Administrator’s Guide

Citrix® ICA® Java Client

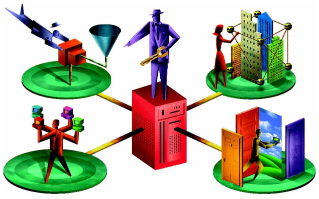
Version 7.0

Citrix Systems, Inc.

Last updated:11 April 2003 (SC)

Contents

Before You Begin



This guide is for system administrators responsible for installing, configuring, and maintaining the Citrix ICA Client for Java. This guide assumes knowledge of:

- The MetaFrame server to which the ICA Clients connect
- The operating system on the client device
- Java-enabled Web browsers, HTML, and JavaScript
- Installation, operation, and maintenance of network and printing systems

How to Use this Guide

To get the most out of the *Citrix ICA Java Client Administrator's Guide*, review the table of contents to familiarize yourself with the topics discussed:

Chapter	Contents
Chapter 2, "Introduction to the Citrix ICA Java Client"	Gives a list of features and system requirements.
Chapter 3, "Deploying the Citrix ICA Java Client"	Describes how to deploy the ICA Java Client.
Chapter 4, "Configuring the Citrix ICA Java Client"	Describes how to start and configure connections to MetaFrame servers.
Chapter 5, "Improving the Performance of the Citrix ICA Java Client"	Provides advice about how to improve performance after initial configuration.
Chapter 6, "Limitations of the Citrix ICA Java Client"	Describes known issues with the Java Client and how to work around them.
Appendix A, "ICA Java Client Parameters and File Descriptions"	Provides a complete listing of all the applet parameters, plus details of other files referred to in the guide.

Conventions

The following terms, text formats, and symbols are used throughout the guide:

Convention	Meaning
Bold	Names of interface items such as dialog boxes and buttons, column headings, command-line commands and options, lists, menu names, tabs, and menu commands.
Italic	Placeholders for information or parameters that you provide. For example, *filename* in a procedure means you type the actual name of a file. Italics also indicate new terms and the titles of books.
ALL UPPERCASE	Keyboard keys (for example, CTRL, ENTER, F2).
[brackets]	Enclose optional items in syntax statements. For example, [password] indicates that you can choose to type a password with the command. Do not type the brackets themselves.
...(ellipsis)	A command element you can repeat.
`Monospace`	Text displayed at the command line or in a text file.

The Citrix ICA Java Client allows users to connect to MetaFrame servers. When describing a feature unique to a particular MetaFrame server, this guide specifies the appropriate server and version number.

Note This guide uses the word Windows to describe Microsoft Windows 95, Windows 98, Windows NT, Windows 2000, Windows XP, and Windows ME operating systems unless otherwise stated.

Finding More Information

This guide contains conceptual information and installation and configuration steps for the ICA Java Client. For additional information, consult the following:

- The online help for the ICA Java Client
- The *Citrix ICA Client Administrator's Guides* for the other ICA Clients you deploy
- The documentation included in the MetaFrame server package for instructions about installing, configuring, and maintaining your MetaFrame servers
- The *Web Interface for MetaFrame XP Administrator's Guide*, if you are deploying the Java Client through the Web Interface.

This guide and other Citrix documentation is available in Adobe PDF format in the following locations:

- The documentation directory of your MetaFrame XP Components CD-ROM.
- http://www.citrix.com/download/. Click the ICA Client platform for which you want information.
- http://www.citrix.com/support/. Click Product Documentation and choose ICA Client.

Using the Adobe Acrobat Reader, you can view and search the documentation electronically or print it for easy reference. You can download the Adobe Acrobat Reader for free from the Adobe Web site at http://www.adobe.com .

Important Always consult the Readme files for your MetaFrame server and the Citrix ICA Client for any last-minute updates, installation instructions, and corrections to the documentation.

Citrix on the World Wide Web

The Citrix Web site is at http://www.citrix.com. The site offers a variety of information and services for Citrix customers and users.

From the Citrix home page, you can access Citrix technical support services and other information designed to assist MetaFrame administrators.

The following are some of the resources available on the Citrix Web site:

Citrix Product Documentation Library. The library, which contains the latest documentation for all Citrix products, is at http://www.citrix.com/support (select Product Documentation). You can download updated editions of the documentation that ships with Citrix products, as well as supplemental documentation that is available only on the Web site.

Citrix ICA Clients. Downloadable Citrix ICA Clients for all supported platforms are available from http://www.citrix.com/download.

Support options. Program information about Citrix Preferred Support Services options is available from the Support area of the Citrix Web site at http://www.citrix.com/support.

Software downloads. An FTP server provides access to the latest service packs, hotfixes, and utilities.

Online knowledge base. The online Solution Knowledge Base contains an extensive collection of application notes, technical articles with answers to frequently answered questions, troubleshooting tips, and white papers. The Knowledge Base is available at http://www.citrix.com/support.

Discussion forums. The interactive online Solution Forums provide outlets for discussion of technical issues with other Citrix users.

Education. Information about programs and courseware for Citrix training and certifications is available from http://www.citrix.com/training/.

Contact information. The Web site provides contact information for Citrix offices, including the worldwide headquarters and headquarters for European, Asia Pacific, and Japan operations.

Developer network. The Citrix Developer Network (CDN) is at http://www.citrix.com/cdn. This open-enrollment membership program provides access to developer toolkits, technical information, and test programs for software and hardware vendors, system integrators, ICA licensees, and corporate IT developers who incorporate Citrix computing solutions into their products.

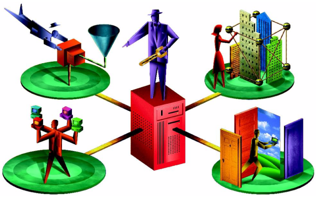
SDKs. The Citrix Server Software Development Kit (SDK) is available for free from www.citrix.com/cdn. Most of the operations that you can perform through Citrix GUI tools can be scripted by using the Citrix Server SDK. The SDK also lets programmers customize most aspects of MetaFrame XP.

Providing Feedback About this Guide

We strive to provide accurate, clear, complete, and usable documentation for Citrix products. If you have any comments, corrections, or suggestions for improving our documentation, we want to hear from you.

You can send email to the documentation authors at documentation@citrix.com. Please include the product name, version number, and the title of the document in your message.

Introduction to the Citrix ICA Java Client



Overview

The ICA Java Client is a Java applet that provides access to applications running on a MetaFrame server farm from any computer device with a standard Web browser. The applet is a download-and-run, zero-install client, optimized for use in Web environments where it is not possible or desirable to install software on the client device.

