.
6. Select **Save Configuration**.

To re-enable access, repeat the procedure and select **Enabled** at Step 5.

14.4 Deleting a Server Configuration

To delete a configured server:

1. From the Web Gateway management pages main menu, select **Server Access**.
2. At the **Server Access** screen, select a server name.
3. Select the **Delete Server** option.
4. Select **Submit**.
5. Confirm by selecting **YES : DELETE**.

15

Configuring Application Access

This page describes how to configure applications to which the InterSystems IRIS® Web Gateway connects. For these configuration tasks, you use the Web Gateway management pages. Other articles describe how to configure default settings and servers.

Each web application must have the path to its CSP files configured. The configuration for each path identifies the InterSystems IRIS server responsible for running the application. Optional directives for specifying failover and load-balancing are included in the application path's configuration. The default application path, root, (/) is automatically configured when the Web Gateway is started for the first time. Inheritance is applied to application paths. For example, if a CSP request asks for a file in /Accounts/Invoices and there is no configuration for /Accounts/Invoices, the Web Gateway uses the configuration defined for /Accounts. If this is not defined, the configuration for the default path of / is used.

15.1 Adding an Application Path

To configure the path to an application:

1. On the Web Gateway management pages main menu, select **Application Access**.
2. Select **Add Application**. Note that many parameters have default settings.
3. In the **Application Path** text box enter a unique path for the application. This path is the path which appears in the application URLs.

   Note: An InterSystems IRIS installation creates a new /csp configuration. If you have configured /csp as your application, your configuration is overwritten when you install a new build of InterSystems IRIS. To maintain your application configuration, enter a path other than /csp.

   Any directory under /csp works fine, such as /csp/myapplication, but the path cannot contain any dots (periods), because these lead to ambiguity for the Web Gateway. In this example: /csp/samples/menu.csp/csp/aaa/bbb/ccc.cls, the Web Gateway could either interpret this as a request for /csp/samples/menu.csp/csp/aaa/bbb/ccc.cls or as a REST request for/csp/samples/menu.csp (where PATH_INFO is /csp/aaa/bbb/ccc.cls). The Web Gateway, working in the web server environment, has no way of resolving these ambiguities.

   CSP is case-sensitive. Specify your path names consistently when you are configuring CSP.

4. Enter the other configuration path and server parameters (described in the tables below) for this application.
5. When you have finished, select **Save Configuration**. Changes you make to the application configuration take effect as new user sessions are created for that application path. Existing users are unaffected.

15.1.1 Application Path Configuration Parameters

The set of base parameters are as follows:

Parameter	Function
Service Status	Enable and disable access to an application via the application path (default is Enabled).
Web Server Physical Path	Path to the corresponding directory on the web server. This setting is particularly important for Microsoft IIS systems where each path configured must be set up as a virtual directory under the web server configuration. Each virtual directory defined within IIS must have a physical path associated with it. The purpose of this additional configuration procedure for IIS is to allow the paths used by InterSystems IRIS (specifically by the CSP engine) to be defined with execute permissions. The default is for execute (and hence access to the CSP engine) to be denied.
Extra CGI Environment Variables	Comma-separated list of additional CGI environment variables to be returned to the InterSystems IRIS environment with each and every request. The commonly-used CGI environment variables are automatically sent with each request. Enter the wildcard character (*) to instruct the Web Gateway to send all environment variables supplied by the web server to the InterSystems IRIS server with each request.
Process with this class	Process files in this path with the specified class. This allows you to build your own request handlers in CSP.
GZIP Compression	Enable or disable GZIP compression for all CSP pages returned in this path (default is Disabled).
GZIP Minimum File Size	Minimum response size, in bytes, for which GZIP compression is invoked. Default is 500 bytes.
GZIP Exclude File Types	This is a list of file types to be excluded from GZIP compression. Files to be excluded can be listed by MIME type (such as image/jpeg) or by common extension (such as jpeg). By default, these common (natively compressed) image files are excluded: `GZIP Exclude File Types: jpeg gif ico png gz zip mp3 mp4 tiff` Separate additional types or extensions with a space.

Parameter	Function
Response Size Notification	This parameter provides configurable control over the method used by the Web Gateway to notify clients of the amount of data contained in each response. Web clients typically require some form of response size notification if HTTP KeepAlive connectivity is used. Under these circumstances, the Web Gateway defaults to using chunked transfer encoding, provided HTTP v1.1 is in use. If an earlier HTTP protocol is in force it buffers the response data received from InterSystems IRIS and generate a content-length header instead. Also, in cases where the entire response fits into one output buffer a content-length header is generated instead of using chunked transfer. There are scenarios in which it is desirable to instruct the Web Gateway to specifically use one method or the other. For example, in cases where HTTP v1.1 is used but some intermediary (such as a proxy) is unable to properly support chunked transfer. Also, while not sending any form of size notification (such as, where the *close connection* event is used as the response terminator) should be supported by all web clients, it is nevertheless recommended as 'good practice' that all responses should be accompanied by some form of size notification. Indeed, some clients require this. The following options are provided: • Chunked Transfer Encoding and Content Length (the default) • Chunked Transfer Encoding • Content Length This parameter is supplemented with a check box to instruct the Web Gateway to always generate a size notification for all requests regardless of whether or not KeepAlive is used. There is a 500 kilobyte limit on the size of HTTP responses that specify a content-length header, as opposed to chunked responses. If you exceed this limit, a warning message appears in the CSP log: `WARNING: Unable to generate a 'Content-Length' header directive for this oversize response (Current size:` *size*`; Maximum buffer size allowed: 500000)`
KeepAlive	Enable or disable HTTP **KeepAlive** connectivity for this path. Default is `No Action` in which case the **KeepAlive** status is determined by the HTTP response headers for each request.
Non-Parsed Headers	Enable or disable **Non-Parsed Headers** protocol for this path. Default is **Enabled** in which case HTTP response headers are streamed directly back to the client. If this property is disabled, the response headers are submitted back to the hosting web server. This gives the web server the opportunity to parse the headers and invoke any output filters that may be indicated. For example, the Apache Group's mod_deflate facility. Note that for the Apache web server, if keep-alive is enabled, then the response headers are submitted back to Apache regardless of the Non-Parsed Headers setting.

15.1.2 Server Parameters

You can define a list of InterSystems IRIS servers to use for an application and the purpose for which they are to be used.

Parameter	Function
Use Alternate Servers For	The first server listed, **Server 0**, is the default InterSystems IRIS server. It is used first. Other listed servers can be used for load balancing or failover, depending on the option checked. • **Fail-Over** If the first server fails (becomes unavailable), use an alternative. • **Load-Balancing and Fail-Over** . If the first server fails, use a server that is configured as either failover or load-balancing.
Server #:	List of servers. The configuration screen initially shows only three server slots, but additional slots appear that enable you to define any number of alternative servers. Each server can be checked as **Enabled** or **Disabled**. Default is always **Enabled**. See Load Balancing and Failover Between Multiple InterSystems IRIS Server Instances for more information.

15.2 Copying an Application Path Configuration

You can quickly configure a new application path by copying the configuration entry of an existing path and editing it.

Tip: This feature is also useful for fine-tuning a configuration. By creating a second (temporary) configuration for an application path, you can test parameter changes without worrying about losing the original configuration.

To copy an existing application path configuration:

1. From the Web Gateway management pages main menu, select **Application Access**.
2. On the **Application Access** screen, select an existing application path.
3. Select **Copy Application**.
4. Select **Submit**.
5. In the **Application Path** text box, enter a new and unique application path.
6. Select **Save Configuration**. The new application configuration takes effect as new user sessions are created for the new application path. Existing users are unaffected.

15.3 Disabling Access via an Application Path

Use this facility to prevent users accessing a configured application through this Web Gateway installation.

To disable access via an application path:

1. From the Web Gateway management pages main menu, select **Application Access**.
2. At the **Application Access** screen, select an application path.

3. Select **Edit Application**.
4. Select **Submit**. The configuration screen for the application path appears.
5. For the **Application Status** parameter, select **Disabled**.
6. Select **Save Configuration**.

To re-enable access, repeat the procedure and select **Enabled** at Step 5.

15.4 Deleting an Application Path Configuration

To delete a configured application path:

1. From the Web Gateway management pages main menu, select **Application Access**.
2. At the **Application Access** screen, select an application path.
3. Select the **Delete Application** option.
4. Select **Submit**.
5. Force a restart of all the applications by restarting the web server.

16

Managing and Monitoring the Web Gateway

This page describes how to manage and monitor the InterSystems Web Gateway via the Web Gateway management pages.

16.1 Checking System Status

The **System Status** option displays the status of all active connections. You must be a system manager to use this feature. In each of the tables below, click a column head to sort by that column.

16.1.1 Connections to InterSystems IRIS

The first status table (Connections to InterSystems IRIS) displays information on connections to InterSystems IRIS®.

Item	Function
Connection Number	Number that the Web Gateway assigns to the connection. Your InterSystems IRIS license determines the number of possible connections.
Gateway PID	The Web Gateway (or hosting web server) process ID for the connection.
Server Name	Name of the InterSystems IRIS system connected to. Mirror members show current configuration name with mirror member name appended.
InterSystems IRIS PID	Process ID on the InterSystems IRIS server.
Status	Indicates whether information is being sent to or from the InterSystems IRIS system, as follows: `Free` — no information is being sent and the connection is ready to process the next request. `In Use` — information is being transmitted through the connection. `Private` — the connection is state-aware (preserve mode 1) and not free for general use. `Server` — the connection is being used by the InterSystems IRIS server.
Idle time/Timeout	Indicates the amount of time that the connection has been idle against the timeout applied to that connection. The timeout is the ‘No Activity Timeout’ for connections in the state-less pool and the ‘Application timeout’ for connections marked as ‘Private’ (state-aware).
Activity	Number of transactions (hits) the connection has processed.

Item	Function
Interrupt	For a connection with status 'In Use', an Interrupt button will attempt to interrupt the corresponding InterSystems IRIS process and return it to the state where it is ready to accept the next web request.
Close	If available, allows you to forcefully close down the connection by selecting it. See Closing Connections Manually.

16.1.2 InterSystems IRIS Servers Table

The second status table (InterSystems IRIS Servers) displays information on InterSystems IRIS servers.

Item	Function
Server Number	The number that the Web Gateway assigns to the server.
Server Name	Name of the InterSystems IRIS system connected to.
Mirror Member	For mirror-aware configurations, the name of the mirror member.
Mirror Status	For mirror-aware configurations, the name of the mirror configuration along with the mirror status of the server. The member type, Failover or Async, will be shown and the Primary will be labeled as 'Primary'.
Total Connections	Number of connections to the InterSystems IRIS system.
Connections In-Use	Number of connections that are currently in use (actively serving a Web request).
Private Connections	Number of connections that are currently in use as state-aware sessions (preserve mode 1).
Total Activity	Number of transactions (hits) the InterSystems IRIS system has processed.
Queued Requests	Number of Web requests that are held in a queue waiting for a free connection to the InterSystems IRIS system. Queued requests are an indication that the InterSystems IRIS license should be increased in order to maintain good performance.
Close	Close all connections on this InterSystems IRIS server. See Closing Connections Manually.

16.1.3 Application Paths Table

The third status table displays information for application paths.

Item	Function
Path Number	The number that the Web Gateway assigns to the application path.
Path	The application path.
Server Number	The number that the Web Gateway assigns to the InterSystems IRIS server.
Server Name	The name of the InterSystems IRIS system connected to.
Activity	The number of requests processed by this server for this path since the last Gateway.

Item	Function
Status	The status for this server, one of `Disabled`, `Enabled` or `Offline`. Also the current master (or primary) server in the set is indicated in this column.
Action	If a server is marked as `Offline`, this column contains a button allowing Administrators to mark it `Online/Enabled` again.

16.1.4 Web Gateway Cache Table

The fourth status table lists the forms held in the Web Gateway response cache.

Item	Function
Cached Forms	Name (including path) of cached form.
Cached Data (Bytes)	Amount of cached form data held in the Gateway (in Bytes).
Cache Blocks In Use	Total number of cache memory blocks in use.
Cache File	Name of physical file if permanent storage (on the web server host) is used to cache the file.
Cache Form Activity	Total number of times this form has been requested from the cache.
Clear	Clear this form from the cache. See Clearing the Cache.

16.1.5 Closing Connections Manually

If your InterSystems IRIS system shuts down while a CSP connection is still active, CSP continues to try to connect to the system until one of the following occurs:

- It successfully reconnects to the system.
- CSP is shut down.
- The connection is manually closed.

If your InterSystems IRIS system is scheduled for extensive downtime, you may want to close the connections. You can close sessions manually using the **Close** button on the **System Status** page.

Note that you can close the connections while the InterSystems IRIS system is down.

16.1.6 Clearing the Cache

Under certain circumstances, such as during the development process for web applications, it may be necessary to clear the Web Gateway cache. To do this:

1. From the Management Portal, navigate to **System Administration** > **Configuration** > **Web Gateway Management** and select **System Status**.
2. On the **System Status** page, there are a number of tables. To clear the cache, in the **Cached Forms** table, select the button in the **Clear** column (the right-most column) and the **Total** row (the bottom row). If the **System Status** page does not display a **Cached Forms** table, then there is no currently cached content. This may be because the cache has been cleared recently and nothing has been cached since then.

This action clears all cached content for the Web Gateway and removes the **Cached Forms** table from the page until there is new cached content.

16.2 Testing Server Connections

The **Test Server Connection** option is useful to test Web Gateway connectivity to your InterSystems IRIS systems. Note that you must be a system manager to use this feature.

To test CSP connectivity:

1. From the Web Gateway management pages main menu, select **Test Server Connection**.
2. Select the desired InterSystems IRIS system from the displayed list.
3. Select **Connect**.

Depending on your selection and the state of the server connection, you receive one of the following results:

Result	Meaning
CSP Test Form	The Web Gateway is working correctly and is able to connect to InterSystems IRIS. The form shows the basic parameters returned by the target InterSystems IRIS server (version and process ID).
Server Availability Error	This error occurs any time that InterSystems IRIS is unreachable. If there are no additional error messages, check to ensure your InterSystems IRIS system is running. Also, check the Web Gateway Event Log for specific connectivity error messages.

In all cases where an error condition is returned, check the Web Gateway Event Log for additional and more specific error information. Consider raising the log level to capture even more diagnostic information where necessary.

16.3 Viewing the Event Log

Use the **View Event Log** option from the Web Gateway management pages main menu to read the contents of the Web Gateway Event Log.

When the log file reaches its capacity as specified by **Event Log Rotation Size**, it is copied to filename.old, where *filename* is the full original filename. If **Event Log Rotation Size** is blank (the default), the file grows until manually cleared. To save all logs in files named with date and time, select **Retain All Log Files** on the **Default Parameters** page. Each log entry is marked with a header record which captures the date, time and additional information with respect to the context in which the log entry was made.

Log entries follow the same machine-readable format which an InterSystems IRIS instance uses for its structured logging feature. This means you can use the same third-party tools to analyze the Web Gateway Event Log which you use to analyze the logs for your InterSystems IRIS instances. For added efficiency, the Web Gateway Event Log uses several of the same field names which other structured logs use: `when`, `level`, `event`, `pid`, and `text`. On the **View Event Log** page, the `text` and `details` fields are presented without their field names. However, the log entries available in the CSP.log file fully subscribe to the name-value pair (NVP) format.

What follows is an example of a Web Gateway Event Log entry as it appears in CSP.log. Each entry in CSP.log occupies one line. However, for readability this example is divided up onto multiple lines (marked by the \ character).

```
local-time="Thu Jul 21 11:39:20 2022" \
  wg-build="RT 2202.1825 (win32/apapi:srv=2.4.52/apr=1.7.0/apu=1.6.1/mpm=WinNT)" \
  wg-log-level=0 when="2022-07-21 15:39:20.831" level=WARNING event=WebGateway.SessionOpen \
  pid=17216 thread-id=2072 text="Warning" \
  details="A Connection between the Web Gateway and InterSystems IRIS has been found to be \
  closed (possibly as a result of an intermediary, such as a firewall, timing-out the TCP session)"
```

Select **Clear Log** to clear all current entries from the Event Log.

The Log can be displayed in either ascending date/time order (the default) or descending date/time order. Select the link at the top right-hand corner of the form to reverse the display order. This link acts as a toggle between the two modes.

Finally, most browsers are unable to render more than about 1MB of log data in a single form. Therefore, as the volume of log data returned approaches 1MB, the Web Gateway terminates the display and prompts for the next page of data. See the **More** link at the bottom left-hand corner of the form. Additionally, a **Top** link is provided at the bottom right-hand corner of the form to allow you to quickly go back to the first form in the series.

16.4 Using the HTTP Trace Facility

The HTTP trace facility is accessed via the **View HTTP Trace** option.

The trace window consists of two main frames. The left-hand frame contains a list of HTTP requests processed by the Web Gateway by time and a unique request ID (assigned by the Web Gateway). As each request is selected, the request and response data is shown in the right-hand frame. Links allow easy navigation between the request and response message.

Note: Note that the HTTP request headers reported by the Web Gateway are reconstituted because the hosting web server always assumes responsibility for parsing the request headers. The Web Gateway reassembles the complete header from the CGI environment variables supplied by the web server. However, if a request is passed directly through the NSD component (that is, effectively bypassing the web server), then the request header recorded is byte-for-byte the same as it was when dispatched from the client.

17

Protecting Web Gateway Connections to InterSystems IRIS

This page describes options for protecting connections from the Web Gateway to InterSystems IRIS®. For more details on CSP authentication, see the Authentication Guide. Web Gateway connections to InterSystems IRIS can be protected according to the following levels of security:

1. Minimal connection security (not recommended)
2. Simple username- and password-based authentication
3. Kerberos-based authentication and data protection
4. SSL/TLS-based authentication and data protection

Remember that security applied here is solely for the purpose of authenticating the Web Gateway host to the InterSystems IRIS server. It protects against the unauthorized creation of connections to the CSP engine (`%cspServer`). It does not, however, identify an individual *user* of a web application. A user of a web application can only be positively identified by whatever user login facility is provided by the application itself. For example, a Systems Manager logging on to the Management Portal can only be identified by the username and password supplied to the Management Portal login form.

The stateless nature of the Web should also be borne in mind. There is no fixed relationship between a Web Gateway connection to InterSystems IRIS and an individual user of a web application. Many users share the same connection.

Authenticating the Web Gateway to InterSystems IRIS at connection time is important. If an attacker can impersonate a Web Gateway, it can redirect traffic through a system under his control (by technical means and/or social engineering) and read and/or modify data at will. This is distinct from authenticating individual users to a web application. The Web Gateway's InterSystems IRIS username and password, Windows network credentials, or UNIX® Kerberos key table should never be used by ordinary users.

17.1 Configuring Connection Security for the Web Gateway

To configure the connection security for the Web Gateway, in all cases you use the Web Gateway management pages. The relevant options are in the **Configuration** > **Server Access** > **Connection Security** section, which provides the following settings:

- **Connection Security Level** — Choice of:

  - Password
  - Kerberos
  - Kerberos with packet integrity
  - Kerberos with encryption
  - SSL/TLS

- **Username**
- **Password**
- **Product**
- **Service Principal Name**
- **Key Table**

17.2 Minimal Connection Security (Not Recommended)

In minimal connection security, the **Connection Security Level** is set to **Password** and the **Username** and **Password** fields are left empty.

In this mode, there is a minimal level of security applied to the connection between the Web Gateway and InterSystems IRIS.

In this mode of operation, ensure that the Web Gateway service (`%Service_WebGateway`) together with the username under which it operates (for example, `CSPSystem`) is not expecting any form of authentication.

17.3 Simple Username/Password Authentication

In username/password, the **Connection Security Level** is set to `Password` and values are supplied for the **Username** and **Password**.

This is the simplest form of authentication that can be applied between the Web Gateway and InterSystems IRIS.

Remember that passwords are a weak form of authentication since they must be sent over the network as plain text for authentication in InterSystems IRIS. Network sniffing is easy to do and can be used to reveal these passwords. Passwords used in this configuration option must be held in the Web Gateway configuration file in accordance with the following guidelines.

In all cases, the default username and password used for the Web Gateway is as follows. The installation process creates the CSPSystem user for this purpose. This user (CSPSystem or any other) should have no expiration date; that is, its Expiration Date property should have a value of 0.

```
Username: CSPSystem
Password: SYS
```

For Windows, passwords are encrypted in the Web Gateway configuration file using functionality provided by Microsoft's Data Protection API (DPAPI). The Web Gateway Management **Default Parameters** page handles the encryption of passwords.

Note: This password encryption is used for Windows because ordinary Windows user accounts are occasionally granted membership in the Administrators Group, although this is not recommended practice for production systems. Encrypting the password offers a higher level of protection for all Windows installations.

17.3.1 Passwords Introduced from Outside

Occasionally, you need to introduce a password outside the context of the Web Gateway Management pages, for example, if the Web Gateway configuration is set up by custom configuration scripts. In this case, the password should be filed as plain text and the Web Gateway encrypts it when it is started for the first time. If you use this method, the password to the Gateway Management forms cannot start with '1' or 'PBKDF2|', and the password to IRIS cannot start with ']]]'. If you need passwords that start with these characters, use the `CSPpwd` utility at the OS command prompt instead. This utility encodes passwords held in the Web Gateway configuration file. The general form is:

`CSPpwd` *<path to the CSPx.so|dll library> <context> <clear text password>*

Where:

- context = 0: Password to the Gateway Management forms
- context = 2: Password to Cache/IRIS servers

The encoded password is written to the standard output.