New Features in this Release

The following list outlines all the new features in this release. Subsequent sections provide further details of individual features where necessary.

- Seamless support
- Content redirection
- Enhanced content publishing
- Reconnection to arbitrarily sized sessions
- Enhanced XML error messages to provide better information for diagnostic troubleshooting
- Smaller SSL libraries, which reduces download time
- Mouse wheel support when using any J2SE (Java 2 Platform, Standard Edition) Version 1.4 environment
- Significant performance enhancements to client drive mapping and printer mapping

Note NFuse Classic and Enterprise Services for NFuse have been integrated as features in MetaFrame XP. They are now called the Web Interface for MetaFrame XP and the Web Interface Extension for Metaframe XP respectively.

Seamless Support

Seamless support has three main aspects:

- Seamless windows
- Session sharing
- Connection Center, a tool that enables users to manipulate both seamless and non-seamless ICA connections

Seamless Windows

Seamless windows means that each remote application appears in a separate resizable window on the client desktop. Users can resize the application window, minimize it, and copy and paste text between published applications and applications running locally on the client device. Copy/paste also works for non-text objects when used between applications sharing an ICA session.

To use seamless windows, Citrix recommends deploying the client through the Web Interface, because this provides the most effective interface for the features provided. You can deploy the client using the sample HTML pages provided with the client package, but this requires more work on your part.

Note Seamless windows are not supported in Netscape 4.x on any UNIX/Linux platform, or on Mac OS 9 or Mac OS X. If the Java Client is configured for seamless mode and run on these platforms, a non-seamless session is launched.

Session Sharing

Session sharing allows seamless application launches to share a single ICA connection rather than creating a new connection for each application. This reduces the system overhead and therefore improves response times for users who have several applications open at the same time. Applications launched in existing sessions also launch more quickly, because a new connection and associated resources do not need to be created.

Connection Center

Connection Center enables users to:

- Disconnect a MetaFrame session
- Switch between full screen and seamless mode
- View properties such as the ICA encryption setting and the user name
- Log off a server session
- Close a published application
- Configure ICA Java Client settings

Server-to-Client Content Redirection

The server can now open URLs in a client-side browser. For example, when you are running Microsoft Outlook in an ICA session, if you click a URL link in an email message, the URL is sent by default to the Web browser on the client's local device. This offers users various benefits: for example, pages open more quickly, and users have the opportunity to use their preferred browser.

No client-side settings are needed to implement this feature; it is enabled by default on the server. However, if you want to disable content redirection, add the following line to the HTML file:

```
<param name="CREnabled" value ="no">
```

Enhanced Content Publishing

You no longer need to download published content to the client device and open it with a local application. Instead, you can associate and open published content with a published application on the MetaFrame server. This enables users to access published content from client devices that have just a Web browser and no other applications installed.

This feature is available only if you are using the Web Interface.

Reconnection to Arbitrarily Sized Sessions

In previous releases of the Java Client, users were unable to reconnect to a session if the size of the applet panel was changed in between the user disconnecting and attempting to reconnect. For example, the user might select a different screen size in the Web Interface, or you might change the applet dimensions on the HTML page.

At this release, users can successfully reconnect to sessions even if the applet panel size is changed.

If the applet panel size is larger than the session, the session is centered within the original panel.

If the applet panel size is smaller than the session, the session is displayed with scroll bars to allow the user access to the entire content. To display the session in a separate window of the new increased size, click **Frame**. When the session is displayed in the separate window click **Embed** to display it in the applet panel again.

To bring the session to the front of the screen at any time, click **Raise**.

Existing Features

The ICA Java Client has the following features:

- The **ICA Java Client Settings** dialog box and status bar. This enables users to configure their own settings using a pop-up dialog box.
- Support for a range of security technologies, including proxy servers, the Secure Gateway for MetaFrame, SSL encryption, TLS encryption, and ICA encryption.
- Automatic reconnection when a session ends (see "Auto-reconnect and Session Termination" on page 38).
- Ability to pass parameters to published applications.
- Video support
    - Resolutions up to 65536 x 65536
    - 256 color to 24-bit (16.7 million colors)
- Client clipboard mapping.
- Client drive, printer, and audio mapping.
- Data compression to reduce the amount of data that needs to be transferred over the network.
- Data caching to improve performance on bandwidth-limited connections.
- SpeedScreen Latency Reduction to improve performance over high latency connections by providing instant feedback to the user in response to typed data or mouse clicks.
- Business recovery to provide consistent connections to published applications in the event of a master ICA browser server disruption.
- Hotkeys to use as substitutes for the standard Windows hotkeys for a published application.

- Shadowing. From MetaFrame XP Feature Release 2, user-to-user shadowing is available as well as administrator-driven shadowing, providing the MetaFrame XP server is configured appropriately. For further information about shadowing, see the MetaFrame XP documentation.

Introduction to the ICA Java Client

The ICA Java Client is a signed Java applet that allows users to connect to MetaFrame servers.

The ICA Java Client has the following advantages:

- **Zero client install**. You do not need to install any software on any client device. Users require only a Java-compatible Web browser. Setup is transparent and automatic.
- **Minimal download size**. At the most basic level of functionality, the applet is around 329KB in size, providing a faster download than any other ICA Client.
- **Platform-independent**. The ICA Java Client can run on any client hardware running a Java-compatible Web browser.

The ICA Java Client applet resides on a Web server and is deployed using an HTML page with the ICA Java Client applet tag. Users run the ICA Java Client by visiting the HTML page using a browser that has Java support. When users request the page, the Java applet is automatically downloaded to the client device. The applet then runs and connects to the server or published application specified in the applet tag.

Unlike the ActiveX, Netscape Plug-in, or Win32 Web Clients, which are downloaded once and then saved for future use by client systems, the Java applet is not stored permanently by the client system. However, most browsers cache Java applets so that they do not need to be downloaded each time they are used. J2SE Plug-in environments also cache the Java Client.

ICA Java Client Requirements

To run the ICA Java Client, the client system must have the following:

- A Web browser with Java Runtime Environment 1.1, or Java 2, Standard Edition Version 1.3 or greater, configured to accept signed Java applets. For more information about signed applets, see “Using Signed Java Applets” on page 28.
- TCP/IP (Dial-up or LAN).
- Mouse.

- Keyboard.
- Access to a Web server that will store the ICA Java Client files.

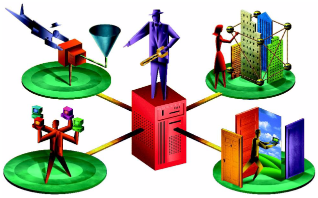
Java Environments

A large number of Java-enabled environments are available, and their functionality varies from platform to platform. To validate proper functionality of the ICA Java Client, Citrix selects a representative group of platforms for testing.