Examples (Windows):

```
CSPpwd C:\cachesys\csp\bin\CSPx.dll 0 MyGatewayManagementPassword
CSPpwd C:\cachesys\csp\bin\CSPx.dll 2 MyIRISServerPassword
```

Examples (UNIX®):

```
CSPpwd /opt/cspgateway/bin/CSPx.so 0 MyGatewayManagementPassword
CSPpwd /opt/cspgateway/bin/CSPx.so 2 MyIRISServerPassword
```

17.3.2 Passwords Encrypted on Other Computers

Because the web server hosting the Web Gateway operates within a protected environment where there is no available user profile on which to base the encryption, it must use the machine store rather than the user store. Consequently, it is not possible to decrypt a Web Gateway password that was encrypted on another computer. This creates a situation for clustered environments in which the configuration file is on a shared drive and is shared among multiple participating computers. Only the computer that actually performs the password encryption can decrypt it. It is not possible to move a configuration file containing encrypted passwords to another computer; the password must be reentered and reencrypted on the new machine.

Here are some possible approaches to this problem:

- Use a machine outside of the cluster as the web server.
- Each time you fail over, reset the same password in the Web Gateway.
- Configure each computer participating in the cluster so that it has its own copy of the Web Gateway configuration file on a disk that does not belong to the cluster. InterSystems IRIS maintains the file in the directory hosting the Web Gateway DLLs. Save and encrypt the password on each individual computer before introducing the node to the cluster.

  For example, where *Disk C* from each machine does not belong to the cluster and InterSystems IRIS is installed on *Disk S*, you may have the following:

  CLUNODE-1: A copy of CSP.ini with password XXX encrypted by CLUNODE-1

  CLUNODE-2: A copy of CSP.ini with password XXX encrypted by CLUNODE-2

- Disable password encryption by manually adding the following directive to the configuration file file before starting the Web Gateway and adding the passwords:

```
[SYSTEM]
DPAPI=Disabled
```

17.4 Kerberos-based Authentication and Data Protection

In Kerberos-based Authentication and Data Protection, three levels of authentication (and data protection) are provided through the **Connection Security Level** parameter.

1. Kerberos. This option provides initial authentication only for the connection.
2. Kerberos with Packet Integrity. This option provides initial authentication and guarantees data packet integrity.
3. Kerberos with Encryption. This is the highest level of security and provides initial authentication, guaranteed data packet integrity, and, finally, encryption for all transmitted messages.

17.4.1 Kerberos Library

To use any of the Kerberos-based modes, the Web Gateway must be able to load the InterSystems Kerberos client library:

- Windows DLL: irisconnect.dll
- UNIX® Shared Object: irisconnect.so

Install the appropriate library in a location specified in the PATH environment variable for the operating system or at one of the following locations relative to the Web Gateway installation.

- . (that is, local to the Web Gateway)
- ./bin
- ../bin
- ../../bin

The Web Gateway attempts to load the library at the time it is first required. If successful, the following status message is written to the Web Gateway Event Log:

```
Web Gateway Initialization The IRISCONNECT library is loaded - Version: 5.3.0.175.0.
```

(This library is used for the optional Kerberos-based security between the Web Gateway and InterSystems IRIS.)

If the Web Gateway is unable to locate or link to the `IRISCONNECT` library, a suitable statement of failure and error message is written to the Web Gateway Event Log.

For Kerberized communications between the Web Gateway and InterSystems IRIS, the Web Gateway is the Kerberos client.

The procedure for configuring the Web Gateway to use Kerberos is in the Windows section.

17.4.1.1 Overriding the Library Path If You Use SSL/TLS

By default, the Web Gateway expects dependent security libraries (shared objects) to be installed in its home directory (that is, the directory with the Web Gateway binaries).

If you use SSL/TLS connectivity between the Web Gateway and InterSystems IRIS, these libraries include the IRISCONNECT library and SSL/TLS libraries (on UNIX®: libssl.so and libcrypto.so, on Windows: libcrypto-1_1-x64.dll and libssl-1_1-x64.dll).

When the Web Gateway and IRISCONNECT libraries, loaded in the web server's process space, load a copy of the SSL/TLS libraries, there is a conflict between different versions of the same libraries that were previously loaded by the hosting web server. To ensure that only one copy of the SSL/TLS libraries are loaded in the web server process space, the Web Gateway must instruct the IRISCONNECT library to source the SSL/TLS libraries from the same location as those used by the hosting web server.

The Web Gateway Management **Default Parameters** page provides the parameter **SSL/TLS Library Path** to allow you to use an alternative set of OpenSSL libraries. For example:

```
SSL/TLS Library Path = /usr/bin/
```

Important: It is possible to create an Apache installation that does not permit OpenSSL usage, and it is possible to configure Apache to disable OpenSSL. In this situation, the Web Gateway loads the libraries it was shipped with unless **SSL/TLS Library Path** is set to a different location.

If a library version mismatch occurs between Apache and the Web Gateway, TLS connections between the Web Gateway and an InterSystems IRIS instance can fail. A TLS error can occur when a connection is attempted, or the Web Gateway might crash with SIGSEGV when calling an OpenSSL function.

17.4.2 Windows

Kerberos key tables are not implemented for Windows. Therefore, authentication uses network credentials that are either obtained when the hosting service starts in a named account or from the Trusted Computing Base (TCB) when the hosting service runs in the System Logon Session (that is, as `LOCAL SYSTEM`).

Windows domain accounts use a permanent key derived from a password to acquire a Kerberos Ticket Granting Ticket (TGT) and service ticket for the local machine. The local machine must also have a permanent Kerberos key, shared with the Key Distribution Centre (KDC) component of the domain controller. That key can be used to acquire a TGT and service ticket to authenticate to another Kerberos principal such as InterSystems IRIS.

For practical purposes the Web Gateway, operating within the context of a Windows-based web server is operating through either the Network Service logon session or the System logon session. The account used must have `Log on as a batch job` rights assigned.

The built-in Network Service logon session has access to the machine's credentials and is designed for services that need network credentials to authenticate to other machines. However, the Network Service logon session is not always present. The System logon session can also be used for the purpose of authenticating the Web Gateway to InterSystems IRIS.

For IIS installations, and ISAPI extensions in particular, using the Network Service login session is the preferred means through which both databases (local and remote) and remote computers should be accessed.

17.4.2.1 Windows Web Gateway Configuration for Kerberos

- Set the **Service Principal Name** to that of the target InterSystems IRIS server that the Web Gateway is connecting to.
- Leave the **Username**, **Password**, and **Key Table** fields empty.
- The client principal name (or client username) is that of the Web Gateway host. This is the Kerberos name representing the Web Gateway hosts' network service session: *<computer_name>$*
- Assign this principal the necessary privileges in the InterSystems IRIS server to allow the Web Gateway's service to operate.

17.4.3 UNIX® Web Gateway Configuration for Kerberos

These operating systems support Kerberos Key Tables.

17.4.3.1 UNIX® Web Gateway Configuration for Kerberos

The Web Gateway configuration is conceptually more straightforward for these systems.

- Set the **Service Principal Name** to that of the target InterSystems IRIS server that the Web Gateway is connecting to.
- Enter the name of the key table file (including the full path) in the **Key Table** field.
- Set the **Username** field to the name of the appropriate key in the key table file.
- Leave the **Password** field empty.
- The client principal name (or client username) is that of the Web Gateway host. This is the name used to identify the key in the Kerberos Key Table. Assign this principal the necessary privileges in the InterSystems IRIS server to allow the service of the Web Gateway to operate.

17.5 SSL/TLS-Based Authentication and Data Protection

You can use the SSL/TLS protocol to secure communications between the Web Gateway and InterSystems IRIS.

In this mode, the SSL/TLS transport, as configured for this host, secures connections to InterSystems IRIS. The **SSL/TLS Configuration Name** field should be set to the appropriate value for the target server. The **Service Principal Name** and **Key Table** fields are not relevant and should be left empty.

For more information on creating SSL/TLS client configurations for InterSystems IRIS systems, see Configuring the Web Gateway to Connect to InterSystems IRIS Using TLS.

18

CGI Environment Variables Passed by the Web Gateway

CGI environment variables are derived both from the client’s HTTP request headers and from the environment in which the web server is operating. The Web Gateway transmits the common environment variables to InterSystems IRIS® with each and every request. If extra environment variables are required by the application, they must be explicitly requested in the Web Gateway configuration (via the **Extra CGI Environment Variables** setting in the **Application Access** section of the configuration). In the InterSystems IRIS Management Portal, navigate to **System Administration** > **Configuration** > **Web Gateway Management** and select **Application Access**.

The list of environment variables transmitted is shown in the table below together with a brief description of each. Further documentation can be obtained from standard web text books.

Environment Variable	Value
AUTH_PASSWORD	Value entered in the client’s authentication dialog. This variable is available only if Basic authentication is used.
AUTH_TYPE	Contains the authentication method that the server uses to validate users when they attempt to access a protected script.
CONTENT_TYPE	For requests which have attached information, such as HTTP POST and PUT, this is the content type of the data.
GATEWAY_INTERFACE	Revision of the CGI specification to which this server complies. Format: CGI/revision
HTTP_ACCEPT	Value of the Accept request header that contains a list of accepted formats (MIME types). For example: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, application/vnd.ms-excel. The values of the fields for the HTTP_ACCEPT variable are concatenated, and separated by a comma (,).
HTTP_ACCEPT_CHARSET	Comma-delimited list of the character encodings that the client accepts.
HTTP_ACCEPT_LANGUAGE	Contains a string describing the language to use for displaying content (such as en-us).
HTTP_AUTHORIZATION	Contains the Base-64 encoded username, password, scheme and realm sent by the client.
HTTP_COOKIE	Holds the contents of the client’s cookie(s).

Environment Variable	Value
HTTP_REFERER	Holds a string that contains the URL of the page that referred the request to the current page using an HTML <A> tag. Note that the URL is the one that the user typed into the browser address bar, which may not include the name of a default document. If the page is redirected, HTTP_REFERER is empty.
HTTP_SOAPACTION	SOAPAction HTTP request header field can be used to indicate the intent of the SOAP HTTP request. The value is a URI identifying the intent. SOAP places no restrictions on the format or specificity of the URI or that it is resolvable. An HTTP client MUST use this header field when issuing a SOAP HTTP Request.
HTTP_USER_AGENT	Browser the client is using to send the request. General format: software/version library/version.
HTTPS	Set to either `On` or `Off` (using word, not numerical value). Set to `on` if the script is being called through a secure server (that is, using SSL/TLS).
PATH_TRANSLATED	Translated version of PATH_INFO, in which any virtual-to-physical mapping is applied to the path.
REMOTE_ADDR	IP address of the remote host making the request.
REMOTE_HOST	Hostname making the request. If the server does not have this information, it should set REMOTE_ADDR and leave this parameter unset.
REMOTE_IDENT	If the HTTP server supports RFC 931 identification, then this variable is set to the remote username retrieved from the server.
REMOTE_USER	Name of the user as it is derived from the authorization header sent by the client
REQUEST_METHOD	Method with which the request was made. For HTTP, this is `GET`, `HEAD`, `POST`, and so on.
SERVER_NAME	The server's hostname, DNS alias, or IP address as it would appear in self-referencing URLs.
SERVER_PORT	Port number to which the request was sent. For example: 80
SERVER_PORT_SECURE	Set to either 0 or 1. If the request is being handled on the web server's secure port, then it is set to 1. Otherwise, it is set to 0.
SERVER_PROTOCOL	Name and revision of the information protocol that the request came in with. Format: protocol/revision
SERVER_SOFTWARE	Name and version of the web server software responding to the request. Format: name/version.

19

HTTP Response Headers Returned by the Web Gateway

Web applications in InterSystems IRIS® (including REST-based applications) usually assume the responsibility for formulating a full HTTP response header. For performance reasons, the Web Gateway by default streams the response headers, together with the following content, directly to the client via the web server. This mode of operation is known as the *non-parsed header* (NPH) approach. The Web Gateway does not grant the hosting web server any control over the response headers by passing them back through the dedicated API functions provided by the server. It is assumed that it is the client that needs to read and interpret the response header directives rather than the web server.

However, this assumption breaks down in cases where it necessary for the web server to interpret the response headers in order to invoke further web server-based functionality implied in the header directives generated by the CSP engine) For example, by invoking output filters to further process the response (compression and encryption utilities etc.). Such output filters are usually found not to work for CSP content returned according to the nonparsed header mode of operation.

A facility exists to instruct the Web Gateway to explicitly pass the response headers through the hosting web server instead of streaming them directly to the client.

To use this facility, set the following CSP header directive: `CSP-nph: false`

This directive must be set in the **OnPreHTTP()** method. For example:

```
<script language=objectscript method=OnPreHTTP arguments=""
returntype=%Boolean>
Do %response.SetHeader("CSP-nph", "false")
Quit 1 </script>
```

When set to `false`, (the default setting for the Web Gateway is `true`), the `CSP-nph` directive ensures that the hosting web server is properly notified as to the nature of the response through the response headers returned from the CSP engine. As a result, any further processing can be performed as necessary. This is parsed header mode.

When the Web Gateway is operating in parsed header mode, the hosting web server interprets the response headers and perhaps add header directives of its own. At the very least it adds a Server header to the response. For example:

```
Server: Apache/2.0.48 (Win32)
```

OR:

```
Server: Microsoft-IIS/5.1
```

Note that this facility only applies to the use of Web Gateway implementations that work directly to web server APIs. In other words: everything other than CGI.

If the Web Gateway CGI modules are used and this facility is required then you must configure the web server to use the non-NPH versions of the CSP CGI modules. For example, use `CSPcgi` instead of `nph-CSPcgi`. The `nph-` prefix used in the name of a CGI module is the standard way of informing the web server that it is not required to read and interpret the response headers returned by the module: in other words operate in non parsed header mode.

The essential difference between the parsed and non-parsed versions of these modules lies in the way the HTTP response status line is formulated. This is the first line in the header block.

For parsed headers, format the HTTP status line as follows:

`Status:` *<status_code>*

Example:

`Status: 200 OK`

For nonparsed headers, format the HTTP status line as follows:

`HTTP/1.1`*<status_code>*

Example:

`HTTP/1.1 200 OK`

The CGI modules supplied with the Web Gateway automatically handle these differences internally. The CSP engine always return a standard HTTP header block (2).

See also the Non-parsed Headers parameter in Adding an Application Path.

20

Compressing the Response to Requests for CSP Forms (GZIP/ZLIB)

Compressing the response generated by the CSP engine before dispatching it to the client is advantageous because it can dramatically reduce the network bandwidth required to transport the response to the client. From the client's perspective the performance of the application is improved. This is particularly true for clients accessing the application through mobile devices over slower telecommunications networks. There is, of course, a cost in terms of the web server host's CPU time that's required to actually compress the data but this is a small price to pay for the advantages.

The advantage of serving compressed response data is particularly marked for CSP pages for which large volumes of response data are generated.

There are two methods for implementing GZIP in a web server environment.

- Using the Web Gateway's own interface to the GZIP library described here.
- Using a GZIP output filter as an add-on to the hosting web server.

Most web servers offer add-on facilities for compressing data. Windows/IIS offers a `gzip` filter (implemented as an ISAPI filter). The Apache Group offer a compression filter implemented as an add-on module (mod_deflate.c – which, rather confusingly, implements `gzip` compression not `deflate`). There is also a third-party module for Apache called mod_gzip.c. There are a number of third-party GZIP products available as add-ons for most web servers.

The advantages of implementing a compression solution directly in the Web Gateway are as follows:

- Ease of setup and configuration.
- Greater flexibility in controlling which CSP files are to be compressed.
- Compression tends to work better if the data is submitted to the compressor functions in large buffers. The Web Gateway receives the response content from InterSystems IRIS in fairly large chunks; therefore the performance of the compression and the degree of compression achieved are good.

It has been discovered that if Chunked Transfer Encoding is enabled at the Web Gateway level and if the Apache mod_deflate output filter is enabled for the same resources, then browsers are occasionally unable to display the response content.

The Web Gateway makes use of the freely available GZIP (or `zlib`) library for implementing data compression. The compression algorithm used is described in RFCs (Request for Comments) 1950 to 1952.

20.1 The GZIP/ZLIB Library

The GZIP/ZLIB library was developed by Jean-loup Gailly and Mark Adler (Copyright (C) 1995-2009). A pre-built version of this library is provided with InterSystems IRIS distributions on Windows. On UNIX systems, the Web Gateway uses the OS-supplied build of ZLIB.

The Web Gateway dynamically links to the ZLIB library when response compression is requested for the first time. Thereafter the ZLIB library remains loaded until the Web Gateway is closed down.

If the Web Gateway is able to load the ZLIB library on demand and identify all the required functions, an initialization message is written to the Event Log in the following format (with *x.x.xx* standing in for the version of the library on your system):

Web Gateway Initialization
The ZLIB library is loaded - Version *x.x.xx*.
(This library is used for the optional GZIP compression facility)

If the Web Gateway cannot find or link to the ZLIB library, it operates as before and pages are returned without being compressed. A statement of failure is written to the Event Log.

20.2 Using the GZIP/ZLIB Library

The Web Gateway implements two modes of operation (1 and 2) for compressing the response data using the ZLIB library:

1. In this mode, the Web Gateway streams all data received from InterSystems IRIS into the compressor. When all the data has been processed, the compressor streams the compressed data back to the Web Gateway at which point it is forwarded on to the client.

   This mode offers the best possible compression at the expense of slightly higher latency. Of course, the latency is more pronounced for larger forms.

2. In this mode, the Web Gateway streams all data received from InterSystems IRIS into the compressor. On each and every call, the compressor makes as much compressed data as it can available to the Web Gateway at which point it is forwarded on to the client.

   This mode offers the lowest possible latency at the expense of slightly reduced level of compression. Of course, the reduction in the degree of compression achieved is more pronounced for larger forms. Generally speaking, mode 2 is more appropriate for web applications where it is usually not possible to know, in advance, how much data a response contains.

If (and only if) the Web Gateway is able to successfully compress the data stream returned from InterSystems IRIS, it modifies the HTTP response headers to include the appropriate Content-Encoding directive. For example:

```
HTTP/1.1 200 OK
Content-Type: text/html; charset=ISO-8859-1
Set-Cookie: CSPSESSIONID=000000000002119qMwh3003228403243; path=/csp/samples/;
Cache-Control: no-cache
Connection: Close
Date: <date and time>
Expires: <date and ttime>
Pragma: no-cache
Content-Encoding: gzip
```

Before attempting to compress response data, the Web Gateway always checks the value of the `Accept-Encoding` HTTP request header (the HTTP_ACCEPT_ENCODING CGI environment variable). The Web Gateway only compresses a response if the client has indicated that it is capable of dealing with compressed content.

For example:

```
Accept-Encoding: gzip, deflate
```

There are several methods for specifying that a CSP response should be compressed. These are discussed in the following sections.

20.3 Specifying Compression for Individual Pages

Within web applications, the `%response` object contains a property called `GzipOutput`. If this property is set to true (or the mode required) the Web Gateway attempts to compress the response.

```
<script language=objectscript method=OnPreHTTP arguments=""
         returntype=%Boolean>
         Set %response.GzipOutput = 2
         Quit 1
</script>
```

Compression can also be specified on a per-page basis by adding the **CSP-gzip** directive to the HTTP response headers. This must, of course, be done in the `OnPreHTTP` method. For example:

```
<script language=objectscript method=OnPreHTTP arguments=""
        returntype=%Boolean>
        Do %response.SetHeader("CSP-gzip", "2")
        Quit 1
</script>
```

The `CSP-gzip` header directive should be set to the compression mode required (1 or 2).

20.4 Specifying Compression for All Pages within an Application Path

Compression can be specified on a per-application path basis. This, incidentally, is the most common method for indicating that compression should be used when using a web server output filter (such as `mod_deflate`).

Use the following configuration parameters in the Web Gateway Application Access section:

Item	Function
GZIP Compression	If Enabled, all CSP output for that path is compressed. Default is Enabled.
GZIP Minimum File Size	Controls the minimum response size in bytes for which compression is activated. If left empty, then all responses for which GZIP is enabled are compressed.
GZIP Exclude File Types	List of file types to be excluded from GZIP compression. Files can be listed by MIME type (such as image/jpeg) or by common extension (such as jpeg). By default, these common (natively compressed) image files are excluded: `GZIP Exclude File Types: jpeg gif ico png gz zip mp3 mp4 tiff`. Separate additional types or extensions with a space.

20.5 Monitoring

Log level V3 instructs the Web Gateway to record the degree of compression achieved for all responses that were successfully compressed. The size of the compressed data and the original uncompressed data stream is recorded.

For example:

```
GZIP Compression for /csp/samples/inspector.csp
GZIP Mode=1; Uncompressed Content Size=19042; Compressed Content Size=2499 (13 percent)
```

21

Implementing HTTP Authentication for Web Applications

The Apache modules (mod_csp*.so/dll and CSPa*[Sys].so/dll) to allow HTTP authentication to be controlled by InterSystems IRIS®.

HTTP authentication of web requests is normally carried out between the web server and client (browser). Because of this, it is not usually possible to implement HTTP authentication in custom request handlers hosted by the web server – such as CGI programs and web server API-based request handlers. Of course, such extensions can issue a `401 Authorization Required` response header and, in response to this, the browser displays the HTTP login dialog. However, in the subsequent request, the web server intercepts the user's login details and attempts to authenticate the user using its own built-in functionality. The username and password are not, at least in the first instance, passed along to the request handling extension until the web server has authenticated the user on its own terms.

This scheme presents a problem for users of third-party development technologies (such as CSP) who wish to perform HTTP authentication locally (and programmatically) within their technology of choice.

The feature described here overcomes these technical difficulties and allows users to perform HTTP authentication in the InterSystems IRIS environment for Apache-hosted web applications. Users of Apache can choose between the three approaches described in the following sections.

21.1 Standard HTTP authentication in Apache (mod_auth)

This method is the standard mechanism provided by Apache (through the `mod_auth` module) and does not involve the Web Gateway. It is mentioned here for the sake of completeness.

As an example, the basic parameters required for protecting the CSP samples using Apache-based authentication are shown in the following configuration block (httpd.conf):

```
<Location "/csp/samples/">
    AuthType Basic
    AuthName "CSP samples"
    AuthUserFile conf/csp.pwd
    require valid-user
</Location>
```

Where:

AuthType is the type of authorization required (usually `Basic`).

AuthName is the realm.

AuthUserFile is the file (relative to the web server root) holding usernames and their associated passwords (in encrypted form). This file is created and maintained by the Apache `htpasswd` utility.

The *require* parameter lists the users who may access the protected resource (the CSP samples in this case). The *valid-user* argument indicates that the user must be defined in the username/password file (as declared in AuthUserFile).

Apache provides for users to be grouped together in user 'groups' – see the *AuthGroupFile* directive for further details:

```
https://httpd.apache.org/docs/2.4/mod/mod_authz_groupfile.html#authgroupfile
```

21.2 Authenticating in CSP at the Same Time as the Request is Processed.

This is the preferred (and best performing) method for implementing HTTP authentication in web applications.

The basic parameters required for protecting the CSP samples using CSP-based authentication are shown in the following Apache configuration block (httpd.conf):

```
<Location "/csp/samples/">
    AuthType Basic
    AuthName "CSP samples"
    require valid-user
    AuthCSPEnable On
    AuthBasicAuthoritative Off
</Location>
```

The parameters *AuthType*, *AuthName* and *require* are the standard Apache parameters used for triggering authentication.

The additional *AuthCSPEnable* parameter instructs the CSP module to bypass the authentication checks that would otherwise be performed by Apache (in mod_auth) and pass the user's name and password, along with the original web request, to InterSystems IRIS for authentication. The web application must check the user using the following CGI environment variables:

- AUTH_TYPE: This is Basic.
- REMOTE_USER: The user's name.
- AUTH_PASSWORD: The user's password (as plain text).

If the user can be successfully authenticated based on the values held in these parameters then the application should continue and process the request (i.e. return the requested CSP resource). If not, it should return a HTTP `401 Authorization Required` response which, at the very least, should be something like:

```
HTTP/1.1 401 Authorization Required
WWW-Authenticate: Basic realm="CSP samples"
Content-Type: text/html
Connection: close
<html>
<head><title>401 Authorization Required</title>
</head><body> <h1>Authorization Required</h1>
<p> The server could not verify that you are authorized
to access the application. Check your username and password.
</p>
<hr>
</body>
</html>
```

On receiving this message the browser redisplays the login dialog unless the user has used-up all his/her login attempts (usually 3) in which case the message following the header is displayed instead.

Users can implement this method of authentication by modifying the login page. If a request comes in and the user does not have the necessary privileges to run the application then the login page is called, the processing for which can extract the authentication information from the request (such as AUTH_TYPE, REMOTE_USER and AUTH_PASSWORD). If these parameters are correct, the login script can then redirect control to the application page that was originally requested. It should not be necessary to repeat the authentication procedure for all public pages provided the InterSystems security control layer is deployed.

21.3 Authenticating in CSP before the Request is Processed.

This is an alternative method for implementing HTTP authentication in InterSystems IRIS. It is intended primarily for cases where performing authentication at request-processing time in the web application would be awkward or time consuming.

In this method, the user is authenticated by calling a dedicated authentication class. The Web Gateway performs this check before dispatching the original request to InterSystems IRIS. When the user's details have been successfully checked by the authentication class, the web application need not perform any further any further checking.

Of course, this method bears the overhead of processing two requests (to InterSystems IRIS) per web request: one for authentication and one for actually dealing with the request for the CSP resource.

The basic parameters required for implementing this method of authentication are shown in the following Apache configuration block (httpd.conf):

```
<Location "/csp/samples/">
    AuthType Basic
    AuthName "CSP samples"
    require valid-user
    AuthCSPEnable On
    AuthCSPClass /csp/samples/%CSP.HTTPAuthentication.cls
    AuthBasicAuthoritative Off
</Location>
```

The parameters *AuthType*, *AuthName*, *require* and *AuthCSPEnable* are the same as for method (2).

The additional *AuthCSPClass* parameter defines a class that performs user authentication. The class must extend %CSP.Page and, using the appropriate CGI environment variables, should check the user's login details and return either a `200 OK` response header if the operation is successful or a `401 Authorization Required` response header if not.

A simple authentication class in which user login details are checked against records held in the %Users file is shown below:

```
Class %CSP.HTTPAuthentication Extends %CSP.Page
{
        ClassMethod OnPreHTTP() As %Boolean
        {
                Set %response.ContentType = "text/html"
                Set %session.Preserve = 0
                Quit 1
        }
        ClassMethod OnPage() As %Status
        {
                Set crlf=$Char(13,10)
                Set type=%request.GetCgiEnv("AUTH_TYPE", "")
                Set user=%request.GetCgiEnv("REMOTE_USER", "")
                Set pwd=%request.GetCgiEnv("AUTH_PASSWORD", "")
                Set httpauth=%request.GetCgiEnv("HTTP_AUTHORIZATION", "")
                If httpauth'="" {
      Set type=$Piece(httpauth," ",1)
      Set user=$system.Encryption.Base64Decode($Piece(httpauth," ",2))
      Set pwd=$Piece(user,":",2)
      Set user=$Piece(user,":",1)
                }
                Set auth=0
                If $ZConvert(type,"L")'="basic" Set auth=1
                If auth=0,user'="",$Get(^%Users(user))=pwd Set auth=1
                If auth=1 {
                        Write "HTTP/1.1 200 OK"_crlf
                        Write "Content-Type: text/html"_crlf
                        Write "Content-Length: 0"_crlf
                        Write "Connection: close"_crlf_crlf
                }
                Else {
                        Write "HTTP/1.1 401 Authorization Required"_crlf
                        Write "WWW-Authenticate: Basic realm=""CSP samples"""_crlf
                        Write "Content-Type: text/html"_crlf
                        Write "Content-Length: 0"_crlf
                        Write "Connection: close"_crlf_crlf
                }
                Quit $$$OK
        }
        ClassMethod OnHTTPHeader(ByRef OutputBody As %Boolean) As %Status
        {
                Quit $$$OK
        }
}
```

For methods (1) and (3) a custom error page can be specified for login failure by using the Apache `ErrorDocument` directive. For example:

```
ErrorDocument /error/my_authentication_error.html
```

Of course, for method (2) the text of the error message is controlled by the web application.

22

Mirrored Configurations, Failover, and Load Balancing

22.1 Load Balancing and Failover Between Multiple Web Servers

In most environments, multiple web servers are used to balance load and provide high availability at the web server layer. A load balancer is typically required to direct user connections to participating web servers. For best performance and resilience, it is recommended that a hardware-based solution is used. A Load Balancing system such as Cisco ACE 4710 or the F5 BigIP LTM appliance is placed in front of a set of web servers. In this configuration, if there are also multiple InterSystems IRIS server instances, such as in a distributed cache cluster, each web server (and by implication, Web Gateway instance) should be configured to connect to a specific InterSystems IRIS® server instance.

Software based load-balancing and failover systems, though not as robust as hardware based solutions, are much less costly to deploy. Examples of software based solutions include HAProxy and the Apache Group's mod_proxy_balancer. For more information, see the HAProxy site www.haproxy.org

Important: Sticky sessions should always be enabled for web applications. It is essential that each user session 'sticks' to the same back-end InterSystems IRIS server for the lifetime of the session – unless, of course, a failover event occurs.

Although the above approach is the primary recommendation, the Web Gateway provides a basic (software-based) system for implementing load balancing and failover between multiple InterSystems IRIS servers (that is, the CSP servers for multiple InterSystems IRIS instances). This facility is described in the following section.

22.2 Load Balancing and Failover Between Multiple InterSystems IRIS Server Instances

In configurations with multiple (equivalent) InterSystems IRIS server instances, such as in a distributed cache cluster, the Web Gateway provides a basic (software-based) facility for implementing load balancing and failover between those InterSystems IRIS instances for web applications. An external solution like those described previously is the primary recommendation, however.

The failover mechanism provided by the Web Gateway is not necessary to implement failover between multiple InterSystems IRIS database servers in a typical High Availability configuration, such as failover clustering or InterSystems IRIS mirroring. Those technologies provide Virtual IP based failover and the Web Gateway can be configured to connect to that IP address.

The remainder of this section describes the load balancing and failover capabilities provided by the Web Gateway.

Web Gateway load balancing and failover is configured in the **Application Access** section of the Web Gateway management pages. See Configuring Application Access.

Navigate to **System Administration** > **Configuration** > **Web Gateway Management** and select **Application Access**. A list of InterSystems IRIS servers may be defined for an application (path). Use the options listed under the **Use Alternative Servers For** parameter to select the purpose for which they are to be used. The following options are available:

- **Fail-Over**
- **Load-Balancing and Fail-Over**

The default course of action is to use the first InterSystems IRIS server defined in the list. Following this default server is the list of alternative InterSystems IRIS servers, each designated as **Server #** where # is the server number.

The configuration screen initially shows only three empty server slots, but additional slots appear that enable you to define any number of alternative servers. Each server can be marked as **Enabled** or **Disabled**. The default setting is **Enabled**.

Load balancing is implemented in a round robin fashion. Each new user session is connected to the next available alternative server. Once a user session is established on a server, the Web Gateway maintains the session on that server unless it becomes unavailable, in which case the session is failed-over to the next available server in the list. State-aware sessions (preserve mode = 1) cannot be failed-over under any circumstances and, consequently, the session is closed if the hosting server becomes unavailable.

If a CSP server does not respond with the span of time specified by the **Server Response Timeout**, the Web Gateway marks the server as offline and does not use it for load balancing. However, if Web Gateway Registry functions are enabled (they are enabled by default), then the Web Gateway periodically attempts to reconnect to CSP servers which are offline. If the connection to the CSP server succeeds, the Web Gateway marks it online and uses it for load balancing again, automatically.

22.3 Mirrored Configurations

With mirrored InterSystems IRIS configurations, a database is duplicated (or *mirrored*) between participating *mirror members*. An InterSystems IRIS *mirror set* configuration represents the set of participating mirror members for an installation. For a complete description of InterSystems IRIS mirroring, see Mirroring in the *High Availability Guide*.

If Mirror Virtual IP (or an equivalent technology) is used to provide network redirection to the primary member, then configure the Web Gateway to connect to that address. No further action is required. The Virtual IP address is always mapped to the mirror primary.

For configurations where the Mirror Virtual IP cannot be used (or does not operate in certain disaster scenarios), it is possible to configure the Web Gateway to be *mirror-aware*. When the Web Gateway is mirror-aware, it assumes responsibility for determining which member is primary. To make a Web Gateway configuration mirror-aware, in the Web Gateway's **Server Access** section, select **Configuration is Mirror Aware** and provide the address of one of the mirror members.

Note: There are situations where it is not appropriate for a Web Gateway configuration to be mirror-aware. For example, a Web Gateway configuration supporting the Management Portal should never be configured to be mirror-aware as the portal must always connect to a specific InterSystems IRIS server regardless of its mirror status.

If a mirror-aware Web Gateway configuration connects to an InterSystems IRIS server that is not a mirror member then the connection fails and the affected client receives a `Server Availability` error.

The Web Gateway obtains – from the Member that it first connects to – a list of failover members and disaster recovery (DR) members. The Web Gateway persists this list in its local configuration file (CSPRT.ini). If the Web Gateway subsequently cannot connect to the member defined in its configuration then it uses the list previously recorded locally to enable it to identify and connect to alternative members.

The Web Gateway cycles through the members list until it finds the primary. If it cannot find the primary, the Web Gateway defaults to the server defined in the Gateway configuration.

- The Web Gateway repeatedly cycles through the list until it finds a member defined as primary.
- To avoid the negative performance impact of a tight looping structure, the Web Gateway pauses after each cycle for a number of seconds equal to the number of tries.
- For a given HTTP request, the Web Gateway spends no more time attempting to find the primary than that defined in the **Server Response Timeout** parameter.
- When searching for the primary, the Web Gateway always connects to failover members first. It only tries async members if it cannot find the primary amongst the failover members. An async member only becomes primary if you manually designate it as primary.

Mirror members appear in the Web Gateway System Status form when the first connection is made. Mirror members are shown named as the current configuration name (as defined under the Web Gateway’s **Server Access** section) with the mirror set name, mirror, and mirror member name shown as a tooltip.

The columns, **Mirror Name** and **Mirror Status** appear in the 'InterSystems IRIS Servers' table. The name of the mirror set and mirror member are shown in the **Mirror Name** column. The current member status is shown in the **Mirror Status** column: the **Member Type** (`Failover` or `Async`) is shown and the primary member is labelled as `Primary`.

23

Process Affinity and State-Aware Mode (Preserve Mode 1)

The architecture of the web is *stateless*. In order to get the best out of web architecture in terms of performance, maintainability and scalability, web applications should embrace the stateless paradigm.

By default, web applications operate in a stateless environment with respect to the hosting InterSystems IRIS® server. The Web Gateway maintains a pool of connections to InterSystems IRIS and distributes the workload amongst them and increases, within configured limits, (or decreases) the size of the connection pool. Each connection is associated with a single InterSystems IRIS process (as identified by the *$Job* variable).

For a normal web application operating in stateless mode, consider the choice of backend InterSystems IRIS process used to serve each request for a client session to be random. The Web Gateway chooses whichever connection/process happens to be free.

However, in the interests of efficiency, the Web Gateway does implement a form of InterSystems IRIS *process affinity*. In other words, it attempts, where possible, to route a request for a session to the same InterSystems IRIS process that was used to serve the previous request for that session.

In addition to a measure of process affinity based on session ID, the Web Gateway also attempts to implement process affinity based on namespace. The Web Gateway keeps track of the namespace to which each connection is pointing and delivers, where possible, requests to a connection that is already pointing at the namespace required to process the request. This helps in avoiding the overhead incurred in moving resources between different namespaces on receiving each web request.

In terms of precedence, session affinity always overrides all other considerations in the selection of a connection. If an incoming request cannot be assigned to the same connection previously used to serve the client session, namespace affinity is used instead to influence the final choice.

CSP includes a mode whereby the Web Gateway routes all requests for a session to a reserved (or private) InterSystems IRIS connection/process. This mode of operation provides a *state-aware* environment with respect to the relationship between web sessions and their corresponding InterSystems IRIS processes.

State-aware mode is implemented as CSP Preserve Mode 1

The original motivation for the provision of a state-aware mode of operation (preserve mode 1) was to make it relatively easy to migrate legacy application code from a fixed client-server environment (e.g. terminal applications) to the web. Support for transactions that spanned several HTTP requests was also a consideration in its introduction. However, the limitations outlined in the following paragraphs should be borne in mind when creating state-aware applications.

State-aware applications do not scale as well as their stateless counterparts and it is therefore recommended that new applications (and modifications to existing ones) be designed to be stateless as far as is practically possible. It is recommended that state-aware mode, if used at all, should be applied sparingly in predominantly stateless applications.

Writing complete applications to operate in state-aware mode is not recommended. Apart from the scalability issues that arise as a consequence of the need to reserve an InterSystems IRIS process for each and every session, state-aware applications are unable to take full advantage of modern load balancing and failover solutions because of the very specific requirements for routing requests. Also, state-aware applications are not as fault-tolerant as their stateless counterparts. For example, the recycling of a web server worker process can happen transparently beneath a stateless application but results in all associated state-aware sessions closing. Of course, you can avoid the latter restriction by using the Web Gateway's NSD component to separate the management of the Web Gateway process pool from the hosting web server.

Creating a successful state-aware application (or state-aware sections within a predominantly stateless application) requires a certain amount of discipline.

Since all requests for a session must be processed by the same InterSystems IRIS process, a queue must be maintained to serialize access to the private InterSystems IRIS process for cases where the client simultaneously dispatches several requests. The original HTTP v1.1 standard mandated that a client should simultaneously open no more than 2 connections to each server (RFC2616). However, this limit is configurable and, indeed, the latest generation of web browsers support, by default, up to 8 connections to each server. Needless to say, an increase in the maximum number of connections to each server can have a profound effect on state-aware web applications: an application can expect up to 8 requests to be fired concurrently and subsequently held in the queue responsible for controlling access to the single private InterSystems IRIS process.

Another potential pitfall in state-aware mode is the effect of the Server Response Timeout operating between the Web Gateway and InterSystems IRIS. When the Web Gateway does not receive a response within the prescribed time limit imposed by the response timeout it has no option but to close the connection with the consequential loss of the state-aware session.

Finally, the effect of client interrupts can cause problems with applications operating in state-aware mode. When a client interrupts a request at (and beyond) the point at which InterSystems IRIS is generating a response, the Web Gateway attempts to absorb the (now unwanted) response payload in order to retain the connection. If it is unable to do this in a timely fashion it, again, has no option but to interrupt whatever the InterSystems IRIS process is doing by closing the connection and the session is lost. Bear in mind that while the Web Gateway is attempting to absorb the payload for an interrupted request, further requests for the same session may be arriving and placed in the queue.

In summary, follow the following design goals when creating state-aware applications.

- As far as possible avoid (or use sparingly) client constructs that generate many simultaneous requests (for example: HTML Frameset documents).
- Ensure that responses are generated quickly. This reduces the scope for issues related to timeout and/or client interrupt events. It also relieves pressure on the session queue. If a task in InterSystems IRIS potentially requires an extended time to complete, then consider performing it in another process so that the primary private process can quickly return a response to the Web Gateway (and client).

23.1 Launching State-Aware Mode

Mark a session as *state-aware* by setting the preserve mode as follows:

```
Set %session.Preserve = 1
```

It is recommended that a session be marked as state-aware in the form's OnPreHTTP method:

```
<script language=objectscript method=OnPreHTTP arguments="" returntype=%Boolean>
Set %session.Preserve = 1
Quit 1
</script>
```

Issuing the instruction here means that the CSP engine can mark the session cookie (or token) as state-aware before formulating and dispatching the HTTP response headers to the Web Gateway.

Sessions can be marked as state-aware after the **OnPreHTTP** method has fired but in this case the session cookie/token has already been formulated. The CSP engine passes the `preserve=1` instruction to the Web Gateway in the response footers (dispatched after the response payload) and the Web Gateway marks the connection as `private` and caches the instruction against the session ID so that it can recognize the unmodified session token as state-aware when subsequent requests arrive.

If the session is marked as state-aware in the **OnPreHTTP** method, the Web Gateway has no need to cache the transition against the session since the information is carried in the session cookie/token which effectively resides on the client.

23.2 Maintaining State-Aware Mode and Responding to Errors

Once a session is marked as state-aware and the Web Gateway has acknowledged the state-transition and marked the connection as *private*, the session transparently operates in state-aware mode until one of the following events occurs:

- The application transitions back to a stateless mode of operation.
- The application programmatically ends the session or the session times out.
- The private connection closes prematurely as a result of some error condition.

If the private connection hosting a state-aware application is prematurely closed (perhaps as a result of an error condition), the Web Gateway routes the request to a free stateless connection in the pool and InterSystems IRIS error number 5974 is returned:

```
CSP error occurred
Error: The persistent session is no longer available because the server process does not exist
ErrorNo: 5974
CSP Page: /csp/samples/loop.csp
Namespace: %SYS
Class: <Unknown>
```

At this point, the request is operating in stateless mode and it is the application’s responsibility to respond to this error: for example, by directing the user back to the login form for the application.

When operating in state-aware mode, the value of %session.NewSession should be checked in every page. Alternatively, the application should check the validity of user specific authentication data stored in %session.Data when the user was first authorized to access the application. These checks are important for security reasons and to ensure that the user session is still securely locked-in to a state-aware mode of operation. An error condition is not automatically raised under these circumstances because it is possible that the session had already (and legitimately) transitioned out of state-aware mode. For example, consider the situation where an incoming session token is still marked as state-aware but the application had already transitioned to stateless mode – this situation arising as a result of a session token being embedded in a form (as CSPCHD) that was served before the transition was made.

Finally bear in mind that when a session is terminated (for example, after it has timed out) the CSP engine deletes all operational data associated with the session, after which point any further incoming requests for that session are treated as though they are for a new session.

The embedded security mechanisms provided by InterSystems IRIS for web applications offer protection against the eventualities outlined above. Users are automatically directed to the login form in all cases where a loss of continuity within a state-aware application occurs (with respect to InterSystems IRIS process).

23.3 Terminating State-Aware Mode

An application can revert back to a *stateless* mode of operation by setting the preserve mode as follows:

```
Set %session.Preserve = 0
```

It is recommended that this code be executed in the form’s OnPreHTTP method:

```
<script language=objectscript method=OnPreHTTP arguments="" returntype=%Boolean>
    Set %session.Preserve = 0
    Quit 1
</script>
```

Issuing the instruction here means that the CSP engine can mark the session cookie (or token) as stateless before formulating and dispatching the HTTP response headers to the Web Gateway.