Java 2, Standard Edition Version 1.2 is not recommended for use with the ICA Java Client. Upgrade to a Web browser with Java 2, Standard Edition Version 1.3 or later, or upgrade the JVM in your Web browser to Java 2, Standard Edition Version 1.3 or later. See http://java.sun.com/ for more information about JVM upgrades.

Note If you are running the Java Client on Windows 98, upgrade the JVM to avoid display issues.

Deploying the Citrix ICA Java Client



This chapter describes a basic deployment of the ICA Java Client applet, followed by details of how to customize the Java applet and how to pass parameters. When deploying the ICA Java Client, you can control almost any aspect of the client configuration, as well as controlling what functionality will be available to the user and how large the download will be.

You can also deploy the ICA Java Client with the Web Interface, which automates many of the tasks described in this chapter. The Web Interface is described in the *Web Interface for MetaFrame XP Administrator's Guide*.

Before You Begin

To deploy the ICA Java Client, you need:

- A copy of the client package. You can download the package from http://www.citrix.com/download or copy it in uncompressed form from the Components CD. Citrix recommends that you obtain the latest version of the client from the Web site.

  On the Web site, the ICA Java Client package is available in two formats:

  - .zip, primarily for Windows systems
  - .tar.gz, primarily for UNIX systems

  Both contain identical contents.
- A third-party application to uncompress and unpack the .zip or .tar.gz package, if you downloaded this from the Web site. If you are copying files from the CD you do not need to uncompress them.
- Administrator access to a Web server.

Unpacking the ICA Java Client

This section describes the contents of the ICA Java Client package.

To unpack the ICA Java Client

1. Copy the ICA Java Client package to a suitable location on the Web server. For Microsoft IIS servers, copy the package to a folder in the Web Root directory. For UNIX systems, consult the Web server documentation.
2. If you downloaded the compressed package from the Web site, extract the program files from the .zip or .tar.gz package using a suitable decompression utility.

A number of files are created on the Web server. Some of these files are the Java archives that make up the ICA Java Client applet. There are three types of archive package, as follows:

Archive type	Description
*M.cab files	Authenticode signed cabinet files. Compatible with: Internet Explorer 4.x or later on Windows systems with the Microsoft VM
*N.jar files	Netscape Object signed Java archives. Compatible with: Netscape 4.x or later (all platforms apart from Mac OS 9.x) Netscape 6.x/7.x (all platforms apart from Mac OS 9.x or Mac OS X) Internet Explorer on Windows platforms with Java Plug-in[1] 1.3.1 or later
*J.jar files	Sun Javasoft signed Java archives. Compatible with: Internet Explorer on Mac OS 9.x Internet Explorer on Mac OS X Netscape 4.76 or later on Mac OS 9.x with MRJ Plug-in Netscape 6.x/7.x (all platforms) Internet Explorer on Windows platforms with Java Plug-in[1] 1.3.1 or later. However, Citrix does not recommend using the Java Plug-in with *J.jar files, because some versions of the Java Plug-in do not work with private CA-signed archives.
	[1] Download the Java Plug-in from http://java.sun.com.

For each type of signed archive package there are a number of different components. For Microsoft Authenticode signed Java archives, the components are named as follows.

Archive File	Approximate Size[1]	Description
JICAEngM.cab	520KB	Complete archive. This archive contains the contents of all of the other archives apart from cryptojM.cab and sslM.cab, which must be included if required.
JICA-coreM.cab	319KB	Core archive. This archive provides only a basic connection. You add functionality by using it in conjunction with the other component archives described below.
cryptojM.cab	115KB	Encryption component. Required for all encryption.
sslM.cab	115KB	SSL component. Adds SSL and TLS encryption support.
JICA-configM.cab	57KB	User configuration component. Adds support for the Status bar, buttons, and **ICA Java Client Settings** dialog box.
JICA-audioM.cab	12KB	Audio component. Adds client audio mapping.
JICA-cdmM.cab	21KB	CDM component. Adds client drive mapping.
JICA-clipboardM.cab	11KB	Clipboard component. Adds client clipboard support.
JICA-printerM.cab	14KB	Printer component. Adds client printer mapping.
JICA-sicaM.cab	15KB	ICA encryption component. Adds ICA encryption support.
JICA-tw1M.cab	35KB	ThinWire 1 component. Required for connections to MetaFrame 1.8 without any Feature Releases installed and activated.
JICA-zlcM.cab	66KB	SpeedScreen Latency Reduction component. Adds support for local text echo and mouse feedback.
JICA-seamlessM.cab	56KB	Seamless and Connection Center components. Adds support for seamless windows and Connection Center.

[1] Sizes of other types of archive package vary from those listed here.

If you are not using Microsoft Authenticode signed Java archives, choose the appropriate archive for the JVM you are using. For example, to deploy the core archive for Netscape, choose the Netscape Object signed archive, JICA-coreN.jar.

Getting Started with the Sample HTML Files

This section describes how to use the sample HTML pages supplied in the package to create an HTML page that specifies the correct archives for the user’s browser.

First the sample HTML pages are described, then instructions for customizing them are provided. If you intend to use seamless windows and Connection Center, read the instructions on how to edit seamless1.html, but also keep the desktop.html instructions close for reference. If you do not want to use this functionality, read the instructions for editing desktop.html.

To access the Java Client Connection Center, session sharing, and seamless windows functionality, Citrix strongly recommends that you use the Web Interface or Web Interface Extension that ships with Feature Release 3. Both these products automate the steps that you otherwise need to implement yourself.

Seven sample HTML files are supplied in the client package:

- **index.html**. This page contains links to and descriptions of the six launching pages:
    - desktop.html, application.html, and autoproxy.html. If you do not want to implement seamless windows and Connection Center, use these pages.
    - seamless1.html, seamless2.html, and seamless3.html. To implement seamless windows and Connection Center, use these pages.
- **desktop.html**. This page launches a desktop session to a MetaFrame server. To make a connection with this page, you need to specify an address for the server.
- **application.html**. This page launches a connection, with 128-bit encryption enabled, to a published application. You need to specify the name of the published application and the name of a MetaFrame server to use for server location.
- **autoproxy.html**. This page launches a connection to a published application through a proxy server, using auto client proxy detection. You need to specify the name of the published application and the name of a MetaFrame server to use for server location. For further details of server location, see “Configuring Network Protocol and Server Location” on page 32.

- **seamless1.html, seamless2.html, and seamless3.html.** These pages start remote applications using Connection Center and seamless windows. The only difference between the three files is that they each start a different application. The applications are launched using an existing ICA session when possible. When session sharing is not possible, a new ICA session is created. You need to specify the name of a published application and the name of a MetaFrame server to use for server location.

Editing desktop.html

Desktop.html contains the following applet tag:

```
<applet name=”javaclient”
    codebase=”../”
    code=com.citrix.JICA
    archive=”JICA-coreN.jar,JICA-configN.jar”
    width=640
    height=480>
    <param name=”cabinets” value=”JICA-coreM.cab,JICA-configM.cab”>
    <param name=”Address” value=”avalon”>
    <param name=”End” value=”end.html”>
</applet>
```

The applet tag is used to configure the ICA Java Client. Some parameters are specified inside the <applet> tag as follows:

- Applet name. This is an optional, unique name for the applet. Use this name to refer to the applet when writing scripts. In desktop.html, the applet name **javaclient** is used by a script that displays a warning message if the user tries to close the Web browser window when an ICA session is running. It is also needed for PAC file support.
- Codebase. The path from the HTML page to the ICA Java Client archives. Change this path if it is not correct for your deployment.
- Code. The name of the class file that is executed. For the ICA Java Client without Connection Center, this is always com.citrix.JICA.
- Archive. Specify Netscape Object signed or JavaSoft signed archives here. Specify Authenticode (Internet Explorer) signed archives in a <param> tag. In desktop.html, a Netscape Object signed core archive is specified here and Authenticode signed archives are specified in the <param> tag so that both Netscape and Internet Explorer are supported. If you specify multiple archives in the same place, separate them with a comma.
- Width. The width of the applet window, in pixels.
- Height. The height of the applet window, in pixels.

All other parameters are specified using <param> tags, located between the <applet> and </applet> tags. Use the <param> tags in the form:

```
<param name="parametername" value="valuename">
```

where *parametername* is the name of the parameter you are specifying and *valuename* is the value you are allocating to it.

Note You cannot combine Javasoft and Netscape Object signed archives in the same <applet> tag. To specify more than one type of .jar archive in the same HTML page, you must use a Web browser identification script.

▸ To customize and use desktop.html

1. Open desktop.html in a plain text editor and find the <applet> tag section:

   ```
   <applet name="javaclient"
    codebase="../"
    code=com.citrix.JICA
    archive="JICA-coreN.jar,JICA-configN.jar"
    width=640
    height=480>
    <param name="cabinets" value="JICA-coreM.cab,JICA-configM.cab">
    <param name="Address" value="avalon">
    <param name="End" value="end.html">
   </applet>
   ```

   This is the section that launches the ICA Java Client.

2. Change the Address value of **avalon** to the address of a MetaFrame server on your local network. If you are connecting to a published application, change the address to the name of the published application and add the InitialProgram parameter. For example, if the published application is called **Notepad**, use the following parameters.

   ```
   <param name="Address" value="Notepad">
   <param name="InitialProgram” value="#Notepad">
   ```

3. Change the relative path specified for the codebase if it is not correct for your deployment.

4. Publish the sample HTML pages using your Web server. See the Web server documentation for more information about how to do this.

5. Open a Web browser and open the URL for the sample HTML pages. The index.html page opens.

6. Click the **Minimal Desktop** link. The ICA Java Client applet appears.

7. To connect to the MetaFrame server, click **Connect** or **Click to connect**. To configure the client using the **ICA Java Client Settings** dialog box, click **Settings**.

You can edit application.html and autoproxy.html in the same way. The additional parameters used in these examples are described in Chapter 4.

To change the warning message displayed when a user tries to close an active ICA session

If you try to close the Web browser window when using an ICA session created with one of the example Web pages, a warning message appears. The message is defined in this section of the HTML page:

```
function onBeforeUnload() {
    var connected = document.javaclientname.isConnected();
    if (connected) {
        alerted = true;
        return "Closing this window will disconnect your ICA session";
        }
    }
```

where *javaclientname* is the name of the applet.

To change the message displayed, edit the text in the HTML page.

Editing seamless1.html

Seamless1.html, contains the following applet tag:

```
<applet name="javaclient"
    code="com.citrix.ConnectionCenter"
    codebase="../"
    archive="JICA-coreN.jar,JICA-configN.jar,JICA-seamlessN.jar"
    width="400"
    height="400">
    <param name="cabinets" value=
        "JICA-coreM.cab,JICA-configM.cab,JICA-seamlessM.cab">
    <param name="Address" value="Notepad">
    <param name="InitialProgram" value="#Notepad">
    <param name="HTTPBrowserAddress" value="plateau">
    <param name="TWIMode" value="on">
</applet>
```

The applet tag is similar to that used in desktop.html (see “Editing desktop.html” on page 23), but note the following differences:

- Code. The name of the class file that is executed. For the ICA Java Client with Connection Center, this is always com.citrix.ConnectionCenter.
- Archive. The JICA-seamless archive is required for Connection Center and seamless windows.
- Width and Height. These are both set to 400 pixels, which are appropriate dimensions for the Connection Center user interface.

- TWIMode. This parameter enables seamless windows. Seamless windows are required for session sharing. If you use Connection Center without seamless windows there is no session sharing, in other words each application is launched in a separate ICA connection and you gain no reduction in system resource overhead.

You can customize the sample HTML files and specify additional HTML files to launch your own published applications. If you have many applications, it may be easier to provide users with HTML links that all reference a server-side script that generates the applet tag as needed, based on the selected link.

To customize and use seamless1.html

1. Open seamless1.html in a plain text editor and find the <applet> tag section:

```
<applet name="javaclient"
 code="com.citrix.ConnectionCenter"
 codebase="../"
 archive="JICA-coreN.jar,JICA-configN.jar,JICA-seamlessN.jar"
 width="400"
 height="400">
 <param name="cabinets" value="JICA-coreM.cab,
         JICA-configM.cab,JICA-seamlessM.cab">
 <param name="Address" value="Notepad">
 <param name="InitialProgram" value="#Notepad">
 <param name="HTTPBrowserAddress" value="plateau">
 <param name="TWIMode" value="on">
</applet>
```

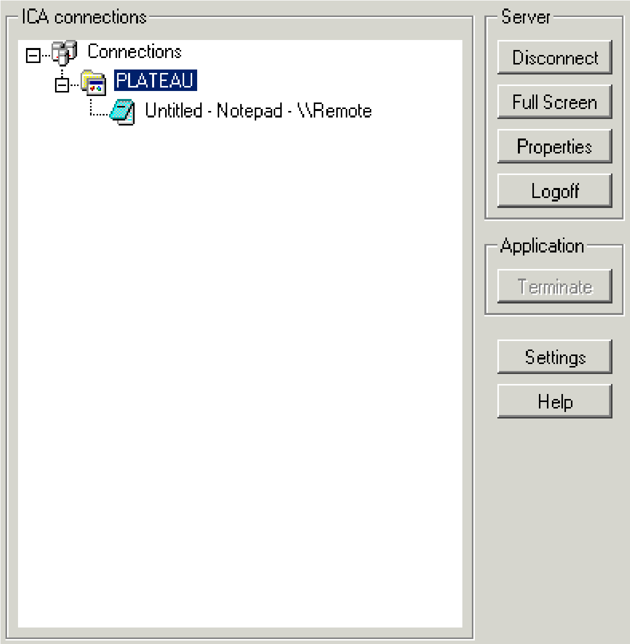
   This is the section that launches the ICA Java Client.