A session can be immediately terminated as follows:

```
Set %session.EndSession = 1
```

When you set this property, the session terminates immediately after serving the current request.

You can set a session to timeout as follows:

```
Set %session.AppTimeout = 900
```

The session times out and terminates after the prescribed number of seconds of inactivity. The default is 900 seconds (15 minutes).

24

Web Gateway Registry in InterSystems IRIS

The InterSystems Web Gateway Registry registers each connected Web Gateway installation with InterSystems IRIS® and provides the infrastructure to allow InterSystems IRIS code to interact with those installations (to clear caches and so on). Such programmatically controlled interactions may include reading and modifying the Web Gateway's runtime configuration and collecting system status and log information. The relevant classes are as follows:

```
%CSP.Mgr.GatewayRegistry (The Gateway Registry)
%CSP.Mgr.GatewayMgr  (A Connected Gateway)
```

The following code lists all connected (i.e. active) Web Gateway installations and writes the web server IP address, port and Web Gateway build number to the console window.

```
Set reqistry = $system.CSP.GetGatewayRegistry()
Set gateways = reqistry.GetGatewayMgrs()
For no=1:1:gateways.Count() {
     Set gateway = gateways.GetAt(no)
     Write !,no, " : "
     Write gateway.IPAddress,":",gateway.Port," ",gateway.Version
}
```

When InterSystems IRIS is first started this list is empty. As Administrator and User activity increases, expect at least two entries to appear: one for the private web server serving the Management Portal and at least one for external web servers supporting applications.

You can find further documentation associated with the classes listed above. Some code examples follow to illustrate common tasks.

List Default Parameters

```
Kill defaults
Do gateway.GetDefaultParams(.defaults)
ZWrite defaults
```

Update Default Parameter(s)

```
Kill newpars
Set newpars("Server_Response_Timeout")=30
Do gateway.SetDefaultParams(.newpars)
```

List Servers

```
Set status = gateway.GetServers(.servers)
For no=1:1:$ListLength(servers) {
     Set server = $List(servers,no)
     Write !,no, " : ",server
}
```

List Server Parameters

```
Kill serverpars
Do gateway.GetServerParams("LOCAL",.serverpars)
ZWrite serverpars
```

Update Server Parameter(s)

```
Kill newpars
Set newpars("Maximum_Server_Connections")=250
Do gateway.SetServerParams("LOCAL",.newpars)
```

List Application Paths

```
Set status = gateway.GetApplicationPaths(.paths)
For no=1:1:$ListLength(paths) {
     Set path = $List(paths,no)
     Write !,no, " : ",path
}
```

List Application Parameters

```
Kill pathpars
Do gateway.GetApplicationParams("/csp",.pathpars)
ZWrite pathpars
```

Update Application Parameter(s)

```
Kill newpars
Set newpars("GZIP_Compression")="Enabled"
```

Clear Gateway cache

```
Do gateway.ClearCache("*")
```

24.1 Forcing the Web Gateway to Reload Its Configuration

There are occasions when the Web Gateway’s configuration is modified by external agents (i.e. agents other than the Web Gateway’s own Systems Management Suite).

There are two methods for interactively instructing the Web Gateway to reload its configuration, and in a way that doesn’t require a complete restart.

24.1.1 Using the InterSystems IRIS Web Gateway Registry

The following Registry Method is provided:

```
Set status = %CSP.Mgr.GatewayMgr.ActivateCSPIni()
```

When successfully called, the Web Gateway reads its configuration file and activates all changes made.

24.1.2 Using Scripts External to InterSystems IRIS

Scripts should add the following line (case-sensitive) to the SYSTEM section of the modified Web Gateway configuration file:

```
[SYSTEM]
RELOAD=1
```

The Web Gateway caretaker daemon checks the *RELOAD* flag approximately every minute and, if correctly set, reloads and reactivates its configuration and removes the flag from the file. The following message is written to the Event Log after a successful reload operation:

```
Gateway Management
Gateway Configuration Reloaded and Reactivated
```

A

Using the NSD (Windows)

This page describes how to use the Network Service Daemon (NSD) with the InterSystems Web Gateway on Microsoft Windows. This is not the typical installation but is appropriate in some cases.

A.1 When to Use the NSD

There are three situations in which you might choose to use the NSD to separate the Web Gateway from the web server so that you can manage the Web Gateway independently of the web server:

- If your web server distributes its load over multiple server processes, an instance of the Web Gateway is then attached to each web server process.
- If you have a very large web server installation for which CSP is only a small part; for example, a web server that serves php, static content, .NET, and .ASP applications, as well as web applications.
- If you are using the Nginx web server.

A.2 NSD Module Install Locations

If you use the NSD Module in Microsoft Windows, you install the following two utilities:

- `CSPnsd.exe`
- `CSPnsdSv.exe`

On an IIS installation, these are installed in this location:

`C:\Inetpub\CSPGateway\nsd`

On an Apache installation, these are installed in this location:

`C:\Program Files\Apache Group\Apache\WebGateway\nsd`

Run the NSD from within its home directory, `C:\Inetpub\CSPGateway\nsd`. The configuration and log files are written in this directory for NSD-based connectivity options.

A.3 Operating the NSD

Use the following procedure to start the NSD.

1. Change to the NSD home directory, such as:

   C:\Inetpub\CSPGateway\nsd

2. Start the NSD with:

   ```
   CSPnsd
   ```

   The NSD starts as a Windows service (`CSPnsdSv.exe`). Once registered as a service, you can manage the NSD entirely through the Windows Service Manager.

3. Close down the NSD, by issuing the following command:

   ```
   CSPnsd -stop
   ```

Alternatively, you can enter:

```
CSPnsd
```

This shows the status of the NSD's Windows Service and allows you to perform one of the following actions:

- Stop the NSD service if it is running.
- Continue the NSD service if it is paused.
- Remove the NSD service from the services database.

Alternatively, you can use the Windows Service Manager to manage the NSD. The NSD can be identified in the Service Manager by the description:

```
Cache Server Pages – Network Service Daemon
```

All errors are reported in the Web Gateway Event Log.

Other Startup Options

1. Display help information.

   ```
   CSPnsd -h
   ```

2. Run the NSD interactively in a command window as opposed to as a Windows service. You must use this mode of operation if you are running multiple instances of the NSD.

   ```
   CSPnsd -v
   ```

A.3.1 Starting NSD on Alternative TCP Port

By default, the NSD listens for incoming requests on TCP port 7038. You can override this by starting the service as follows:

`CSPnsd –v` [*port_no*]

Or:

`CSPnsd –v -p`[*port_no*]

- where *port_no* is the TCP port number of your choice.

On startup, the NSD creates the CSPnsd.ini file, which typically contains the following lines:

```
[SYSTEM]
Ip_Address=127.0.0.1
TCP_Port=7038
```

In this context, the clients are the Web Gateway modules contained within, or dynamically linked to, the web server and/or the CSP CGI modules invoked by the server. It is, therefore, essential that this file is not deleted or moved. It is also important that the web server processes can read this file. Set the privileges accordingly, bearing in mind the Windows user under which your web server is operating. The NSD clients attempt to find this file in a location contained within the Windows PATH variable (for example: C:\Windows). Consequently, the CSPnsd.ini file must be moved to this location before starting the web server.

It is inappropriate to store the NSD port number in the CSPnsd.ini file for the scenario in which multiple instances of the NSD are running. For Apache servers, there is a much better mechanism for communicating the TCP port number of the NSD to its clients. Specifically, set the following environment variables in the Apache configuration to indicate the address and port of the target NSD installation. The values specified in these environment variables take precedence over any values found in the CSPnsd.ini file:

- `CSP_NSD_NAME` — This is the IP address of the NSD. Only use this parameter if the NSD is operating on a remote computer.
- `CSP_NSD_PORT` — This is the TCP port of the NSD.

A.3.1.1 Example 1: Two Apache Virtual Hosts

Distribute the load for two Apache virtual hosts (say, 123.123.1.1 and 123.123.1.2) between two independent NSD installations (listening on TCP port 7038 and 7039).

Add the following directives to the Apache configuration (httpd.conf):

```
<VirtualHost 123.123.1.1>
    ServerName 123.123.1.1
    SetEnv CSP_NSD_PORT 7038
</VirtualHost>
<VirtualHost 123.123.1.2>
    ServerName 123.123.1.2
    SetEnv CSP_NSD_PORT 7039
</VirtualHost>
```

A.3.1.2 Example 2: Two Web Applications

Distribute the load for two web applications (say, /csp1 and /csp2) between two independent NSD installations (listening on TCP port 7038 and 7039).

1. Add the following directives to the Apache configuration (httpd.conf):

   ```
   <Location /csp1>
       SetEnv CSP_NSD_PORT 7038
   </Location>
   <Location /csp2>
       SetEnv CSP_NSD_PORT 7039
   </Location>
   ```

2. Restart Apache after making changes to its configuration.

In cases where multiple instances of the NSD are running, it is recommended that the separate instances be installed in separate directories, each maintaining its own copy of the configuration and log file. The Web Gateway management pages for each instance can easily be accessed by using the NSD internal HTTP server. For example:

`http://localhost:7038/csp/bin/Systems/Module.cxw`

`http://localhost:7039/csp/bin/Systems/Module.cxw`

B

Alternative Options for IIS 7 or Later (Windows)

This page contains instructions for configuring atypical options for configuring Microsoft IIS for use with the InterSystems Web Gateway. For these options:

- Follow the steps in Microsoft IIS All Versions.
- Install ISAPI and GCI services, as described in Installing the ISAPI and CGI Services.
- Use the instructions in *one* of the following section:
  - Alternative Option 1: Using the ISAPI Modules (CSPms*.dll)
  - Alternative Option 2: Using a Native Module with the NSD (CSPcms.dll)
  - Alternative Option 3: Using an ISAPI Module with the NSD (CSPcms.dll)
  - Alternative Option 4: Using the CGI Modules with the NSD (nph-CSPcgi*.exe)

B.1 Installing the ISAPI and CGI Services

IIS 7 does not, by default, run **ISAPI extension**, **ISAPI filters**, or **CGI modules**. For all the atypical options for IIS 7, you must install these services.

Note that, with the **ISAPI extensions** service installed, all versions of the Web Gateway work with IIS 7.

Install these legacy services through the Windows Control Panel.

1. Open the Windows Control Panel.
2. Select **Programs and Features** and select **Turn Windows Features on or off**.
3. Navigate to **Internet Information Services** and expand **World Wide Web Services** and **Application Development Features**.

   Select **ISAPI Extensions**. Also select **ISAPI Filters** and **CGI**, if these additional services are required. Select **OK**.
4. In the Windows **Control Panel**, open **Administrative Tools** and **Internet Information Services (IIS) Manager**.
5. In the left panel, highlight **[MACHINE_NAME] ([machine_name]\[user_name])**
6. In the middle panel, double-click the **Modules** icon.

7. In the right panel, select **Add Native Module**.
8. In the left panel, expand the top level, expand **Web Sites** and expand **Default Web Site**

```
[MACHINE_NAME] ([machine_name]\[user_name])
              Web Sites
                     Default Web Site
```

9. In the middle panel, double-click **Handler Mappings**.
10. In the middle panel, highlight the **ISAPI-dll** handler.
11. In the right panel, select **Edit Handler Permissions**.
12. Select **Execute** and select **OK**. This allows ISAPI extensions to be invoked through direct calls to the name of the ISAPI DLL.

B.2 Alternative Option 1: Using the ISAPI Modules (CSPms*.dll)

Use this option if your Web Gateway DLLs are unable to support the Native Module interface (the Recommended Option). This is the default (and best performing) solution that was supplied for earlier versions of IIS.

IIS 7 does not, by default, run **ISAPI extensions**, **ISAPI filters** or **CGI modules**. This option requires the **ISAPI extensions** service.

Follow the instructions in Installing the ISAPI and CGI Services (If Required) for installing and configuring the ISAPI extensions service.

The web server should be configured such that it recognizes InterSystems file types and passes them to the Web Gateway for processing.

B.2.1 Enabling the ISAPI Extensions

DLLs: CSPms.dll and CSPmsSys.dll

Before these extensions can be used they must be registered with IIS as being “Allowed” applications. This is done in the **Internet Information Services (IIS) Manager** control panel.

1. Open the **Internet Information Services (IIS) Manager** window.
2. In the left panel, highlight **[MACHINE_NAME] ([machine_name]\[user_name])**.
3. In the middle panel, double-click **ISAPI and CGI Restrictions**.
4. In the right panel, select **Add**.
5. In the **Add ISAPI or CGI Restriction** dialog, enter the following details:

   ISAPI or CGI Path: C:\Inetpub\CSPGateway\CSPms.dll

   Description: WebGatewayRunTime

   Allow extension path to execute: Select

   Select **OK**

B.2.2 Mapping InterSystems IRIS File Extensions

Choose *one* of the following configuration methods:

- Serve all content (including static content) from InterSystems IRIS. Map * to the Web Gateway. See Mapping Additional File Types.
- Serve static content from the web server. Map *only* the InterSystems file types to the Web Gateway.

If you are serving static files from the web server, map the InterSystems file types to the Web Gateway ISAPI extensions as follows:

Extension	Binary
*.csp	C:\Inetpub\CSPGateway\CSPms.dll
*.cls	C:\Inetpub\CSPGateway\CSPms.dll
*.zen	C:\Inetpub\CSPGateway\CSPms.dll
*.cxw	C:\Inetpub\CSPGateway\CSPms.dll

1. Open the **Internet Information Services (IIS) Manager** window.
2. In the left panel expand the top level to reveal the **Web Sites** section, then the **Default Web Site** section. Highlight the **Default Web Site** section:

```
[MACHINE_NAME] ([machine_name]\[user_name])
            Web Sites
                    Default Web Site
```

Note: This activates CSP for the whole web site. To restrict the use of CSP to specific virtual sub-directories (such as /csp/) focus control on the appropriate subdirectory (under **Default Web Site**) before creating the mappings. Repeat the process for each virtual subdirectory from which CSP content is to be served.

3. In the middle panel, double-click the **Handler Mappings** icon.
4. In the right panel, select **Add Script Map**.
5. In the **Add Script Map** dialog, enter:

   Request Path: *.csp

   Executable: C:\Inetpub\CSPGateway\CSPms.dll

   Name: WebGateway_csp
6. Select **Request Restrictions**.

   Clear: **Invoke handler only if request is mapped to**

   Select **OK** to return to **Add Script Map** dialog.

   Select **OK**.
7. At this point you may be prompted as follows:

   “Would you like to enable this ISAPI extension? If yes, we add your extension as an “Allowed” entry in the ISAPI and CGI Restrictions list. If the extension already exists we allow it.”

   Select **Yes**.

   You can later find the list of allowed applications as follows:

In the left panel, highlight:

[MACHINE_NAME] ([machine_name]\[user_name])

In the middle panel, double-click **ISAPI and CGI Restrictions**.

If the Web Gateway ISAPI components are not included in the list of allowed applications, add them.

You can add text of your own choice in the **Description** field. For example:

`WebGatewayManagement for CSPmsSys.dll`

`WebGatewayRunTime for CSPms.dll`

8. Repeat the above process: Use the **Add Script Map** dialog to enter the following two mappings:

   Request Path: *.cls

   Executable: C:\Inetpub\CSPGateway\CSPms.dll

   Name: WebGateway_cls

   Request Path: *.zen

   Executable: C:\Inetpub\CSPGateway\CSPms.dll

   Name: WebGateway_zen

   Request Path: *.cxw

   Executable: C:\Inetpub\CSPGateway\CSPms.dll

   Name: WebGatewayManagement

B.2.3 Operating and Managing the Web Gateway

To access the Web Gateway’s systems management suite, point your browser at one of the following locations:

http://<ip_address>/csp/bin/Systems/Module.cxw

http://<ip_address>/csp/bin/CSPmsSys.dll

If you see an unauthorized user error message, see “Web Gateway and Security.”

B.3 Alternative Option 2: Using a Native Module with the NSD (CSPcms.dll)

IIS 7 does not, by default, run **ISAPI extensions**, **ISAPI filters**, or **CGI modules**. This option requires the **CGI modules** service for running the Web Gateway Management module (nph-CSPcgiSys.exe).

Follow the instructions in Installing the ISAPI and CGI Services (If Required.

Configure the web server so that it recognizes InterSystems file types and passes them to the Web Gateway for processing.

B.3.1 Registering the Runtime Native Module

DLL: CSPcms.dll

Before this module can be used, it must be registered with IIS. This is done in the **Internet Information Services (IIS) Manager** control panel.

1. Open the **Internet Information Services (IIS) Manager** window.
2. In the left panel, highlight:

   [MACHINE_NAME] ([machine_name]\[user_name])

3. In the middle panel, double-click the **Modules** icon.
4. In the right panel, select **Add Native Module**.
5. Select **Register** and enter the following details in the **Register Native Module** dialog:

   Name: CSPcms

   Path: C:\Inetpub\CSPGateway\CSPcms.dll

   Select **OK**.

6. In the left panel, expand the top level to reveal the **Web Sites** section, then the **Default Web Site** section. Highlight the **Default Web Site** section:

```
[MACHINE_NAME] ([machine_name]\[user_name])
            Web Sites
                    Default Web Site
```

7. In the right panel, select **Add Native Module**.
8. In the **Add Native Module** dialog select **CSPcms** then select **OK**.

B.3.2 Enabling the CGI module for Web Gateway Management

Executable: nph-CSPcgiSys.exe

Before this module can be used, it must be registered with IIS as being an Allowed application. This is done in the **Internet Information Services (IIS) Manager** control panel.

1. Open the **Internet Information Services (IIS) Manager**.
2. In the left panel, highlight:

   [MACHINE_NAME] ([machine_name]\[user_name])

3. In the middle panel, double-click the **ISAPI and CGI Restrictions** icon.
4. In the right panel, select **Add**.
5. In the **Add ISAPI or CGI Restriction** dialog, enter:

   ISAPI or CGI Path: C:\Inetpub\CSPGateway\nph-CSPcgiSys.exe

   Description: WebGatewayManagement

   Allow extension path to execute: Select

   Select **OK**.

B.3.3 Mapping InterSystems IRIS File Extensions

Note: Do not use **Add Wildcard Script Mapping** utility for this file extension mapping process; it gives an error. Instead, use the utility called **Add Module Mapping for ***.

Choose *one* of the following configuration methods:

- Serve all content (including static content) from InterSystems IRIS. Map * to the Web Gateway. If you are configuring the web application so that the InterSystems IRIS server serves all static files, then see Mapping Additional File Types.
- Serve static content from the web server.

  Map *only* InterSystems file types to the Web Gateway.

If you are serving static files from the web server, map the InterSystems file types to the Web Gateway modules as follows:

Extension	Native Module	Binary
*.csp	CSPms	C:\Inetpub\CSPGateway\CSPms.dll
*.cls	CSPms	C:\Inetpub\CSPGateway\CSPms.dll
*.zen	CSPms	C:\Inetpub\CSPGateway\CSPms.dll
*.cxw		C:\Inetpub\CSPGateway\nph-CSPcgiSys.exe

1. Open the **Internet Information Services (IIS) Manager** window.
2. In the left panel, expand the top level to reveal the **Web Sites** section, then the **Default Web Site** section. Highlight the **Default Web Site** section:

   ```
   [MACHINE_NAME] ([machine_name]\[user_name])
               Web Sites
                       Default Web Site
   ```

   Note: This activates CSP for the whole web site. To restrict the use of CSP to specific virtual sub-directories (such as /csp/) focus control on the appropriate subdirectory (under **Default Web Site**) before creating the mappings. Repeat the process for each virtual subdirectory from which CSP content is to be served.

3. In the middle panel, double-click the **Handler Mappings** icon.
4. In the right panel, select **Add Module Mapping**.
5. In the **Add Module Mappings** dialog, enter:

   Request Path: *.csp

   Module: Select CSPcms

   Name: WebGateway_csp

6. Select **Request Restrictions**.

   Clear: **Invoke handler only if request is mapped to**

   Select **OK** to return to the **Add Module Mappings** dialog.

   Select **OK**.

7. Repeat the above process to add the following Module Mappings:

   Request Path: *.cls

   Module: Select **CSPcms**

   Name: WebGateway_cls

   and

   Request Path: *.zen

   Module: Select **CSPcms**

Name: WebGateway_zen

8. In the left panel, highlight the **Default Web Site** section:

```
[MACHINE_NAME] ([machine_name]\[user_name])
            Web Sites
                    Default Web Site
```

9. In the middle panel, double-click the **Handler Mappings** icon.
10. In the right panel, select **Add Script Map**.
11. In the **Add Script Map** dialog, enter:

    Request Path: *.cxw

    Executable: C:\Inetpub\CSPGateway\nph-CSPcgiSys.exe

    Name: WebGatewayManagement

12. Select **Request Restrictions**.

    Clear: **Invoke handler only if request is mapped to**

    Select **OK** to return to the **Add Script Map** dialog.

    Select **OK**.

13. You may be prompted as follows: “Would you like to enable this ISAPI extension? If yes, we add your extension as an “Allowed” entry in the ISAPI and CGI Restrictions list. If the extension already exists we allow it.”

    Select **Yes**.

    You can later find the list of allowed applications as follows:

    In the left panel, highlight:

    [MACHINE_NAME] ([machine_name]\[user_name])

    In the center panel, double-click the **ISAPI and CGI Restrictions** icon.

    If the Web Gateway Management CGI module is not included in the list of allowed applications, add it.