2. Change the value of the Address parameter to the name of the published application to which you want to connect.
3. Change the value of the InitialProgram parameter to the application name specified in Address, preceded by a pound sign or hash mark (#). For example, if the published application is called **Word**, use the following parameters:

```
<param name="Address" value="Word">
<param name="InitialProgram" value="#Word">
```

4. Change the relative path specified for the codebase if it is not correct for your deployment.
5. Change the value of the HTTPBrowserAddress parameter to the address of the MetaFrame server used for HTTP browsing.
6. Publish the sample HTML pages using your Web server. See your Web server documentation for more information about how to do this.
7. Open a Web browser and open the URL for the sample HTML pages. The index.html page opens.

8. Click the **Launch seamless application 1** link. The first time in each Web browser session that you select a seamless link, the Connection Center applet starts up, launches an ICA connection to the specified application, and displays it in a separate window. The illustration below shows the Connection Center window:



9. If you select another seamless link while this session is open, the Connection Center applet starts up again. However, this new applet is displayed as a progress indicator rather than a duplicate Connection Center, and it closes itself when the new application is opened. In this way the launching and management of all applications is centralized by the initial Connection Center instance, which performs session sharing when possible.

The tree in the Connection Center window displays the current connections, with desktop connections identified by 🖥 and seamless ones by 🗔 . Below a seamless connection node are nodes representing windows opened in that connection.

Double-clicking a window node brings that window to the front. If the window is minimized, double-clicking the node has no effect.

Right-clicking a window node displays the standard **System** menu.

Closing the Connection Center window disconnects all connected sessions, after prompting the user for confirmation.

Using Signed Java Applets

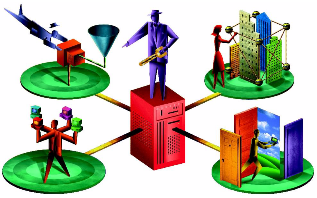
Due to network security restrictions imposed by Java, many Web browsers do not permit users to connect to other computers on the network when using Java applets.

When a Java applet attempts to make a connection to the server specified in the HTML page, the Java security manager detects the attempt to connect to another computer and cancels the operation. The result of this security restriction is that, under normal conditions, a client system can connect to a server only if the server is also the same device as the Web server that contains the applet class files.

To overcome this restriction, the ICA Java Client uses signed archives. The signature confirms that the files being downloaded came from Citrix and have not been altered since the signature was applied. You must ensure that users' Web browsers are configured to accept signed Java applets.

When attempting a connection to the MetaFrame server, the user is prompted with the Citrix signed certificate. When the user accepts the signature, the connection is permitted.

Configuring the Citrix ICA Java Client



This chapter describes how to configure the ICA Java Client using an HTML page. Topics covered include:

- Initial configuration
  - Specifying the language for displaying the user interface
  - Configuring network protocol and server location
  - Changing the client name
  - Passing parameters to published applications
- User interface configuration
  - Setting the number of colors used in the ICA session window
  - Showing or hiding the status bar and settings button
  - Controlling client behavior at the start and end of sessions
  - Specifying what type of keyboard to use
  - Configuring hotkeys
- Client device mapping
  - Client drive mapping
  - Client printer mapping
  - Client audio mapping

- Integrating the ICA Java Client with security solutions
  - Proxy servers
  - The Secure Gateway for MetaFrame and SSL Relay
  - Firewalls
  - ICA encryption
- Advanced configuration options

How to Configure the ICA Java Client

The ICA Java Client can be configured using:

- An HTML page, as described in this chapter.
- The **ICA Java Client Settings** dialog box.

  To display the **ICA Java Client Settings** dialog box, click the **Settings** button. In seamless mode, the **Settings** button is on the **Connection Center** dialog box. If the client is not in seamless mode, the **Settings** button is on the status bar. To get help on the tasks you can carry out using the dialog box, click **Help**. All the features that can be configured from the dialog box are also described in detail in this chapter.
- The Web Interface for MetaFrame XP.

  The Web Interface automatically generates the necessary Web pages to launch the ICA Java Client. See the *Web Interface for MetaFrame XP Administrator's Guide* for details of how to use the Web Interface to configure the Java Client.

Initial Configuration

Starting the ICA Java Client in a Specific National Language

The ICA Java Client allows you to specify what language to use to display the user interface. By default a session uses the language specified for the client device to display the user interface. If you specify a language code that is not recognized or not supported, English is used.

The following languages are supported by the ICA Java Client:

- English
- French
- German
- Spanish
- Japanese

To set the ICA Java Client language

Specify the following parameter in the HTML page:

```
<param name=”Language” value=”yourlanguage”>
```

Where *yourlanguage* is the two letter abbreviation for the language you want to use. The standard two letter abbreviations are as follows:

en - English

fr - French

de - German

es - Spanish

ja - Japanese

For example, to use Japanese as the language on a non-Japanese device when connecting to the MetaFrame server named CitrixServer, create an applet tag like the following:

```
<applet code=com.citrix.JICA
    archive=JICAEngN.jar
    width=1024 height=768>
    <param name=”cabinets” value=”JICAEngM.cab”>
    <param name=”Address” value=”CitrixServer”>
    <param name=”Language” value=”ja”>
</applet>
```

If you use languages other than English, ensure that your Web server sends HTML files with the correct Content-Type to avoid possible corruption of the applet parameter strings. Configuration details depend on the server software in use.

If you cannot configure your Web server correctly, indicate your character set in your HTML files. For example:

```
<HEAD>
    ...
    <META HTTP-EQUIV=”Content-Type” CONTENT=”text/html;charset={your
                                                 character set}”>
    ...
</HEAD
```

This second method does not work with all browsers, and may fail with certain proxy Web servers.

When troubleshooting suspected problems with parameter string encoding, it can be useful to copy the strings outside the applet tag and check that they display correctly in the Web browser.