    You can add text of your own choice in the **Description** field. For example:

    `WebGatewayManagement for nph-CSPcgiSys.exe`

B.3.4 Operating and Managing the Web Gateway

This connectivity option depends on the Web Gateway’s network service daemon (NSD).

Start the CSP NSD as described in the section, *Starting the NSD*.

To access the Web Gateway’s Systems Management suite, point your browser at one of the following locations:

http://<ip_address>/csp/bin/Systems/Module.cxw

http://<ip_address>/csp-bin/nph-CSPcgiSys

If you see an unauthorized user error message, see *Web Gateway and Security*.

B.4 Alternative Option 3: Using an ISAPI Module with the NSD (CSPcms.dll)

Use this option if your Web Gateway DLLs are unable to support the Native Module interface (Alternative Option 2).

IIS 7 does not, by default, run **ISAPI extensions**, **ISAPI filters** or **CGI modules**. This option requires both the **ISAPI extensions** and the **CGI modules** service.

Follow the instructions in Installing the ISAPI and CGI Services.

The web server should be configured such that it recognizes InterSystems file types and passes them to the Web Gateway for processing.

B.4.1 Enabling the Runtime ISAPI Extension

DLLs: CSPcms.dll

Before this extension can be used, it must be registered with IIS as being “Allowed” applications. This is done in the **Internet Information Services (IIS) Manager** control panel.

1. Open the **Internet Information Services (IIS) Manager** window.
2. In the left panel, highlight: **[MACHINE_NAME] ([machine_name]\[user_name])**
3. In the middle panel, double-click the **ISAPI and CGI Restrictions** icon.
4. In the right panel, select **Add**.
5. In the **Add ISAPI or CGI Restriction** dialog, enter:

   ISAPI or CGI Path: C:\Inetpub\CSPGateway\CSPcms.dll

   Description: WebGatewayRunTime

   Allow extension path to execute: Select

   Select **OK**

B.4.2 Enabling the CGI module for Web Gateway Management

Executable: nph-CSPcgiSys.exe

Before this module can be used, it must be registered with IIS as being an Allowed application. This is done in the **Internet Information Services (IIS) Manager** control panel.

1. Open the **Internet Information Services (IIS) Manager** window.
2. In the left panel, highlight: **[MACHINE_NAME] ([machine_name]\[user_name])**
3. In the middle panel, double-click the **ISAPI and CGI Restrictions** icon.
4. In the right panel, select **Add**.
5. In the **Add ISAPI or CGI Restriction** dialog, enter:

   ISAPI or CGI Path: C:\Inetpub\CSPGateway\nph-CSPcgiSys.exe

   Description: WebGatewayManagement

   Allow extension path to execute: Select

Select **OK**.

B.4.3 Mapping InterSystems IRIS File Extensions

Choose *one* of the following configuration methods:

- Serve all content (including static content) from InterSystems IRIS. Map * to the Web Gateway. If you are configuring the web application in InterSystems IRIS so that the InterSystems IRIS server serves all static files, see Mapping Additional File Types.
- Serve static content from the web server.

  Map *only* InterSystems file types to the Web Gateway.

If you are serving static files from the web server, map the InterSystems file types to the Web Gateway Modules as follows:

Extension	Binary
*.csp	C:\Inetpub\CSPGateway\CSPcms.dll
*.cls	C:\Inetpub\CSPGateway\CSPcms.dll
*.zen	C:\Inetpub\CSPGateway\CSPcms.dll
*.cxw	C:\Inetpub\CSPGateway\nph-CSPcgiSys.exe

1. Open the **Internet Information Services (IIS) Manager** window.
2. In the left panel, expand the top level and expand **Web Sites**. Highlight **Default Web Site**.

```
[MACHINE_NAME] ([machine_name]\[user_name])
               Web Sites
                         Default Web Site
```

   Note: This activates CSP for the whole web site. To restrict the use of CSP to specific virtual sub-directories (such as /csp/) focus control on the appropriate subdirectory (under **Default Web Site**) before creating the mappings. Repeat the process for each virtual subdirectory from which CSP content is to be served.

3. In the middle panel, double-click **Handler Mappings**.
4. In the right panel, select **Add Script Map**.
5. In the **Add Script Map** dialog, enter:

   Request Path: *.csp

   Executable: C:\Inetpub\CSPGateway\CSPcms.dll

   Name: WebGateway_csp

6. Select **Request Restrictions**.

   Clear: **Invoke handler only if request is mapped to**

   Select **OK** to return to the ‘Add Script Map’ dialog.

   Select **OK**.

7. At this point you may be prompted as follows:

   “Would you like to enable this ISAPI extension? If yes, we add your extension as an “Allowed” entry in the ISAPI and CGI Restrictions list. If the extension already exists we allow it.”

   Select **Yes**.

You can later find the list of allowed applications as follows:

In the left panel, highlight:

[MACHINE_NAME] ([machine_name]\[user_name])

In the middle panel, double-click the **ISAPI and CGI Restrictions** icon.

If the Web Gateway ISAPI module is not included in the list of allowed applications, add it.

You can add text of your own choice in the **Description** field. For example:

`WebGatewayRunTime for CSPcms.dll`

`WebGatewayManagement for nph-CSPcgiSys.exe`

8. Repeat the above process: Use the **Add Script Map** dialog to enter the following two mappings:

   Request Path: *.cls

   Executable: C:\Inetpub\CSPGateway\CSPcms.dll

   Name: WebGateway_cls

   Request Path: *.zen

   Executable: C:\Inetpub\CSPGateway\CSPcms.dll

   Name: WebGateway_zen

   Request Path: *.cxw

   Executable: C:\Inetpub\CSPGateway\nph-CSPcgiSys.exe

   Name: WebGatewayManagement

B.4.4 Operating and Managing the Web Gateway

This connectivity option depends on the Web Gateway’s network service daemon (NSD).

1. Start the CSP NSD as described in the section dedicated to this service.

To access the Web Gateway’s Systems Management suite, point your browser at one of the following locations:

http://<ip_address>/csp/bin/Systems/Module.cxw

http://<ip_address>/csp-bin/nph-CSPcgiSys

If you see an unauthorized user error message, see “Web Gateway and Security.”

B.5 Alternative Option 4: Using the CGI Modules with the NSD (nph-CSPcgi*.exe)

In most cases, the all-inclusive Native Module-based solution (the Recommended Option) is the option of choice, and is the implementation that gives the best performance. The CGI/NSD hybrid is useful for cases where it is necessary, for operational reasons, to manage the Web Gateway independently of the hosting web server. For example, if multiple instances of the web server are to share the same Web Gateway installation. In option 1 each instance of the core web server process binds to its own instance of the Web Gateway.

Another factor in choosing this approach might be that the in-house requirements of your web master (or ISP) dictate that all web server extensions are implemented using the CGI protocol.

IIS 7 does not, by default, run **ISAPI extensions**, **ISAPI filters** or **CGI modules**. This option requires the **CGI modules** service.

Follow the instructions in for installing the CGI service, Installing the ISAPI and CGI Services.

Configure the web server so that it recognizes InterSystems file types and passes them to the Web Gateway for processing.

B.5.1 Enabling the CGI Modules

Executables: nph-CSPcgi.exe and nph-CSPmsSys.exe

Before these modules can be used they must be registered with IIS as being “Allowed” applications. This is done in the **Internet Information Services (IIS) Manager** control panel.

1. Open the **Internet Information Services (IIS) Manager** window.
2. In the left panel, highlight:

   [MACHINE_NAME] ([machine_name]\[user_name])

3. In the middle panel, double-click the **ISAPI and CGI Restrictions** icon.
4. In the right panel, select **Add**.
5. In the **Add ISAPI or CGI Restriction** dialog, enter:

   ISAPI or CGI Path: C:\Inetpub\CSPGateway\nph-CSPcgi.exe

   Description: WebGatewayRunTime

   Allow extension path to execute: Select

   Select **OK**.

6. Repeat the above steps for nph-CSPcgiSys.exe, entering the following details in the **Restrictions** dialog:

   ISAPI or CGI Path: C:\Inetpub\CSPGateway\nph-CSPcgiSys.exe

   Description: WebGatewayManagement

   Allow extension path to execute: Select

B.5.2 Mapping InterSystems IRIS File Extensions

Choose *one* of the following configuration methods:

- Serve all content (including static content) from InterSystems IRIS. Map * to the Web Gateway. If you are configuring the web application in InterSystems IRIS so that the InterSystems IRIS server serves all static files, see Mapping Additional File Types.
- Serve static content from the web server.

  Map *only* InterSystems file types to the Web Gateway.

If you are serving static files from the web server, map the InterSystems file types to the Web Gateway CGI modules as follows:

Extension	Binary
*.csp	C:\Inetpub\CSPGateway\nph-CSPcgi.exe
*.cls	C:\Inetpub\CSPGateway\nph-CSPcgi.exe

Extension	Binary
*.zen	C:\Inetpub\CSPGateway\nph-CSPcgi.exe
*.cxw	C:\Inetpub\CSPGateway\nph-CSPcgiSys.exe

1. Open the **Internet Information Services (IIS) Manager** window.
2. In the left panel expand the top level to reveal the **Web Sites** section, then the **Default Web Site** section. Highlight the **Default Web Site** section:

   ```
   [MACHINE_NAME] ([machine_name]\[user_name])
               Web Sites
                       Default Web Site
   ```

   Note: This activates CSP for the whole web site. To restrict the use of CSP to specific virtual sub-directories (such as /csp/) focus control on the appropriate subdirectory (under **Default Web Site**) before creating the mappings. Repeat the process for each virtual subdirectory from which CSP content is to be served.

3. In the middle panel, double-click the **Handler Mappings** icon.
4. In the right panel, select **Add Script Map**.
5. In the **Add Script Map** dialog, enter:

   Request Path: *.csp

   Executable: C:\Inetpub\CSPGateway\nph-CSPcgi.exe

   Name:WebGateway_csp

6. Select **Request Restrictions**.

   Clear: **Invoke handler only if request is mapped to**

   Select **OK** to return to the **Add Script Map** dialog.

   Select **OK**.

7. At this point you may be prompted as follows: “Would you like to enable this ISAPI extension? If yes, we add your extension as an “Allowed” entry in the ISAPI and CGI Restrictions list. If the extension already exists we allow it.”

   Select **Yes**.

8. You can later find the list of allowed applications as follows:

   In the left panel, highlight:

   [MACHINE_NAME] ([machine_name]\[user_name])

   In the middle panel, double-click the **ISAPI and CGI Restrictions** icon.

   If the Web Gateway CGI components are not included in the list of allowed applications, add them.

   You can add text of your own choice in the **Description** field. For example:

   `WebGatewayManagement for nph-CSPcgiSys.exe`

   `WebGatewayRunTime for nph-CSPcgi.exe`

9. Repeat the above process: Use the **Add Script Map** dialog to enter the following two mappings:

   Request Path: *.cls

   Executable: C:\Inetpub\CSPGateway\nph-CSPcgi.exe

   Name: WebGateway_cls

Request Path: *.zen

Executable: C:\Inetpub\CSPGateway\nph-CSPcgi.exe

Name: WebGateway_zen

Request Path: *.cxw

Executable: C:\Inetpub\CSPGateway\nph-CSPcgiSys.exe

Name: WebGatewayManagement

B.5.3 Operating and Managing the Web Gateway

This connectivity option depends on the Web Gateway’s network service daemon (NSD).

1. Start the CSP NSD as described in the section dedicated to this service.

To access the Web Gateway’s Systems Management suite, point your browser at one of the following locations:

http://<ip_address>/csp/bin/Systems/Module.cxw

http://<ip_address>/csp-bin/nph-CSPcgiSys

If you see an unauthorized user error message, see Web Gateway and Security.

C

Alternative Options for Apache (Windows)

This page contains information for installations with Apache web servers on Microsoft Windows, for use with the InterSystems Web Gateway. Read the sections that apply to your installation.

This page describes additional Apache configurations for Microsoft Windows. To get started with all these configurations, read the first section. Then follow the directions in the section that applies to your installation.

C.1 Install Locations (All Atypical Options)

The following modules are installed:

- CSPcgi.exe (Runtime module)
- nph-CSPcgi.exe (Copy of CSPcgi.exe)
- CSPcgiSys.exe (Systems-Management module)
- nph-CSPcgiSys.exe (Copy of CSPcgiSys.exe)

Note: There are separate binaries for Apache Version 2.4.x are shown below:

- mod_csp24.dll (Apache built-in module as a DLL, if supplied)
- CSPa24.dll (Runtime module, if supplied)
- CSPa24Sys.dll (Web Gateway Systems Management module, if supplied)

The default location for these binaries is:

C:\Program Files\Apache Group\Apache\WebGateway\bin

The original location (*install-dir*\csp\bin) is used to hold the Web Gateway components required for serving the Management Portal for the specific instance of InterSystems IRIS.

The configuration and log file are written in this directory for non NSD-based connectivity options.

The modules with Sys appended are special modules for accessing the Web Gateway systems management suite. The runtime modules (that is, those without Sys) have no access to the systems management forms.

C.2 Alternative Option 1: Apache and CGI Modules with NSD (nph-CSPcgi.exe)

Configure the web server such that it recognizes InterSystems file types and passes them to the Web Gateway for processing.

The web server configuration file (httpd.conf) is in the following directory:

C:\Program Files\Apache Group\Apache\conf

Add the following section to the end of httpd.conf:

```
<LocationMatch "/*\.([Cc][Ss][Pp]|[Cc][Ll][Ss]|[Zz][En][Nn])$">
    AllowOverride None
    Options FollowSymLinks ExecCGI
    Require all granted
</LocationMatch>
ScriptAliasMatch /*\.([Cc][Ss][Pp]|[Cc][Ll][Ss])$ "c:/iris/csp/bin/nph-CSPcgi.exe"
Alias /csp/ c:/iris/csp/
<Directory "c:/iris/csp">
    AllowOverride None
    Options MultiViews FollowSymLinks ExecCGI
    Require all granted
    <FilesMatch "\.(log|ini|pid|exe)$">
        Require all denied
    </FilesMatch>
</Directory>
ScriptAlias /csp-bin/ "c:/iris/csp/bin/"
ScriptAliasMatch /csp/bin/Systems/Module.cxw "c:/iris/csp/bin/nph-CSPcgiSys.exe"
ScriptAliasMatch /csp/bin/RunTime/Module.cxw "c:/iris/csp/bin/nph-CSPcgi.exe"
<Directory "c:/iris/csp/bin/">
    AllowOverride None
    Options None
    Require all granted
    <FilesMatch "\.(exe)$">
        Allow from all
    </FilesMatch>
</Directory>
```

The above configuration block relies on the Regular Expressions (`regex`) processor being available to the Apache environment. Sometimes this is not the case, particularly with Windows systems, and CSP files are consequently not served (`File not found` errors are returned). To remedy this situation, associate the (virtual) root location of your web applications with the CGI module instead of making the association through the CSP file extensions. For example, your web applications are in /csp. To associate the CSP CGI module with files under /csp, replace the following configuration block:

```
<LocationMatch "/*\.([Cc][Ss][Pp]|[Cc][Ll][Ss]|[Zz][En][Nn])$">
    AllowOverride None
    Options FollowSymLinks ExecCGI
    Require all granted
</LocationMatch>
ScriptAliasMatch /*\.([Cc][Ss][Pp]|[Cc][Ll][Ss])$
      "c:/iris/csp/bin/nph-CSPcgi.exe"
```

with

```
<Location "/csp">
    AllowOverride None
    Options FollowSymLinks ExecCGI
    Require all granted
</Location>
ScriptAlias /csp "c:/iris/csp/bin/nph-CSPcgi.exe"
```

These directives work for URLs of the form:

```
http://localhost:<port_no>/csp/*.csp
```

Duplicate this configuration block for other root locations. For example, repeat the process for /myapps for URLs of the form:

```
http://localhost:<port_no>/myapps/*.csp
```

Another approach to avoiding the `regex` issue is to use an `Action` directive in conjunction with a CSP MIME type. However, note that `Action` is a content filtering technique and, as such, requires that your CSP files are physically present on the web server host even if the InterSystems IRIS server is installed on a separate computer.

To use this approach:

1. Add a new MIME type to the end of the Apache mime.types file and associate it with the InterSystems file types that represent classes: .csp, .cls, and .zen. The mime.types file is in the same directory as the httpd.conf file:

   ```
   text/csp                csp cls
   ```

2. Add the `Action` directive to the end of the CGI configuration block in httpd.conf such that it reads:

   ```
   Alias /csp/ c:/iris/csp/
   <Directory "c:/iris/csp">
       AllowOverride None
       Options MultiViews FollowSymLinks ExecCGI
       Require all granted
   <Files CSPnsd.exe>
          Require all denied
       </Files>
   <Files CSP.ini>
          Require all denied
       </Files>
   <Files CSP.log>
          Require all denied
       </Files>
   <Files CSPnsd.ini>
          Require all denied
       </Files>
   <Files CSPnsd.pid>
          Require all denied
       </Files>
       <FilesMatch "\.(log|ini|pid|exe)$">
          Require all denied
       </FilesMatch>
   </Directory>
   ScriptAlias /csp-bin/ "c:/iris/csp/bin/"
   <Directory "c:/iris/csp/bin/">
       AllowOverride None
       Options None
       Require all granted
   </Directory>
   Action text/csp “/csp-bin/nph-CSPcgi.exe”
   ```

   Finally, note that because CGI is an open standard, the CSP CGI modules work with any web server.

3. Restart Apache after making changes to httpd.conf.

C.2.1 Mapping Additional File Types

Apache API modules always recognize InterSystems file types. For other file extensions, see Configuring Apache to Pass Additional File Types.

C.2.2 Operating and Managing the Web Gateway with Apache NSD

This connectivity option depends on the Web Gateway’s network service daemon (NSD).

1. Start the CSP NSD as described in Operating the NSD.
2. Restart Apache after making changes to its configuration (httpd.conf).

   The order in which Apache and the NSD are started is unimportant.

3. To access the Web Gateway management pages, point your browser at one of the following locations:

   ```
   http://localhost:<port_no>/csp/bin/Systems/Module.cxw
   http://localhost:<port_no>/csp-bin/nph-CSPcgiSys
   ```

   If you see an `Unauthorized User` error message, see Enabling Access from Additional Client Addresses.

C.3 Alternative Option 2: Apache API Module with NSD (mod_csp24.dll)

Note: This connectivity option is not used as often as the stand-alone API modules described in Option 1; however, it can be used if you need to use the NSD. The CSP module, built as a DLL (mod_csp24.dll – for Apache 2.4), performs better than the CGI-based solution (Option 2).

1. Edit the Apache configuration file httpd.conf. For the standard Apache distribution this file is in:

   C:\Program Files\Apache Group\Apache\conf

   To invoke CSP for files with the .csp, .cls, and .zen extensions, add the following section to the end of httpd.conf. For Apache v2.4.x, specify mod_csp24.dll.

   ```
   LoadModule csp_module c:/iris/csp/bin/mod_csp24.dll
   CSPFileTypes csp cls zen cxw
   Alias /csp/ /iris/csp/
   <Directory "c:/iris/csp">
           AllowOverride None
           Options MultiViews FollowSymLinks ExecCGI
           Require all granted
           <FilesMatch "\.(log|ini|pid|exe)$">
                   Require all denied
           </FilesMatch>
           <Files CSPnsd>
                   Require all denied
           </Files>
   </Directory>
   ScriptAlias /csp-bin/ "c:/iris/csp/bin/"
   ScriptAliasMatch /csp/bin/Systems/Module.cxw
                 "c:/iris/csp/bin/nph-CSPcgiSys.exe"
   ScriptAliasMatch /csp/bin/RunTime/Module.cxw
                 "c:/iris/csp/bin/nph-CSPcgi.exe"
   <Directory "c:/iris/csp/bin/">
           AllowOverride None
           Options None
           Require all granted
   </Directory>
   ```

2. Restart Apache after making changes to httpd.conf.

C.3.1 Mapping Additional File Types

To configure additional file types to be processed by the CSP engine, include the new file extension(s) in the list of usual file extensions (.csp, .cls, .zen) to be processed by the CGI module. For example, add them to the following line:

```
ScriptAliasMatch /*\.([Cc][Ss][Pp]|[Cc][Ll][Ss]|[Zz][En][Nn])$  "C:/iris/csp/bin/nph-CSPcgi.exe"
```

If you need to serve other static files through the Web Gateway or need to access the Management Portal through this web server, add mappings for file types .jpg, .gif, .png, .svg, .css, and .js.

The following directive can be used to map requests for all files to InterSystems IRIS (specifically by the CSP engine) for a given path.

```
ScriptAliasMatch ^/csp/*/.* "C:/iris/csp/bin/nph-CSPcgi.exe"
```

Therefore, a basic configuration block for mapping requests for all files in the /csp path would be:

```
ScriptAliasMatch ^/csp/*/.* "C:/iris/csp/bin/nph-CSPcgi.exe"
<Directory "/iris/csp/bin/">
   AllowOverride None
   Options None
   Require all granted
</Directory>
```

C.3.2 Operating and Managing the Web Gateway with Apache API and NSD

This connectivity option depends on the Web Gateway’s network service daemon (NSD).

1. Start the CSP NSD as described in Operating the NSD.
2. Restart Apache after making changes to its configuration (httpd.conf).

   The order in which Apache and the NSD are started is unimportant.
3. To access the Web Gateway management pages, point your browser at one of the following locations.