Configuring Network Protocol and Server Location

This section explains how to:

- Set the network protocol used by the ICA Java Client
- Control the way in which the ICA Java Client locates MetaFrame servers
- Configure a list of MetaFrame servers for business recovery

Setting the Network Protocol

The network protocol setting allows you to control the way the ICA Java Client searches for MetaFrame servers and how it communicates with them.

The protocols are:

- TCP/IP + HTTP - The client uses the HTTP protocol to search for MetaFrame servers. The client communicates with the MetaFrame server using ICA protocol over TCP/IP. This is the default protocol.
- SSL/TLS + HTTPS - The client uses the HTTPS protocol to search for a list of MetaFrame servers. The client communicates with the MetaFrame server using the SSL or TLS protocols. These protocols are described in detail in “Integrating the ICA Java Client with the Secure Gateway for MetaFrame or SSL Relay” on page 53.

- TCP/IP - The client uses the UDP protocol to search for MetaFrame servers. The client communicates with the MetaFrame server using ICA protocol over TCP/IP. Select this option if all the MetaFrame servers and clients are located on the same network and a secure connection is not required.

▸ **To change the network protocol to TCP/IP (UDP)**

Add the following parameter to the HTML page:

```
<param name=”BrowserProtocol” value=”UDP”>
```

▸ **To change the network protocol to SSL/TLS+HTTPS**

Add the following parameters to the HTML page:

```
<param name=”BrowserProtocol” value=”HTTPonTCP”>
<param name= “SSLEnable” value=”on”>
```

Server Location

Server location (also called server browsing) provides a method for a network-connected ICA Client to connect to a published application. The way in which server location works depends on which network protocol has been configured:

- TCP/IP+HTTP and SSL/TLS+HTTPS - The default server address is **ica** or **ica.<*current DNS domain*>** depending on the behavior of your JVM. You must set specific server addresses for MetaFrame servers. The ICA Java Client uses the HTTP or HTTPS protocol respectively to contact the servers.
- TCP/IP - The default setting for server location is **auto-locate**. The ICA Java Client attempts to contact all the servers on the network by broadcasting on the UDP protocol. Alternatively, you can set specific addresses for MetaFrame servers.

Example

In the following example, the HTTPBrowserAddress parameter is specified to be the MetaFrame server **Wizard**. This device is responsible for resolving the name of the published application **Notepad** to the address of a MetaFrame server that can host this application.

```
<html>
    <body>
        <applet code=com.citrix.JICA
            archive=JICAEngN.jar
            width=640 height=480>
            <param name=cabinets value=JICAEngM.cab>
            <param name=BrowserProtocol value=”HTTPonTCP”>
            <param name=HTTPBrowserAddress value=”Wizard”>
            <param name=Address value=”Notepad”>
            <param name=InitialProgram value=”#Notepad”>
        </applet>
    </body>
</html>
```

Business Recovery

Business recovery provides consistent connections to published applications in the event of a master ICA browser server disruption. You can define up to three groups of MetaFrame servers to which you want to connect: a primary and two backups. Each group can contain from one to five servers.

▸ **To specify a business recovery server group using the TCP/IP+HTTP or SSL/TLS+HTTPS protocols**

Use the following parameters to specify server groups:

- **HTTPBrowserAddress to HTTPBrowserAddress5**: Specify the primary group of servers
- **HTTPBrowserAddress6 to HTTPBrowserAddress10**: Specify the first backup group of servers
- **HTTPBrowserAddress11 to HTTPBrowserAddress15**: Specify the second backup group of servers

Fill in any unused server addresses with five dashes (-----). These dashes are required to fill in any gaps in the list but are not required at the end of the list.

▸ **To specify a business recovery server group using the TCP/IP protocol**

Use the following parameters to specify server groups:

- **TCPBrowserAddress to TCPBrowserAddress5**: Specify the primary group of servers

- **TCPBrowserAddress6 to TCPBrowserAddress10**: Specify the first backup group of servers
- **TCPBrowserAddress11 to TCPBrowserAddress15**: Specify the second backup group of servers

Fill in any unused server addresses with five dashes (-----). These dashes are required to fill in any gaps in the list but are not required at the end of the list.

Example

In the following example, the primary group of servers contains **Arthur**, **Morgana**, and **Merlin**. The first backup group contains the servers **Excalibur** and **Stone**. There is no secondary backup group.

```
<html>
    <body>
        <applet code=com.citrix.JICA
            archive=JICAEngN.jar
            width=640 height=480>
            <param name=”cabinets” value=”JICAEngM.cab”>
            <param name=”BrowserProtocol” value=”HTTPonTCP”>
            <param name=”HTTPBrowserAddress” value=”Arthur”>
            <param name=”HTTPBrowserAddress2” value=”Morgana”>
            <param name=”HTTPBrowserAddress3” value=”Merlin”>
            <param name=”HTTPBrowserAddress4” value=”-----”>
            <param name=”HTTPBrowserAddress5” value=”-----”>
            <param name=”HTTPBrowserAddress6” value=”Excalibur”>
            <param name=”HTTPBrowserAddress7” value=”Stone”>
        </applet>
    </body>
</html>
```

Changing the Client Name

The *client name* is used to identify the ICA Client to the MetaFrame server, and is also used to name auto-created printers. See “Client Printer Mapping” on page 45 for more information about auto-created printers. You may need to change the client name if you are trying to access resources shared with another ICA Client with the same client name.

By default, the ICA Java Client uses the value for the client device’s host name (if it exists and is not set to **localhost**) as the client name reported to the MetaFrame server. If the ICA Java Client cannot use the client device’s host name, it uses **AnonJava** as the client name. The client name sent to the MetaFrame server is always truncated to 20 characters.

You can change the client name on the HTML page or by using the **ICA Java Client Settings** dialog box.

To change the client name

To change the client name, you need to add two parameters on the HTML page:

```
<param name=”client.wfclient.UseHostname” value=”off”>
<param name=”client.wfclient.Clientname” value=”yourclient”>
```

where *yourclient* is the client name you want to use.

Passing Parameters to Applications

When connecting to a published application, you specify the application name using the InitialProgram parameter on the HTML page. For example, to connect to an application called **Notepad**, specify the following parameters:

```
<param name=”Address” value=”Notepad”>
<param name=”InitialProgram” value=”#Notepad”>
```

You can also specify a command line parameter that the MetaFrame server will pass to a published application when it runs that application. For example, if you provide a file name parameter to Notepad, Notepad will start up with that file loaded. The parameters that an application will honor are built into the application and do not have to be file names.