   ```
   http://localhost:<port_no>/csp/bin/Systems/Module.cxw
   http://localhost:<port_no>/csp-bin/nph-CSPcgiSys.exe
   ```

   If you see an `Unauthorized User` error message, see Enabling Access from Additional Client Addresses.

C.4 Locked-down Apache Environments for Microsoft Windows

Occasionally Apache is locked-down such that you cannot easily configure the server to access files outside the Apache file system.

For configurations locked down in this way, the Web Gateway configurations discussed in previous sections result in `HTTP 403 Forbidden` error codes being returned when you try to access CSP resources. To work with these secure configurations, copy the file system under: *install-dir*\csp\

to a location under the Apache root:

C:\Program Files\Apache Group\Apache\

Specify appropriate changes to the paths specified in the Apache configuration.

An alternative approach is to configure the Web Gateway to work within the pre-configured directories provided by Apache.

1. Copy CGI modules to: C:\Program Files\Apache Group\Apache\cgi-bin\ as follows:

   ```
   copy c:\iris\csp\bin\*cgi*.exe C:\Program Files\Apache Group\Apache\cgi-bin\
   ```

2. Copy API modules to C:\Program Files\Apache Group\Apache\modules:

   ```
   copy c:\iris\csp\bin\*.dll C:\Program Files\Apache Group\Apache\modules\
   ```

3. Copy static files (and their subdirectories) to locations under C:\Program Files\Apache Group\Apache\htdocs\csp\samples.

```
copy c:\iris\csp\samples\*.*
  C:\Program Files\Apache Group\Apache\htdocs\csp\samples\
copy c:\iris\csp\broker\*.*
  C:\Program Files\Apache Group\Apache\htdocs\csp\broker\
copy c:\iris\csp\sys\*.*
  C:\Program Files\Apache Group\Apache\htdocs\csp\sys\
```

4. Install the NSD component (if required) in C:\Program Files\Apache Group\Apache\nsd.

Using the pre-configured directories in Apache simplifies the Web Gateway configuration in httpd.conf. Modified configuration blocks are shown below.

C.4.1 Configuration for Recommended Option: Apache API Modules (CSPa24.dll)

```
LoadModule csp_module_sa
    C:/Program Files/Apache Group/Apache/modules/CSPa24.dll
CSPFileTypes csp cls zen cxw
```

C.4.2 Configuration for Alternative Option 2: CGI Modules with NSD (nph-CSPcgi.exe)

```
<LocationMatch "/*\.([Cc][Ss][Pp]|[Cc][Ll][Ss]|[Zz][En][Nn])$">
AllowOverride None
Options FollowSymLinks ExecCGI
Require all granted
</LocationMatch>
ScriptAliasMatch /csp/bin/Systems/Module.cxw "C:/Program Files/Apache
Group/Apache/cgi-bin/nph-CSPcgiSys.exe"
ScriptAliasMatch /csp/bin/RunTime/Module.cxw "C:/Program Files/Apache Group/Apache/cgi-bin/nph-CSPcgi.exe"
ScriptAliasMatch /*\.([Cc][Ss][Pp]|[Cc][Ll][Ss])$ "C:/Program Files/Apache
Group/Apache/cgi-bin/nph-CSPcgi.exe"
```

C.4.3 Configuration for Alternative Option 3: Apache API Module with NSD (mod_csp24.dll)

```
LoadModule csp_module
           C:/Program Files/Apache Group/Apache/modules/mod_csp24.dll
CSPFileTypes csp cls zen cxw
ScriptAliasMatch /csp/bin/Systems/Module.cxw "C:/Program Files/Apache
Group/Apache/cgi-bin/nph-CSPcgiSys.exe"
ScriptAliasMatch /csp/bin/RunTime/Module.cxw "C:/Program Files/Apache Group/Apache/cgi-bin/nph-CSPcgi.exe"
```

C.4.4 Configuration for Alternative Option 4: ISAPI Modules (CSPms.dll)

Legacy only

```
AddHandler isapi-isa dll
AddHandler isapi-isa csp
AddHandler isapi-isa cls
AddHandler isapi-isa zen
AddHandler isapi-isa cxw
Alias /csp/bin/Systems/Module.cxw /csp/bin/CSPmsSys.dll
```

D

Using the NSD (UNIX®/Linux/macOS)

This page describes how to use the Network Service Daemon (NSD) or use with the Web Gateway on UNIX®, Linux, or macOS. This is not the typical installation but is appropriate in some cases.

D.1 When to Use the NSD

There are three situations in which you might choose to use the NSD to separate the Web Gateway from the web server so that you can manage the Web Gateway independently of the web server:

- If your web server distributes its load over multiple server processes, an instance of the Web Gateway is then attached to each web server process.
- If you have a very large web server installation for which CSP is only a small part; for example, a web server that serves php, static content, .NET, and .ASP applications, as well as web applications.
- If you are using the Nginx web server.

D.2 NSD Module Install Locations

The NSD Module, if required, is `CSPnsd`.

The default location for this module is:

```
/opt/webgateway/bin
```

The NSD should be run from within its home directory (above). The configuration and log files are written in this directory for NSD-based connectivity options.

D.3 Operating the NSD

To run the NSD:

1. Change to the following directory:

   /opt/webgateway/bin

2. Enter the following command to start the NSD:

   ```
   ./CSPnsd
   ```

Before retiring to the background, the NSD displays a banner indicating its running configuration. It shows the TCP port number dedicated to this service, which is, by default, port number 7038.

You can suppress all startup messages for this command using the `-s` qualifier. For example, to start the NSD from a script invoked at system boot, use:

```
/opt/webgateway/bin/CSPnsd –s
```

Other common startup options:

- Display help information.

  ```
  ./CSPnsd -h
  ```

- Pause the operation of the NSD. This command sends a stop signal (SIGSTOP) to the NSD process.

  ```
  ./CSPnsd -pause
  ```

- Continue the operation of the NSD (after a pause). This command sends a continue signal (SIGCONT) to the NSD process.

  ```
  ./CSPnsd -cont
  ```

- Give permission to others to run the NSD. Administrators of the NSD (CSPnsd) component can give permission to a group or others to start/stop the NSD using `CSPnsd —m=`*s* where *s* is a startup option.

  s can be one of

  - `u` for the current user (default)
  - `g` for the current group
  - `o` for others
  - `a` for everyone (`m=ugo`)

  Example: `CPSnsd -m=ug` gives permissions to the group (the Administrator group) to run the NSD. This command gives the CPSnsd.pid permissions of: `-rw-rw—--`

  When the command to stop the CSPnsd is issued, it tries to signal the CSPnsd parent process to shut down as before. If this is not possible because the service was started by a different user, a flag is written to the CSPnsd.ini file and the service gracefully closes itself down when it acknowledges this flag. This process takes up to 20 seconds to complete.

To close down the NSD, enter:

```
./CSPnsd -stop
```

Alternatively:

```
kill –TERM `cat /opt/webgateway/bin/CSPnsd.pid`
```

These commands close down the NSD in an orderly manner – it gracefully terminates all open connections to InterSystems IRIS and releases all its system resources before terminating. Do not use the **kill –9** command to terminate the NSD.

All errors are reported in the Web Gateway Event Log.

D.3.1 Starting the NSD on Alternative TCP Port

By default, the NSD listens for incoming requests on TCP port 7038. You can override this by starting the service as follows, where *port_no* is the TCP port number of your choice.

```
./CSPnsd [port_no]
```

Or:

```
./CSPnsd -p=[port_no]
```

On startup, the NSD creates the following file:

/opt/webgateway/bin/CSPnsd.ini

Typically, this file contains the following lines:

```
[SYSTEM]
Ip_Address=127.0.0.1
TCP_Port=7038
```

In this context, the clients are the Web Gateway module contained within, or dynamically linked to, the web server and/or the Web Gateway CGI modules invoked by the server. It is, therefore, essential that this file is not deleted or moved. It is also important that the web server processes can read this file. Set the privileges accordingly, bearing in mind the UNIX® username under which your web server is operating. The NSD clients attempt to find this file in the following locations:

/opt/webgateway/bin

/etc

If the NSD is operating in a different directory, you have to move the CSPnsd.ini file to one of the locations listed.

It is inappropriate to store the NSD port number in the CSPnsd.ini file in situations in which multiple instances of the NSD are running. For Apache servers, there is a much better mechanism for communicating the TCP port number of the NSD to its clients. Specifically, set the following environment variables in the Apache configuration to indicate the address and port of the target NSD installation.

- `CSP_NSD_NAME` — This is the IP address of the NSD. Only use this parameter if the NSD is operating on a remote computer.
- `CSP_NSD_PORT` — This is the TCP port of the NSD.

The values specified in these environment variables take precedence over any values found in the CSPnsd.ini file.

D.3.1.1 Example 1: Two Apache Virtual Hosts

To distribute the load for two Apache virtual hosts (123.123.1.1 and 123.123.1.2) between two independent NSD installations (listening on TCP port 7038 and 7039), add the following directives to the Apache configuration (httpd.conf):

```
<VirtualHost 123.123.1.1>
    ServerName 123.123.1.1
    SetEnv CSP_NSD_PORT 7038
</VirtualHost>
<VirtualHost 123.123.1.2>
    ServerName 123.123.1.2
    SetEnv CSP_NSD_PORT 7039
</VirtualHost>
```

D.3.1.2 Example 2: Two Web Applications

To distribute the load for two web applications (/csp1 and /csp2) between two independent NSD installations (listening on TCP port 7038 and 7039), add the following directives to the Apache configuration (httpd.conf):

```
<Location /csp1>
    SetEnv CSP_NSD_PORT 7038
</Location>
<Location /csp2>
    SetEnv CSP_NSD_PORT 7039
</Location>
```

Restart Apache after making changes to its configuration.

In cases where multiple instances of the NSD are running, it is recommended that the separate instances be installed in separate directories, each maintaining its own copy of the configuration and log files. The Web Gateway management pages for each instance can easily be accessed by using the NSD's internal HTTP server. For example:

```
http://localhost:7038/csp/bin/Systems/Module.cxw
```

```
http://localhost:7039/csp/bin/Systems/Module.cxw
```

D.3.1.3 Spreading the Load over Multiple NSD Processes

By default, the NSD operates in a two-process mode of operation (one parent and one child worker).

However, there are limits to the number of threads that a single UNIX® process can start. If the concurrent load of the web application is resulting in requests queuing for available threads, consider raising the number of processes used by the NSD.

```
./CSPnsd –c=[no_processes]
```

- where no_processes is the number of child (or worker) processes to start.

It should be noted that there are even advantages in setting the number of child processes to one.

```
./CSPnsd –c=1
```

Under these circumstances, the NSD actually starts two processes: a parent and one child worker process. The presence of the parent processes when using the '–c' directive improves the resilience of the NSD because if a fault develops in one of the worker processes the parent can replace the process. For the single, multi-threaded architecture, the NSD cannot always recover from serious internal error conditions.

State-aware connectivity (preserve mode 1) should not be used in cases where the number of worker processes exceeds one.

D.3.1.4 Granting Administrator Rights to the NSD

Administrators of the NSD (CSPnsd) component can have some control over the user (or group) permitted to start/stop this service.

In the default scenario, the CSPnsd master process ID (PID) file (CSPnsd) is created such that only the user who started the service can subsequently close it down.

Administrators can now choose, for example, to allow all users belonging to the current UNIX® group to manage the service. This is the group to which the administrating user belongs.

```
NSD start-up option: [-m=s]
    Define the user(s) permitted to manage this service
        where 's' is:
            'u' for the current user (the default),
            'g' for the current group,
            'o' for others,
            'a' for everyone (m=ugo),
```

Example:

```
./CSPnsd -m=ug
```

This allows the current user and all others in the current user's group to manage the NSD.

When the command to stop the NSD is issued, it first tries to signal the CSPnsd parent process to shut down as before. If this is not possible due to the service having been started by a different user, a flag is written to the CSPnsd.ini file and the service gracefully closes itself down when it acknowledges this flag. This process takes up to 20 seconds to complete.

E

Alternative Options for Apache (UNIX®/Linux/macOS)

This page describes additional possible Apache configurations for use with the InterSystems Web Gateway on UNIX®, Linux, and macOS (apart from a locked-down Apache, discussed separately). To get started with all these configurations, read the first section. Then follow the directions in the section that applies to your configuration.

E.1 Install Locations (All Atypical Options)

This section describes directory locations for Web Gateway files and CSP static files.

1. The NSD module is:

   ```
   CSPnsd
   ```

   The default location of this module is:

   ```
   /opt/webgateway/bin
   ```

   The NSD should be run from within its home directory /opt/webgateway/bin. The configuration and log files are written in this directory.

   In order to avoid disrupting existing Gateway installations on upgrading InterSystems IRIS®, the installation places the following modules in the common location /opt/webgateway/bin. This location is not related to a particular InterSystems IRIS instance.

2. CGI and other dynamically-linked modules:

   - CSPcgi (Runtime module)
   - nph-CSPcgi (Copy of CSPcgi)
   - CSPcgiSys (Systems-Management module)
   - nph-CSPcgiSys (Copy of CSPcgiSys)
   - mod_csp24.so (Apache Version 2.4.x — Apache module as a DSO, if supplied)

In order to avoid disrupting existing Gateway installations on upgrading InterSystems IRIS, the installation procedures place these modules in the following common location. This location is not related to a particular InterSystems IRIS instance.

```
/usr/cspgateway/bin
```

The original location (*install-dir*/csp/bin) is used to hold the Web Gateway components required for serving the Management Portal for the specific instance of InterSystems IRIS.

The modules with Sys appended access the Web Gateway management pages. The runtime modules (that is, those without Sys) have no access to the Web Gateway management pages.

3. The default location for the HyperEvents components:

   - CSPBroker.js
   - CSPxmlhttp.js

   and miscellaneous static resources (such as image files) are required by the CSP Samples and the Management Portal is:

   install-dir\csp\broker

E.1.1 Requirements for using Apache API Modules (Recommended Option and Alternative Option 1)

Before following instructions for either the recommended option (Recommended Option: Apache API Module without NSD (CSPa24.so)) or atypical option 1 (Alternative Option 1: Apache API Module with NSD (mod_csp24.so)), check that your build of Apache includes the built-in module for managing shared objects (mod_so). To perform this check, run the following command which lists the modules currently available within Apache:

```
httpd -l
```

The shared object module (mod_so) should appear in the list of modules displayed. The following shows a typical module listing (with mod_so included):

```
Compiled in modules:
  core.c
  mod_access.c
  mod_auth.c
  mod_include.c
  mod_log_config.c
  mod_env.c
  mod_setenvif.c
  prefork.c
  http_core.c
  mod_mime.c
  mod_status.c
  mod_autoindex.c
  mod_asis.c
  mod_cgi.c
  mod_negotiation.c
  mod_dir.c
  mod_imap.c
  mod_actions.c
  mod_userdir.c
  mod_alias.c
  mod_so.c
```

If mod_so is not included in the list for your Apache installation, see your Apache documentation and follow the procedure for rebuilding Apache to include this module.

E.2 Alternative Option 1: Apache API Module with NSD (mod_csp24.so)

If the CSP module is supplied with your distribution as a pre-built shared object (mod_csp24.so), then start at Runtime Configuration. To build the shared object from the supplied source file mod_csp.c choose Method 1 or Method 2 below. Method 1 is preferred.

Be sure to read the following instructions regarding the creation of shared objects in conjunction with the specific documentation contained within your Apache distribution. Note that the instructions given here assume that the root directory for the Apache installation is /usr/apache. In practice, this directory name usually has the Apache version number appended to it.

E.2.1 Method 1: Building the CSP Module as Shared Object with apxs (APache eXtenSion) Tool

The following command builds and installs the shared library, mod_csp24.so, in the Apache /modules directory using the Apache extension tool, **apxs**. It also adds a directive to load the module to the Apache configuration file /conf/httpd.conf.

```
apxs –c –o mod_csp24.so mod_csp.c
```

Copy the shared object produced (mod_csp24.so) into the following directory: /opt/webgateway/bin.

E.2.2 Method 2: Building the CSP Module as Shared Object Manually

Perform the following steps to manually build the CSP module as a shared object:

1. Install the module source file mod_csp.c in the following directory: /usr/apache/src/modules/extra
2. Return to the /usr/apache/src directory and edit the Configuration file. Near the end of this file, locate the following line:

   ```
   # AddModule modules/example/mod_example.o
   ```

   After this line, add the following line:

   ```
   ShareModule modules/extra/mod_csp24.so
   ```

3. Configure the build process using the following command:

   ```
   ./Configure
   ```

4. Build the shared object using the following command:

   ```
   make
   ```

   To produce shared object mod_csp24.so in /usr/apache/src/modules/extra

Note: For further information about the `apxs` tool, see the Apache documentation at https://httpd.apache.org/docs/2.4/programs/apxs.html.

E.2.3 Runtime Configuration

Edit the Apache configuration file httpd.conf. For the standard Apache distribution, this file is in:

```
/usr/apache/conf
```

For Red Hat Linux, the runtime version of httpd.conf is in:

```
/etc/httpd/conf
```

Assuming that you wish to invoke the CSP engine for requested files that contain a .csp, .cls, or .zen extension, add the following section to the end of httpd.conf.

```
LoadModule csp_module /opt/webgateway/bin/mod_csp24.so
CSPFileTypes csp cls zen cxw
Alias /csp/ /opt/webgateway/csp/
<Directory "/opt/webgateway/csp">
        AllowOverride None
        Options MultiViews FollowSymLinks ExecCGI
        Require all granted
        <FilesMatch "\.(log|ini|pid|exe)$">
                Require all denied
        </FilesMatch>
        <Files CSPnsd>
                Require all denied
        </Files>
</Directory>
ScriptAlias /csp-bin/ "/opt/webgateway/bin/"
ScriptAliasMatch /csp/bin/Systems/Module.cxw "/opt/webgateway/bin/nph-CSPcgiSys"
ScriptAliasMatch /csp/bin/RunTime/Module.cxw "/opt/webgateway/bin/nph-CSPcgi"
<Directory "/opt/webgateway/bin/">
        AllowOverride None
        Options None
        Require all granted
</Directory>
```

Restart Apache after making changes to httpd.conf.

E.2.3.1 Controlling Connection Pooling

The size of the connection pool can be controlled by the following Apache configuration parameter (specified in http.conf):

```
CSPMaxPooledNSDConnections <no>
```

In the absence of this parameter, a default value of 32 is used internally – which is effectively:

```
CSPMaxPooledNSDConnections 32
```

To switch-off connection pooling, set this parameter to zero:

```
CSPMaxPooledNSDConnections 0
```

If, for any reason, it becomes necessary to use the legacy (asymmetric) mode of operation (whereby the Web Gateway notifies the end of response transmission by closing the connection on its side), set this parameter to minus 1:

```
CSPMaxPooledNSDConnections -1
```

E.2.3.2 Operating and Managing the Web Gateway with Apache API and NSD

This connectivity option depends on the Web Gateway's network service daemon (NSD).

1. Start the CSP NSD as described in Operating the NSD.
2. Restart Apache after making changes to its configuration (httpd.conf).

The order in which Apache and the NSD are started is unimportant.

3. To access the Web Gateway management pages, enter the following URL in your browser.

   ```
   http://localhost:<port_no>/csp-bin/nph-CSPcgiSys
   ```

   If you see an `Unauthorized User` error message, see Enabling Access from Additional Client Addresses.

E.2.4 Mapping Additional File Types

Apache API modules always recognize the InterSystems file types. For other file extensions, see Configuring Apache to Pass Additional File Types .

E.3 Alternative Option 2: CGI Modules with NSD (nph-CSPcgi)

The web server should be configured such that it recognizes InterSystems file types and passes them to the Web Gateway for processing.