To pass a parameter to an application

Specify the following parameters on the HTML page:

```
<param name=”Address” value=”application”>
<param name=”InitialProgram” value=”#application”>
<param name=”Param” value=”parameter”>
```

where *application* is the name of the published application and *parameter* is the parameter. You can specify a maximum value length of 256 characters for these parameters. For example, to open a file called M:\new.txt in Notepad, specify the following parameters:

```
<param name=”Address” value=”Notepad”>
<param name=”InitialProgram” value=”Notepad”>
<param name=”Param” value=”M:\new.txt”>
```

Note For parameter passing to work, you must configure the published application to receive parameters by appending %* to the published application's command line, for example:

```
notepad %*
```

Full details of how to publish applications and set up commands are in the documentation included in the MetaFrame server package.

File name parameters are interpreted by the remote application relative to the server’s file system. If you want to pass a client-side file to a remote published application, you must use client drive mapping (see “Client Drive Mapping” on page 44).

User Interface Configuration

Window Properties

You set the size of the window that ICA sessions run in using the applet tag. See “Editing desktop.html” on page 23 for more information. The number of colors used in the session window is defined with a parameter on the HTML page.

▸ **To set the number of colors used for the ICA session window**

Specify the following parameter in the HTML page

```
<param name=”DesiredColor” value=”2|4|8”>
```

where 2 specifies 256 colors, 4 specifies thousands of colors, and 8 specifies millions of colors. The ICA Java Client cannot be configured to use only 16 colors but can display applications published in 16 color mode; these are run in 256 colors.

Status Bar and Settings Button

The status bar displays status information about the client. When the user clicks the **Settings** button, the **ICA Java Client Settings** dialog box appears.

You can display or hide the status bar and **Settings** button using parameters on the HTML page. Both are displayed by default. However, if you do not want users to make configuration changes, you may decide to hide the **Settings** button. If you decide that maximum screen real estate is a priority, you may decide to hide the status bar.

When the client is in seamless mode, the status bar is not visible and the user accesses the **Setting**s dialog box from the Connection Center.

▸ **To hide the status bar**

Specify the following parameter on the HTML page:

```
<param name=”ShowStatusBar” value=”no”>
```

▸ **To hide the Settings button**

Specify the following parameter on the HTML page:

```
<param name=”ShowSettingsButton” value=”no”>
```

Note To display the **Settings** button and status bar when using the core archives, you must include the JICA-config archive. This functionality is included in the complete archives.

Auto-reconnect and Session Termination

Note The information in this section does not apply if you are using Connection Center. Sessions always start automatically and, if the network connection is lost, an attempt is always made to reconnect.

You can control how the client behaves when starting or ending a session, by specifying parameters on the HTML page.

You can configure the client to connect when the user clicks the startup splash screen, or to connect automatically, as the HTML page displays. You can also change what happens when a session ends, either as a result of the user terminating it or because there was a problem.

▸ To change the ICA Java Client startup

Specify the following parameter on the HTML page:

```
<param name=”Start” value=”Manual|Auto”>
```

If you set this parameter to **Manual** (the default), the user must click to connect to a MetaFrame server. If you set it to **Auto**, the message **Connecting to server** appears as the HTML page is displayed and the user is automatically connected to the server.

▸ To change what happens when a session ends

Specify the following parameter on the HTML page:

```
<param name=”End” value=”Manual|Auto|Terminate|URL”>
```

where:

- **Manual -** displays the startup splash screen when the session ends, and the message **Click to reconnect**. To reconnect, the user can click anywhere on the splash screen.
- **Auto -** displays the **Reconnecting** dialog box when the session ends for any reason. The number in the dialog box counts down to 0 and the client reconnects.
- **Terminate -** displays either **Connection Terminated** or **Connection Error** when the session ends, depending on whether the user chose to end the session or whether there was a problem that caused the session to end.

- ***URL*** - displays the splash screen and redirects in two seconds to the specified URL. You can specify the URL of any Web page.

The HTML examples supplied with the client include the End parameter. The value specified is a URL to a page called end.html. The applet tag section of each of the examples has this parameter:

```
<param name="End" value="end.html">
```

When you end the session, the client redirects to end.html, which contains a script to close the browser window. You can edit end.html to display anything you want.

▶ **To change the time-out period for automatic reconnection**

The default time-out period is five seconds. To change the period, specify the following parameter in the HTML page:

```
<param name=”ReconnectDelay” value=”delay”>
```

where *delay* is the delay in seconds. Specifying this parameter does not affect the delay before connection to an HTML page if you specified a URL for the End parameter.

Note If the Start and End parameters are both set to **Auto**, the startup splash screen is displayed and you must click on it to connect.

Keyboard and Mouse

The ICA Java Client lets you specify what type of keyboard to use in ICA sessions. By default, if you do not specify a keyboard preference, the session uses the keyboard that is specified as the default keyboard for the user’s MetaFrame server.

You can specify the keyboard type on the HTML page or the user can do it using the **ICA Java Client Settings** dialog box.

Note Different JVMs interpret key detection differently. If you experience problems with unexpected keyboard behavior, consider upgrading or changing your JVM and/or Web browser.

The ICA Java Client supports the use of any keyboard supported by the MetaFrame server to which the user is connecting.

▶ **To specify a keyboard other than the server’s default**

Specify the following parameter on the HTML Page:

```
<param name=”user.wfclient.keyboardlayout” value=”layout”>
```

where *layout* is a value from the server’s list of supported keyboards. A list of the supported keyboards is provided in “Supported Keyboards” on page 84.

For example, to specify a Danish keyboard, create an HTML page like the following:

```
<html>
    <body>
        <applet code=com.citrix.JICA
            archive=”JICAEngJ.jar”
            width=640 height=480>
            <param name=”cabinets” value=”JICAEngM.cab”>
            <param name=”Address” value=”CitrixServer”>
            <param name=”user.wfclient.keyboardlayout” value=”Danish”>
        </applet>
    </body>
</html>
```

The ICA Java Client supports the following keyboard types to distinguish between subtypes of the Japanese keyboard layout:

```
"(Default)"
"IBM PC/XT or compatible keyboard"
"101 Keyboard (Japanese)"
"106 Keyboard (Japanese)"
"NEC PC-9800 on PC98-NX (Japanese)"
"NEC PC-9800 on PC98-NX 2 (Japanese)"
"NEC PC-9800 Windows 95 and 98 (Japanese)"
"NEC PC-9800 Windows NT (Japanese)"
"Japanese Keyboard for 106 (Japanese)"
"DEC LK411-JJ Keyboard (Japanese)"
"DEC LK411-AJ Keyboard (Japanese)”
```

Note If you are using a 109 key Japanese keyboard, specify the keyboard type **106 Keyboard (Japanese)**.