The web server configuration file (httpd.conf) is found in the following directory:

```
/usr/apache/conf
```

For Red Hat Linux, the runtime version of httpd.conf is found in:

```
/etc/httpd/conf
```

Add the following section to the end of httpd.conf:

```
<LocationMatch "/*\.([Cc][Ss][Pp]|[Cc][Ll][Ss]|[Zz][En][Nn])$">
    AllowOverride None
    Options FollowSymLinks ExecCGI
    Require all granted
</LocationMatch>
ScriptAliasMatch /csp/bin/Systems/Module.cxw "/opt/webgateway/bin/nph-CSPcgiSys"
ScriptAliasMatch /csp/bin/RunTime/Module.cxw "/opt/webgateway/bin/nph-CSPcgi"
ScriptAliasMatch /*\.([Cc][Ss][Pp]|[Cc][Ll][Ss])$ "/opt/webgateway/bin/nph-CSPcgi"
Alias /csp/ instance-installation-directory
<Directory "instance-installation-directory">
    AllowOverride None
    Options MultiViews FollowSymLinks ExecCGI
   Require all granted
    <FilesMatch "\.(log|ini|pid|exe)$">
       Require all denied
    </FilesMatch>
    <Files CSPnsd>
       Require all denied
    </Files>
</Directory>
ScriptAlias /csp-bin/ "/opt/webgateway/bin/"
<Directory "/opt/webgateway/bin/">
    AllowOverride None
    Options None
    Require all granted
</Directory>
```

The above configuration block relies on the Regular Expressions (`regex`) processor being available to the Apache environment. Sometimes this is not the case and CSP files are consequently not served (`File not found` errors are returned). To remedy this situation, you can associate the (virtual) root location of your web applications with the CGI module instead of making the association through the CSP file extensions. For example, your web applications are contained in /csp. To associate the CSP CGI module with files under /csp, replace the following configuration block:

```
<LocationMatch "/*\.([Cc][Ss][Pp]|[Cc][Ll][Ss]|[Zz][En][Nn])$">
    AllowOverride None
    Options FollowSymLinks ExecCGI
    Require all granted
</LocationMatch>
ScriptAliasMatch /*\.([Cc][Ss][Pp]|[Cc][Ll][Ss])$ "/opt/webgateway/bin/nph-CSPcgi"
```

with:

```
<Location "/csp">
    AllowOverride None
    Options FollowSymLinks ExecCGI
   Require all granted
</Location>
ScriptAlias /csp "/opt/webgateway/bin/nph-CSPcgi"
```

These directives work for URLs of the form:

```
http://localhost:<port_no>/csp/*.csp
```

Duplicate the configuration block for other root Locations. For example, repeat the process for /myapps for URLs of the form:

```
http://localhost:<port_no>/myapps/*.csp
```

Another approach to avoiding the `regex` issue is to use an `Action` directive in conjunction with a CSP MIME type. However, it should be noted that `Action` is essentially a content filtering technique and, as such, requires that your CSP files are physically present on the web server host even if the InterSystems IRIS server is installed on a separate computer. If you wish to use this approach, first add a new MIME type to the end of the Apache mime.types file and associate it with file types representing CSP content. The mime.types file are found in the same directory as the httpd.conf file.

```
text/csp                csp cls
```

Now, add the `Action` directive to the end of the CGI configuration block in httpd.conf such that it reads:

```
Alias /csp/ /opt/webgateway/csp/
<Directory "/opt/webgateway/csp">
    AllowOverride None
    Options MultiViews FollowSymLinks ExecCGI
    Require all granted
<Files CSPnsd>
       Require all denied
    </Files>
<Files CSP.ini>
       Require all denied
    </Files>
<Files CSP.log>
       Require all denied
    </Files>
<Files CSPnsd.ini>
       Require all denied
    </Files>
<Files CSPnsd.pid>
       Require all denied
    </Files>
</Directory>
ScriptAlias /csp-bin/ "/opt/webgateway/bin/"
<Directory "/opt/webgateway/bin/">
    AllowOverride None
    Options None
    Require all granted
</Directory>
Action text/csp “/csp-bin/nph-CSPcgi”
```

Restart Apache after making changes to httpd.conf.

Finally, note that because CGI is an open standard, The CSP CGI modules work with any web server.

E.3.1 Mapping Additional File Types

Apache API modules always recognize the InterSystems file types. For other file extensions, see Configuring Apache to Pass Additional File Types.

E.3.2 Operating and Managing the Web Gateway with CGI and NSD

This connectivity option depends on the Web Gateway’s network service daemon (NSD).

1. Start the CSP NSD as described in Operating the NSD
2. Restart Apache after making changes to its configuration (httpd.conf).

   The order in which Apache and the NSD are started is unimportant.
3. To access the Web Gateway management pages, enter one of the following URLs in your browser.

   ```
   http://localhost:<port_no>/csp/bin/Systems/Module.cxw
   http://localhost:<port_no>/csp-bin/nph-CSPcgiSys
   ```

   If you see an `Unauthorized User` error message, see Enabling Access from Additional Client Addresses.

E.4 Alternative Option 3: Built-in Apache API Module with NSD (mod_csp.c)

Before embarking on setting up this more complicated option you should bear in mind that, for most modern UNIX® systems, the performance advantage in static linking over linking the module at runtime as a shared object (option 1) is minimal (if anything at all).

Be sure to read these instructions in conjunction with the specific documentation contained within your Apache distribution.

E.4.1 Build Apache to Include CSP Module Source Code

Refer to the Apache documentation for this step.

http://httpd.apache.org/

E.4.2 Check the Apache Binary Produced

Run the following command to check that the CSP module has been successfully included in the Apache core (this command lists all modules currently built-into Apache):

```
./httpd -l
```

For example:

```
Compiled in modules:
        core.c
        mod_access.c
        mod_auth.c
        mod_include.c
        mod_log_config.c
        mod_env.c
        mod_setenvif.c
        prefork.c
```

```
        http_core.c
        mod_mime.c
        mod_status.c
        mod_autoindex.c
        mod_asis.c
        mod_cgi.c
        mod_negotiation.c
        mod_dir.c
        mod_imap.c
        mod_actions.c
        mod_userdir.c
        mod_alias.c
        mod_csp.c
```

E.4.3 Runtime Configuration

Edit the Apache configuration file httpd.conf. For the standard Apache distribution this file is in:

```
/usr/apache/conf
```

For Red Hat Linux, the runtime version of httpd.conf is in:

```
/etc/httpd/conf
```

Assuming that you wish to invoke the CSP engine for requested files that contain a .csp, .cls, or .zen extension, add the following section to the end of httpd.conf:

```
CSPFileTypes csp cls zen cxw
ScriptAliasMatch /csp/bin/Systems/Module.cxw "/opt/webgateway/bin/nph-CSPcgiSys"
ScriptAliasMatch /csp/bin/RunTime/Module.cxw "/opt/webgateway/bin/nph-CSPcgi"
Alias /csp/ /opt/webgateway/csp/
<Directory "/opt/webgateway/csp">
        AllowOverride None
        Options MultiViews FollowSymLinks ExecCGI
        Require all granted
        <FilesMatch "\.(log|ini|pid|exe)$">
                Require all denied
        </FilesMatch>
        <Files CSPnsd>
                Require all denied
        </Files>
</Directory>
ScriptAlias /csp-bin/ "/opt/webgateway/bin/"
<Directory "/opt/webgateway/bin/">
        AllowOverride None
        Options None
        Require all granted
</Directory>
```

Note that all requests to Apache are serviced by a set of modules invoked in a predefined sequence. The CSP module is one of the first modules invoked, provided its definition was added near the end of the Configuration file as suggested.

Restart Apache after making changes to httpd.conf.

E.4.4 Mapping Additional File Types

Apache API modules always recognize the InterSystems file types. For other file extensions, see Configuring Apache to Pass Additional File Types.

E.4.5 Operating and Managing the Web Gateway with Apache API and NSD

This connectivity option depends on the Web Gateway’s network service daemon (NSD).

1. Start the CSP NSD as described in Operating the NSD.
2. Restart Apache after making changes to its configuration (httpd.conf).

The order in which Apache and the NSD are started is unimportant.

3. To access the Web Gateway management pages, point your browser at one of the following locations.

```
http://localhost:<port_no>/csp/bin/Systems/Module.cxw
http://localhost:<port_no>/csp-bin/nph-CSPcgiSys
```

If you see an `Unauthorized User` error message, see Enabling Access from Additional Client Addresses.

F

Locked-down Apache (UNIX®/Linux/macOS)

In some cases, Apache is locked down so that you cannot easily configure the server to access files outside the Apache file system. For example, this is the case for Security Enhanced Linux (SELinux).

For configurations locked down in this way, the Web Gateway configurations discussed elsewhere result in `HTTP 403 Forbidden` error codes being returned on attempting to access CSP resources. There are two strategies for dealing with this problem.

- Modify the security context for the Web Gateway's home directory so that Apache can access files held in this location.
- Move the Web Gateway's home directory to a location under the Apache root file system (which is pre-configured to be accessible to Apache in the SELinux setup).

First, modifying the SELinux security context for the Web Gateway's home directory is usually straightforward and involves the following steps.

We use, as an example, a Web Gateway home directory of /opt/webgateway/bin, the InterSystems IRIS Superserver listening on port 1972 and InterSystems IRIS installed in /usr/iris/.

The *chcon* command sets file context and takes effect immediately.

```
sudo chcon -R -t httpd_sys_content_t /usr/iris/csp
sudo chcon -R -t httpd_sys_rw_content_t /opt/webgateway/conf/CSP.ini
sudo chcon -R -t httpd_sys_rw_content_t /opt/webgateway/conf/CSPRT.ini
sudo chcon -R -t httpd_sys_rw_content_t /opt/webgateway/logs/CSP.log
sudo chcon -t httpd_modules_t /opt/webgateway/bin/CSPa2.so
sudo chcon -t httpd_modules_t /opt/webgateway/bin/CSPa2Sys.so
sudo chcon -t httpd_modules_t /opt/webgateway/bin/CSPa22.so
sudo chcon -t httpd_modules_t /opt/webgateway/bin/CSPa22Sys.so
sudo chcon -t httpd_modules_t /opt/webgateway/bin/CSPa24.so
sudo chcon -t httpd_modules_t /opt/webgateway/bin/CSPa24Sys.so
```

However, changes made by the chcon command are lost after the next relabeling; therefore it is necessary use the semanage fcontext facility in addition to chcon. The following commands allow the Web Gateway to run when SELinux is enforced:

```
semanage fcontext -a -t lib_t "/opt/webgateway/bin/(.*\.so)?" 2> /dev/null
semanage fcontext -a -t httpd_sys_rw_content_t "/opt/webgateway/bin/temp(/.*)?" 2> /dev/null
semanage fcontext -a -t httpd_sys_rw_content_t "/opt/webgateway/bin/CSP(.*\.ini)?" 2> /dev/null
semanage fcontext -a -t httpd_sys_rw_content_t "/opt/webgateway/bin/CSP.log" 2> /dev/null
restorecon -vr /opt/webgateway/bin > /dev/null
```

Then use the commands shown below. Note that it is extremely important to properly set the context of the Superserver port (as shown in the last line); otherwise, the Web Gateway will not be able to access it, resulting in "Server unavailable" errors.

```
sudo /usr/sbin/semanage fcontext -a -t httpd_sys_content_t
    "/usr/iris/csp(/.)?"
sudo /usr/sbin/semanage fcontext -a -t httpd_sys_rw_content_t
    "/opt/webgateway/conf/CSP.ini"
sudo /usr/sbin/semanage fcontext -a -t httpd_sys_rw_content_t
    "/opt/webgateway/conf/CSPRT.ini"
sudo /usr/sbin/semanage fcontext -a -t httpd_sys_rw_content_t
    "/opt/webgateway/logs/CSP.log"
sudo /usr/sbin/semanage fcontext -a -t httpd_modules_t
    "/opt/webgateway/bin/CSPa2.so"
sudo /usr/sbin/semanage fcontext -a -t httpd_modules_t
    "/opt/webgateway/bin/CSPa2Sys.so"
sudo /usr/sbin/semanage fcontext -a -t httpd_modules_t
    "/opt/webgateway/bin/CSPa22.so"
sudo /usr/sbin/semanage fcontext -a -t httpd_modules_t
   "/opt/webgateway/bin/CSPa22Sys.so"
sudo /usr/sbin/semanage fcontext -a -t httpd_modules_t
    "/opt/webgateway/bin/CSPa24.so"
sudo /usr/sbin/semanage fcontext -a -t httpd_modules_t
    "/opt/webgateway/bin/CSPa24Sys.so"
sudo /usr/sbin/semanage port -a -t http_port_t -p tcp 51773
```

These are the basic steps for granting the Web Gateway (operating in the context of the hosting Apache server) access to files in its home directory.

An alternative approach (and the one that should be used if the method suggested above is not acceptable) is to configure the Web Gateway to work within the pre-configured directories provided by Apache. The following commands assume that Apache is installed in /usr/apache.

- CGI modules should be copied to: /usr/apache/cgi-bin/

  ```
  cp /usr/iris/csp/bin/*cgi* /usr/apache/cgi-bin/
  ```

- API modules should be copied to: /usr/apache/modules/

  ```
  cp /usr/iris/csp/bin/*.so /usr/apache/modules/
  ```

- Static files should be copied to locations under: /usr/apache/htdocs/

  ```
  cp /usr/iris/csp/samples/* /usr/apache/htdocs/csp/samples/
  cp /usr/iris/csp/broker/* /usr/apache/htdocs/csp/broker/
  cp /usr/iris/csp/sys/* /usr/apache/htdocs/csp/sys/
  ```

  Also, copy any sub-directories held under the above locations.

Having moved the Web Gateway installation, the appropriate changes to the paths specified in the Apache configuration must be made.

F.1 Recommended Option: Apache API Modules (CSPa24.so)

```
LoadModule cspsys_module_sa /usr/apache/modules/CSPap24.so
CSPSYSModulePath /usr/apache/modules/
CSPFileTypes csp cls zen cxw
```

F.2 Alternative Option 1: Apache API Module with NSD (mod_csp.so)

```
LoadModule csp_module /usr/apache/modules/mod_csp.so
CSPFileTypes csp cls zen cxw
ScriptAliasMatch /csp/bin/Systems/Module.cxw "/usr/apache/cgi-bin/nph-CSPcgiSys"
ScriptAliasMatch /csp/bin/RunTime/Module.cxw "/usr/apache/cgi-bin/nph-CSPcgi"
```

F.3 Alternative Option 2: CGI Modules with NSD (nph-CSPcgi)

```
<LocationMatch "/*\.([Cc][Ss][Pp]|[Cc][Ll][Ss]|[Zz][En][Nn])$">
AllowOverride None
Options FollowSymLinks ExecCGI
Require all granted
</LocationMatch>
ScriptAliasMatch /csp/bin/Systems/Module.cxw "/usr/apache/cgi-bin/nph-CSPcgiSys"
ScriptAliasMatch /csp/bin/RunTime/Module.cxw "/usr/apache/cgi-bin/nph-CSPcgi"
ScriptAliasMatch /*\.([Cc][Ss][Pp]|[Cc][Ll][Ss])$ "/usr/apache/cgi-bin/nph-CSPcgi"
```

F.4 Alternative Option 3: Built-in Apache API Module with NSD (mod_csp.c)

```
CSPFileTypes csp cls zen cxw
ScriptAliasMatch /csp/bin/Systems/Module.cxw "/apache/cgi-bin/nph-CSPcgiSys"
ScriptAliasMatch /csp/bin/RunTime/Module.cxw "/usr/apache/cgi-bin/nph-CSPcgi"
```

G

Apache Considerations (UNIX®/Linux/macOS)

This page contains information about the recommended option for UNIX®, Linux, and macOS (Recommended Option: NSAPI Modules (CSPn3.so)) and atypical option 1 (Alternative Option 1: Apache API Module with NSD (mod_csp24.so)).

G.1 Apache Process Management and Capacity Planning

Apache provides three process management modules for UNIX® operating systems. In this architecture, the InterSystems Web Gateway modules are directly bound to the Apache *worker processes*. Therefore, the way Apache is configured to manage its process pool has a direct effect on the Web Gateway.

Apache implements each of its process management models as a *Multi-Processing Module* (MPM).

Prefork MPM is the traditional multi-process (UNIX®) server architecture. It does not use threads and, as a result, there is no requirement that third-party API modules (DSOs) should be thread-safe. Reference: http://httpd.apache.org/docs/current/mod/prefork.html.

Worker MPM is the newer hybrid multithread/multi-process server architecture. It does use threads and all third-party API modules (DSOs) used should be thread-safe. Reference: http://httpd.apache.org/docs/current/mod/worker.html.

Event MPM is designed to allow more requests to be served simultaneously by passing off some processing work to the listener threads, freeing up the worker threads to serve new requests. Reference: http://httpd.apache.org/docs/current/mod/event.html.

In order to determine which of the server models is in use for an existing installation, call the Apache executable directly but qualified as follows:

```
httpd -V
```

Two further related listings are provided:

- `httpd –l` List all modules built-in to the server
- `httpd –L` List all modules and related configuration directives

The Web Gateway DSOs are thread-safe and can be deployed in any server model. A useful guide for Apache tuning can be found here: http://httpd.apache.org/docs/current/misc/perf-tuning.html.

G.1.1 Security

The parent process of all three server architectures is usually started from an account with superuser privileges assigned (root under UNIX®) in order to bind to TCP port 80. The child processes launched by Apache run as a lesser-privileged user. The User and Group directives (in the Apache configuration) are used to set the privileges of the Apache child processes. The child processes must be able to read all the content that they are responsible for serving (and have read/write access to the Web Gateway's configuration and Event Log files), but, beyond this, should be granted as few privileges as possible. Refer to the Apache documentation for further information.

G.1.2 Apache MPMs and the Web Gateway DSOs

The Web Gateway dynamically linked modules (DSOs) are thread-safe and can be deployed in any server model.

For all Multi-Processing Modules (MPMs), the `StartServers` directive specifies the number of child (worker) processes to start. This directive also indicates the number of instances of the Web Gateway DSOs that can be present – such as one per Apache child process.

All MPMs involve spreading the load over multiple child (worker) processes.

Although each Gateway instance is independently loaded by each and every Apache child process, the running configuration, connection table and form cache is held in a shared memory sector. The contents of the Web Gateway System Status form remain constant with every refresh (apart from changes happening as a result of activity updates, of course). The connection table (and connection numbers) displayed is common to the whole Apache instance and, because of this, an additional column indicating the web server process ID to which each InterSystems IRIS connection is associated is included.

G.1.2.1 Maximum Server Connections

While the Web Gateway load is spread over multiple web server processes, the Maximum Server Connections configuration parameter sets a single overall limit on the number of connections the Web Gateway can make to a particular InterSystems IRIS server. This means that the number of worker processes started by the hosting web server does not affect the maximum number of connections the Web Gateway can create. The maximum is also unaffected by the type of process making the connection and the MPM in use. (This model represents a change from previous versions, in which the Maximum Server Connections parameter was effected on a per process basis and acted as a general throttle influenced by multiple factors.)

For installations where most of the Apache workload is made up of InterSystems file types, it is better to not assign a value to the Web Gateway's Maximum Server Connections directive and control the amount of concurrent work that can be done (and by implication the number of connections to InterSystems IRIS) with the corresponding Apache configuration parameters. Setting an independent value for the Web Gateway's Maximum Server Connections directive would, however, make sense in installations where InterSystems file types represent only part of the workload for the Apache installation as a whole.

G.2 State-Aware Sessions (Preserve mode 1)

Support for state-aware sessions in a web server that distributes load over multiple worker processes relies on InterProcess Communications (IPC) protocols to manage the routing of requests between individual worker processes. Operating in this web server architecture, the Web Gateway has no control over which worker process handles any particular request.

The Web Gateway uses *UNIX® domain sockets* for its IPC protocol and the method for supporting state-aware sessions is described below.

As an example, consider a web server installation that distributes its load over 3 worker processes: P1, P2 and P3. Each worker process can potentially start any number of threads (T1, T2 … Tn) according to the web server MPM and configuration in use.

Suppose an application makes a request to mark its session as state-aware (preserve mode 1) and the Web Gateway acknowledges this instruction in process P2. The connection and (security context) to the now private InterSystems IRIS process is hosted by web server worker process P2. All further requests for that user/session must now be processed by worker process P2. However, the Web Gateway has no control over which worker process the web server routes subsequent requests to, so the Web Gateway must establish an IPC channel between P2 and (potentially) any other worker process in the set.

When the Web Gateway marks the connection as state-aware in P2, it starts a listening service in a separate, detached, thread. For log level v2, a message similar to the one shown below is written to the Event Log.

```
IPC Server
Process ID: 28457 Listening on Domain Socket: /tmp/csp28457.str
```

Now, say a further request for the same session is processed by worker process P3. The Web Gateway forwards that request to process P2 via the IPC channel previously established and waits for the response. For log level v2, a message similar to the one shown below is recorded:

```
Route request over IPC to another web server process
PrivateSession=2; pid_self=28456; ipc_to_pid=28457;
```

Of course, if a request for the session happens to be routed by the web server directly to P2, then no further routing is necessary in the Web Gateway environment, since P2 is hosting the session's private connection.

If the Web Gateway is unable to connect and forward a request to a previously created IPC channel, one of the following messages is recorded depending on the context in which the error was raised:

```
IPC CLIENT: Error
Cannot connect
```

Or:

```
IPC CLIENT: Error
Cannot send request
```

The most common reason for problems in this area is if Apache has closed (or recycled) a worker process (in the case of the example, P2). Of course, a process can crash (for example, with an access violation/SIGSEGV error) and, in this case, an error message is probably reported in the Apache error log.

Apache also, by default, periodically recycles worker processes.

If you use state-aware sessions, configure Apache such that it doesn't recycle worker processes by configuring the installation as follows.

- Set the value of `MaxConnectionsPerChild` to zero
- Set the value of `MaxSpareThreads` to the same value as `MaxRequestWorkers`

If it is not possible to prevent Apache periodically recycling processes (perhaps as a result of a malfunctioning module) and state-aware sessions must be used, then an NSD based Gateway configuration can be used. An NSD-based architecture avoids the problems discussed above because it effectively separates the process management of the Web Gateway from the web server. Options for using the Web Gateway's network service daemon (NSD) are covered in Using the NSD on Microsoft Windows and Using the NSD on UNIX®, Linux, and macOS.

H

IIS Technical Notes

For those interested who use IIS, this page describes application pools, web gardens, and bitness.

H.1 IIS Application Pools and Web Gardens

H.1.1 Application Pools

An application pool is a configuration that links one or more applications to a set of one or more worker processes. Because applications in an application pool are separated from other applications by worker process boundaries, an application in one application pool is not affected by problems caused by applications running in other application pools.

By creating new application pools and assigning Web sites and applications to them, it is possible to make the server more efficient and reliable. Applications working through pools are always available, even when a worker process serving a different application develops a fault.

Applications are defined by their path in IIS. For example: `/csp`

H.1.2 Web Gardens

For even greater reliability, it is possible to configure an application pool to be supported by multiple worker processes. An application pool that uses more than one worker processes is called a web garden. The worker processes in a web garden share the requests that arrive for that particular application pool. If a worker process fails, another worker process can continue to process other requests.

It should be noted that web gardens are different from web farms. A web garden is configured on a single server by specifying multiple worker processes for an application pool. Web farms use multiple servers for supporting a web site.

Creating a web garden for an application pool can enhance performance in the following situations:

- Robust processing of requests: When a worker process in an application pool is tied up (for example, when a script engine stops responding), other worker processes can accept and process requests for the application pool.
- Reduced contention for resources: When a web garden reaches a steady state, each new TCP/IP connection is assigned, according to a round-robin scheme, to a worker process in the web garden. This helps smooth out workloads and reduce contention for resources that are bound to a worker process.

H.1.3 Application Pools, Web Gardens, and CSP

Application Pool and Web Garden configurations do not affect the operation of NSD-based Gateway configurations because the ISAPI module communicating with the NSD does not pool any persistent information or other resources (such as connections to InterSystems IRIS). All persistent resources are held in the NSD module. The ISAPI module communicating with the NSD is unaffected by changes in the way it is managed by IIS.

The non-NSD based Gateway configuration (CSPms.dll and CSPmsSys.dll) is more sensitive to changes in the way ISAPI extensions are managed in IIS because the pooling of persistent resources (such as connections to InterSystems IRIS) takes place in the extension itself.

Application pools that are configured to use no more than one worker process have no visible impact on the way the Web Gateway operates within the context of a single web application path (for example, /csp). However, for configurations where multiple worker processes are used (a Web Garden) the workload for the Web Gateway is evenly distributed amongst all participating worker processes in the pool. Each worker process manages its own instance of the Web Gateway modules. This process management architecture does not pose a problem with respect to the way the Web Gateway operates but the following restrictions must be borne in mind:

- IIS must be restarted in order for changes to the Web Gateway's configuration to take effect. This must be done by completely restarting the World Wide Web Publishing service from the main Windows Services control panel; not through the Internet Services Manager control panel.
- The Web Gateway's Systems Management form (System Status) cannot be used to accurately monitor the connections used by web applications. At any given time the Systems Status reflects the status for the instance of the Web Gateway that happens to be attached to the current worker process (that is, the worker process that happens to service the Web Gateway's request).
- Each web application (as defined by the web path to the application) maintains its own pool of persistent connections to InterSystems IRIS. Also, each worker process within an application pool maintains its own pool of persistent connections to InterSystems IRIS. This gearing should be remembered when configuring the maximum and minimum number of connections to InterSystems IRIS that the Web Gateway uses. These settings apply to each and every Gateway instance in the pool.
- State-aware sessions (preserve mode 1) cannot be used with Web Garden configurations because there is no control over the instance of the Web Gateway which is used to serve any particular request. The net result is that it's not possible to route state-aware requests to their dedicated InterSystems IRIS processes in these configurations.

Note that the NSD-based options are not subject to these restrictions because the Web Gateway is managed independently of IIS.

Finally, the effect of certain worker process configuration parameters on the non-NSD version of the Web Gateway should be considered. In particular, the effect of the idle timeout and process recycling facility should be borne in mind.

H.1.4 Idle Timeout for Worker Processes

Often it is necessary to conserve system resources by terminating unused worker processes. It is possible to configure a worker process to gracefully close after a specified period of time. This feature can be used to better manage the resources when the processing load is heavy, when identified applications consistently fall into an idle state, or when new processing space is not available.

When a worker process is terminated, the instance of the Web Gateway that it manages also closes, and the pool of connections to InterSystems IRIS held by that Gateway instance is terminated. Of course, additional stateless connections can always be replaced in a way that is transparent to users of a web application, but state-aware sessions (preserve mode 1) terminate when their hosting connection is closed.

H.1.5 Recycling Worker Processes

IIS can be configured to periodically restart worker processes so that faulty web applications can be recycled. This facility helps to ensure that application pools remain healthy and that any leaked system resources are recovered.

It is possible to configure worker processes to restart based on elapsed time, number of requests served, scheduled times, and on the basis of memory usage.

The effect on the Web Gateway of closing worker processes was discussed in the previous section (Idle Timeout). The same considerations apply here. Because web applications can only interact with the Web Gateway through carefully managed channels, it is recommended that worker processes supporting the web applications should not be recycled.

H.2 Bitness — 32-bit Apps on 64-bit Servers for Windows

Note: This section applies to modules that are loaded into the address space of the hosting web server: ISAPI Extensions and Native Modules (CSPms[Sys].dll and CSPcms.dll). CGI modules are not affected since they run as a detached process with respect to IIS.

The **Enable 32-Bit Applications** setting applies to the Application Pool level, which makes it possible to set the bitness for a particular Application Pool. In a single server installation, you can configure one Application Pool to run native 64-bit applications and another to run 32-bit applications.

To access the bitness setting for an Application Pool, enter the IIS control panel:

1. Select **Application Pools** in the left hand panel.
2. Select the appropriate Application Pool.
3. Select **Advanced Settings** in the right hand panel.
4. The **Advanced Settings** dialog appears. The **Enable 32-Bit Applications** setting is found in the **General** section. This can be set to `True` or `False`.

Incidentally, this configuration setting can be manipulated at the Windows Command line using the *appcmd* command. For example:

```
appcmd set apppool /apppool.name:DefaultAppPool/enable32bitapponwin64:true
```

This sets the Application Pool `DefaultAppPool` to run in 32 bit mode.

It is also possible to list the Application Pools based on bitness using the *appcmd* command. For example, to list all the application pools running in 64 bit mode use the following command:

```
appcmd list apppools /enable32bitapponwin64:false
```

Finally, since application pools can be run in different bitness modes, it is necessary to ensure that Native Modules (and ISAPI extensions) that are loaded by the Application Pool are themselves of the correct bitness for the pool. For example, if the hosting application pool is 64-bit, then the 64-bit Gateway modules (such as CSPms[Sys].dll) must be used. If the hosting Application Pool is 32-bit, then the 32-bit Gateway modules must be used instead.

The bitness check for individual modules is done via a *preCondition* in the module's web.config file. For the Web Gateway, this file typically looks something like the following:

```
<?xml version="1.0" encoding="UTF-8"?>
<configuration>
   <system.webServer>
      <handlers>
         <add name="WebGateway_All" path="*" verb="*" modules="CSPms" resourceType="Unspecified" \
          preCondition="bitness64" />
      </handlers>
      <security>
         <requestFiltering>
            <hiddenSegments>
               <remove segment="bin" />
            </hiddenSegments>
         </requestFiltering>
      </security>
   </system.webServer>
</configuration>
```

Note the bitness setting in the *precondition* clause. In this case bitness is set to `bitness64` which means that IIS checks for 64-bit Gateway modules operating in a 64-bit Application Pool.

If a 32-bit Application Pool is used, then the 32-bit Gateway modules must be used and the *preCondition* set to `bitness32`.

If there is an inconsistency between the modules installed, the precondition clause, and/or the expectations of the hosting Application Pool, IIS returns an error condition similar to the one shown below.

```
Error:
The module(s) assigned to this handler mapping has the following preconditions that are not
present in the handler mapping:
bitness64
Runtime errors may occur if a handler mapping does not have a set of preconditions that are
equally as restrictive as, or more restrictive, than the module(s) assigned to the mapping.
Please ensure that this handler mapping has the correct preconditions, and that the
preconditions are not in conflict.
```

I

Using Web Applications with a Remote Web Server

I.1 Configuring the Web Server and Web Gateway

This section discusses how to set up a web server and the Web Gateway to provide access to a web application installed on a remote InterSystems IRIS server. The instructions refer to the web server as *Machine W* and to the computer running InterSystems IRIS® as *Machine I*. The setup includes the following procedures:

1. Install the Web Gateway on the Web Server Machine
2. Configure the Web Gateway
3. If Serving Static Files from the Web Server
4. Configure Web Server Paths

I.1.1 Install the Web Gateway on the Web Server Machine

Install the Web Gateway on the web server machine, *Machine W*, where IIS or Apache is running. See Installing the Web Gateway Only in the *Installation Guide* if you need more detailed information. During the installation process, follow these instructions:

1. In the **Setup Type** dialog box, select **Web Server** and select **Next**.
2. Review the installation name, type, and destination directory and, if correct, select **Install**.

This installation process creates the CSP directory structure on the web server, *Machine W* and creates virtual directory references for the /CSP and /CSP/Bin files.

I.1.2 Configure the Web Gateway

Next, adjust the Web Gateway configuration on the web server, *Machine W*. Although this configuration information is stored in the configuration file, always use the Web Gateway management pages to update the configuration:

1. Navigate to the Web Gateway management pages main menu by pointing a browser to:

   ```
   http://localhost/csp/bin/Systems/Module.cxw
   ```

(Bookmarking this URL is helpful). This link is for your external web server, not the Private Web Server supplied with InterSystems IRIS.

Note that the link above is correct if you are on the same system that the web server is running on using port 80. If you are trying to access the management pages from a remote system, you are denied access by default. You can configure the web server to allow remote access through the Web Gateway management pages **Default Parameters** page. Set the **System Manager Machine/s** field to the IP address of the remote system. This field accepts a comma- or plus-separated list of IP address of machines that can access the Web Gateway management pages.

For additional information, see Enabling Access from Additional Client Addresses.

2. Select **Server Access** in the left-hand menu. If IRIS is installed on the web server, the InterSystems IRIS installation configures a LOCAL web server to connect to the InterSystems IRIS instance on the local machine, *Machine W*.
3. Create a new server to represent the InterSystems IRIS instance running on the remote machine, *Machine I*:
   a. Select **Add Server**.
   b. Enter a name for the server (Machine I for example).
   c. Enter the **IP Address** and **Superserver TCP Port** of the remote InterSystems IRIS server running on *Machine I*.
   d. Modify the **Connection Security** settings to match the level of authentication expected by *Machine I* for Web Gateway connections. See Web Gateway and Security for details.
   e. Select **Save Configuration**.

   For additional information, see Adding a Server Configuration.
4. Select **Application Access** in the left-hand menu to associate the path to the web application on the remote InterSystems IRIS server, *Machine I*, with the server configuration previously created for *Machine I*. The default paths are predefined for / and /csp.
5. Create a new application path to represent the web application running on the remote machine, *Machine I*. You can either copy an existing configuration (such as /csp) or select **Add Application** to manually create a new path configuration. The path you create for the application must match that defined for the application in the InterSystems IRIS instance on *Machine I*.

   For example, the default path to the Management Portal is /csp/sys. If you are creating a new application, choose your own path name. For example: /myapp or /csp/myapp. Having created the new path, modify the **Server 0** parameter for the path such that it takes the value of the InterSystems IRIS server configuration that you previously set up for *Machine I*.
6. Finally, save the new path configuration.

   For additional information, see Adding an Application Path.

I.1.3 If Serving Static Files from the Web Server

If you are planning to serve static files from the web server, create directories on *Machine W* to represent your application path. These directories exist solely to hold static content such as image files. You *do not* have to place any CSP files here; they reside on *Machine I*.

Under the directory *install-dir*\CSP on *Machine W*, create \Samples and \User directories. Also create directories to represent other paths which may contain static components referenced in CSP pages. The example in the previous section requires you to create a directory for \myapp.

I.1.4 Configure Web Server Paths

The application paths in the previous steps correspond to requests for CSP pages in the equivalent locations. For example:

```
http://domain.com/myapp/login.csp
http://domain.com/csp/myapp/login.csp
http://domain.com/csp/sys/login.csp
```

Inheritance is applied in a hierarchical fashion. Consider the following request:

```
http://domain.com/csp/newapp/login.csp
```

The application path configuration for /csp/newapp is used if it exists. If not, the configuration defined for /csp is used instead.

The Web Gateway installation procedures configure the hosting web server to recognize the /csp virtual path. Typically, these same settings also apply to directories placed under /csp (/csp/myapp, for example).

If you create a new path (such as in the first example, /myapp), you must configure the web server to recognize this new virtual path. These procedures are different depending on the web server you use. Follow the procedures in that applies to your web server:

- Add Virtual Directories to IIS
- Add Aliases to Apache Configurations

I.1.4.1 Add Virtual Directories to IIS

The installation procedure for the Web Gateway configures the virtual directory *install-dir*\csp for web applications. If all of your applications are under this virtual directory (for example, *install-dir*\csp\myapp) and you are not using virtual hosts, you do not need to add virtual directories. The instructions in this section apply specifically to the IIS (Internet Information Services) web server version 7 or later.

Set up the application path resembling /myapp in the previous examples with properties similar to the /csp virtual directory which is automatically created for you during the Web Gateway installation.

1. Navigate to the **Internet Information Services** management dialog box, which is likely accessible from the **Administrative Tools** menu of the Windows Control Panel.
2. Expand the folders and right-click **Default Web Site**.
3. Point to **New** and select **Virtual Directory** to create a new directory record with the following values:

   ```
   Alias:                  myapp
   Directory:              C:\iris\csp\myapp
   ```

4. Either select **Save** and **Apply** all changes, or if you are using the wizard, select **Finish**.

Restart IIS to apply the changes.

I.1.4.2 Add Aliases to Apache Configurations

If you are using an Apache web server to control a remote InterSystems IRIS server and your application path is altered from the /csp default, you must manually add a corresponding alias to the Apache configuration file pointing to the local CSP directory.

For example, to remotely serve the web applications on the InterSystems IRIS instance iris on *Machine I* from the application path defined on the web server, /myapp/csp, add the following alias line to the httpd.conf file on *Machine W*:

```
Alias /myapp/csp "C:/iris/CSP"
```

Restart the Apache web server to apply the changes.

I.2 Accessing CSP on Multiple InterSystems IRIS Servers

Read this section if you need to configure a single web server to access one or more web applications on multiple InterSystems IRIS servers. This section uses the Management Portal as the example application. Adapt these procedures for your own web application. The Management Portal application is usually called with a URL in this format:

```
http://domain.com/csp/sys/UtilHome.csp
```

For additional information, see Define a Remote Server Connection in *System Administration Guide*.

I.2.1 Configuring the InterSystems IRIS Server for the Application Path

If you are content with using the InterSystems IRIS server name in the web application URL, follow this procedure. If you do not want the InterSystems IRIS server name displayed in the URL, then follow the procedure in the following section Changing the InterSystems IRIS Server Name in the URL.

1. On your web server, access the Web Gateway management pages main menu with:

   http://localhost:<port_no>/csp/bin/Systems/Module.cxw

2. Select **Server Access**. Add server configurations for iris1 and iris2. See Configuring Server Access for details.
3. Select **Application Access**. Create an application path /iris1/csp/sys/ with a **Default Server** of iris1. Create an application path /iris2/csp/sys/ with a **Default Server** of iris2. See Configuring Application Access for details.
4. If the web server is IIS, then set up virtual directories for /iris1 and /iris2 as described in Add Virtual Directories to IIS.

   If using an Apache web server, see Add Aliases to Apache Configurations.

To access the Management Portal on InterSystems IRIS servers iris1 and iris2, include the server name as part of the URL as follows:

```
http://domain.com/iris1/csp/sys/UtilHome.csp
http://domain.com/iris2/csp/sys/UtilHome.csp
```

I.2.2 Changing the InterSystems IRIS Server Name in the URL

If you do not want the InterSystems IRIS server name displayed in the web application URL, then follow the procedure in this section to create a substitute name.

Use the `CSPConfigName` parameter of the **%System.CSP.SetConfig** method for each server. This example uses linda as the substitute name for server iris1 and perry as the substitute name for a server iris2. You would use your own server and substitute names.

In an ObjectScript shell on the iris1 server, run:

```
d $System.CSP.SetConfig("CSPConfigName","linda")
```

In an ObjectScript shell on the iris2 server, run:

```
d $System.CSP.SetConfig("CSPConfigName","perry")
```

Then, complete the following steps:

1. On the web server, access the Web Gateway management pages main menu with:

   `http://localhost/csp/bin/Systems/Module.cxw`

2. Select **Server Access**. Add server configurations for iris1 and iris2. See Configuring Server Access for details.
3. Select **Application Access**. Create an application path /linda/csp/sys/ with a **Default Server** of iris1. Create an application path /perry/csp/sys/ with a **Default Server** of iris2. See Configuring Application Access for details.
4. If the web server is IIS, then set up virtual directories for /iris1 and /iris2 as described in Add Virtual Directories to IIS.
5. If using an Apache web server, see Add Aliases to Apache Configurations.

`CSPConfigName` also accepts a comma delineated list for CSP configuration names. This allows you to have multiple configuration names instead of a single value. For example:

```
d $System.CSP.SetConfig("CSPConfigName","linda,linda1,linda2,linda3")
```

To see other CSP global parameters, enter `%SYS>d $system.CSP.DisplayConfig()`. If you have set `CSPConfigName`, you may want to also set `WebServerURLPrefix` so that Studio uses the same URL construction. See WebServerURLPrefix.

I.3 Configuring Apache Virtual Hosts

An alternative method for accessing an application on multiple servers is to use virtual host arrangements. Virtual hosts are a common feature in Apache web server configurations and are straightforward to set up in this server environment. For example, consider two virtual hosts, each listening on a separate TCP port:

```
http://virtual_host1:81/csp/sys/UtilHome.csp
http://virtual_host2:82/csp/sys/UtilHome.csp
```

Both virtual_host1 and virtual_host2 are served by the same web server and Web Gateway.

The following shows the Apache configuration (httpd.conf) for this arrangement:

```
<VirtualHost virtual_host1:81>
   ServerName virtual_host1
</VirtualHost>

<VirtualHost virtual_host2:82>
   ServerName virtual_host2
</VirtualHost>
```

Configure the use of these virtual hosts as follows:

1. Navigate to the Web Gateway management pages main menu by pointing a browser to:

   `http://localhost/csp/bin/Systems/Module.cxw`

2. Select **Server Access** to create a server configuration for iris1 and iris2.
3. Select **Application Access** to create the application paths //virtual_host1/csp/sys/ and //virtual_host2/csp/sys/.

   Note the use of the double forward-slash (`//`) to introduce the virtual host name.

   Set the **Server 0** for path //virtual_host1/csp/sys/ to be the name of the server configuration set up for iris1 in the previous step.

   Set the **Server 0** for path //virtual_host2/csp/sys/ to be the name of the server configuration set up for iris2 in the previous step.

4. No changes are required in the configuration of the two remote InterSystems IRIS servers. The application path for the portal remains as /csp/sys/ in both cases.

See Virtual Hosts Overview for more information.

I.3.1 Virtual Hosts Overview

Virtual hosts are a means through which you can transparently serve applications on one or more instances through a common web server. Each server installation appears to operate as a separate web server.

The differentiating factor in virtual host setups can be one of the following:

1. Web server IP address — The server hosting the web server is exposed through two IP addresses. For example:

   ```
   123.123.123.1 == www.serverA.com
   123.123.123.2 == www.serverB.com
   ```

2. Web server port — This method is useful for testing different configurations, though it involves including the port number in the request for cases where non-standard TCP ports are used (TCP ports other than 80). For example:

   ```
   Web Server TCP Port 80 == www.serverA.com
   Web Server TCP Port 81 == www.serverB.com
   ```

3. Path — the preferred way of implementing virtual hosts. You register the two names and they translate to a single physical IP address for the web server. For example:

   ```
   www.serverA.com == 123.123.123.1
   www.serverB.com == 123.123.123.1
   ```

Regardless of which way you choose, set up a named slot for each InterSystems IRIS installation in the Web Gateway configuration (it does not need to be the same as the InterSystems IRIS instance name). The superserver port that the Web Gateway configuration (for each server) is pointing to is what is important.

For example:

```
www.serverA.com
www.serverB.com
```

Both are served by a single web server installation.

You can implement servers including mixtures of all three. Options 1 and 3 are identical from the browser perspective. You can configure each virtual host to have its own documents root, etc.

To extend the virtual host concept through to CSP, suppose you wish to run the same web application through two virtual hosts, but on different InterSystems IRIS instances. For example, one site for testing and another for production.

```
www.serverA.com/csp/login.csp ==> irisA
www.serverB.com/csp/login.csp ==> irisB
```

A web application's access to an InterSystems IRIS server is controlled through the Web Gateway **Application Access** configuration option. Typically, the following two entries are defined:

```
/
/csp
```

The name of the InterSystems IRIS server is associated with these application path definitions:

```
/ (Default Server == irisA)
/csp (Default Server == irisA)
```

The Web Gateway allows you to extend this configuration to include the name of a virtual host through which you access the application.

```
/ (Default Server == irisA)
/csp (Default Server == irisA)
//www.serverA.com/csp (Default Server == irisA)
//www.serverB.com/csp (Default Server == irisB)
```

You can then configure a separate InterSystems IRIS server for www.serverA.com/csp and www.serverB.com/csp as shown above. Introduce server names by `//`, as shown.

The current rules of inheritance apply. For example, if you request www.serverA.com/xxx/yyy.csp, then the InterSystems IRIS server defined for `/` is ultimately used, unless, you define an ultimate default for serverA as shown below:

```
/ (Default Server == irisL)
/csp (Default Server == irisL)
//www.serverA.com/ (Default Server == irisL)
//www.serverA.com/csp (Default Server == irisA)
//www.serverB.com/csp (Default Server == irisB)
```

Note: The servers specified in the Web Gateway configuration do not necessarily have to be *virtual*. For example, you can configure a single NSD installation to support several real Apache installations with a different set of InterSystems IRIS servers defined for each one. Further, you can configure each Apache server to support many virtual hosts.

The Web Gateway identifies the host for the application through the CGI environment variable *SERVER_NAME*.