To specify a Japanese keyboard type

Specify the following parameter on the HTML page:

```
<param name=”user.wfclient.keyboardtype”
value=”101 Keyboard (Japanese) | 106 Keyboard (Japanese)”>
```

For example, to specify a Japanese 106 key keyboard, create an HTML page like the following:

```
<html>
  <body>
    <applet code=com.citrix.JICA
    archive=”JICAEngN.jar”
    width=640 height=480>
    <param name=”cabinets” value=”JICAEngM.cab”>
    <param name=”Address” value=”CitrixServer”>
    <param name=”user.wfclient.keyboardlayout” value="Japanese">
    <param name=”user.wfclient.keyboardtype” value="106 Keyboard
                                                 (Japanese)">
    </applet>
  </body>
</html>
```

Using Applications that Require a 3-Button Mouse

In the ICA Java Client, the middle button of a 3-button mouse can be emulated by clicking both buttons of a 2-button mouse at the same time.

Hotkeys

Hotkeys are used to control the behavior of the client and as substitutes for the standard Windows hotkeys for a published application. For example, if you want to display the Windows NT Security Desktop on a Windows PC, you press CTRL+ALT+DEL. If you are running the ICA Client on a Windows PC and are working in a MetaFrame session, this key combination opens the Security Desktop on the local device. Hotkey functionality allows you to map common key combinations like CTRL+ALT+DEL to a key combination such as CTRL+F1 that is ignored by your local operating system. When you press this new combination, the ICA Java Client sends CTRL+ALT+DEL to the MetaFrame server, displaying the Windows NT Security Desktop in your session.

You can specify hotkeys on the HTML page or the user can do it by using the **ICA Java Client Settings** dialog box.

ICA Java Client hotkeys use a pair of keys. The first key defines the shift state and the second defines the character. The following hotkeys are available:

Hotkey (Default Mapping)	Description
Hotkey2 (SHIFT+F3)	Close Remote Application. The Close Remote Application hotkey disconnects applications opened in an ICA session. If no programs are open, the session is disconnected after the user is prompted for confirmation.
Hotkey3 (SHIFT+F2)	When the client is running in seamless mode, this hotkey toggles between seamless mode and “windowed” mode. When the client is not in seamless mode, this hotkey is used when a reconnected session is larger than the applet panel; it toggles between embedding the session inside the applet panel with scroll bars and displaying it in a separate window.
Hotkey4 (CTRL+F1)	Substitute for the standard Windows hotkey CTRL+ALT+DEL. The CTRL+ALT+DEL hotkey displays the Windows Security Desktop in the ICA session.
Hotkey5 (CTRL+F2)	Substitute for the standard Windows hotkey CTRL+ESC. On MetaFrame XP servers, the remote Windows **Start** menu appears.
Hotkey6 (ALT+F2)	Substitute for the standard Windows hotkey ALT+ESC. This hotkey is used to bring the focus to maximized and minimized windows of programs that are open in an ICA session, in the order that they were opened.
Hotkey7 (ALT+PLUS)	Substitute for the standard Windows hotkey ALT+TAB. Use this hotkey to cycle through applications that are open in the ICA session. A popup box appears and displays the programs as you cycle through them. The chosen application receives keyboard and mouse focus.
Hotkey8 (ALT+MINUS)	Substitute for the standard Windows hotkey ALT+SHIFT+TAB. Like the ALT+TAB hotkey, this key sequence cycles through applications that are open in the ICA session but in the opposite direction. The chosen application receives keyboard and mouse focus.
Hotkey9 (CTRL+F3)	Substitute for the standard Windows hotkey CTRL+SHIFT+ESC. Use this hotkey to display the task manager.
Hotkey10 (CTRL+F4)	Toggle SpeedScreen Latency Reduction. Toggles mouse click feedback and local text echo on and off.

▸ To change the hotkey sequence from the default

For each hotkey, specify two parameters on the HTML page: one for the shift state and a second for the character state, as follows:

```
<param name=”user.wfclient.hotkey*shift” value=”shiftstate”>
<param name=”user.wfclient.hotkey*char” value=”character”>
```

where `*` is the hotkey number, *shiftstate* is the shift state (CTRL, SHIFT, or ALT), and *character* can be any of the following:

F1, F2, F3, F4, F5, F6, F7, F8, F9, F10, F11, F12, tab, star, plus, minus, escape or (none)

Specifying **(none)** for the character disables the hotkey.

The following example describes how to map the Close Remote Application hotkey to the key sequence CTRL+F1 and the ALT+TAB hotkey key sequence to SHIFT+TAB.

```
<html>
    <body>
        <applet code=com.citrix.JICA
            archive=”JICAEngN.jar”
            width=640 height=480>
            <param name=”cabinets” value=”JICAEngM.cab”>
            <param name=”Address” value=”CitrixServer”>
            <param name=”user.wfclient.hotkey2shift” value=”ctrl”>
            <param name=”user.wfclient.hotkey2char” value=”f1”>
            <param name=”user.wfclient.hotkey7shift” value=”shift”>
            <param name=”user.wfclient.hotkey7char” value=”tab”>
        /applet>
    </body>
</html>
```

Client Device Mapping

The Citrix ICA Java Client supports *client device mapping* for connections to MetaFrame servers. Client device mapping allows a remote application running on the MetaFrame server to access printers and disk drives attached to the local client machine. The applications and system resources appear to the user at the client machine as if they are running locally. Ensure that client device mapping is supported on your MetaFrame server before using these features.

Note Drive mapping and client audio mapping are not supported by MetaFrame for UNIX Operating Systems Version 1.0. A hotfix is available for Version 1.1, Feature Release 1, to support client drive mapping.

This section includes more information about:

- Client drive mapping
- Client printer mapping
- Client audio mapping

Client Drive Mapping

Client drive mapping can make any specified directory on the client machine, including CD-ROMs, available to the user during ICA sessions on MetaFrame servers. When a MetaFrame server is configured to allow client drive mapping, users can access their locally stored files, work with them during their ICA sessions, and then save them again either on a local drive or on a drive on the MetaFrame server.

When drive mapping is configured, the ICA Java Client attempts to use it for all connections. If the server does not support drive mapping, or if the JVM is configured not to allow access to local drives, drive mapping is not available.

Drive mapping can be configured only by the user through the **ICA Java Client Settings** dialog box.You cannot configure client drive mapping on an HTML page because this is a violation of the client's security.

To make client drive mapping available to users when using the core archives, you must include the JICA-cdm archive. If you do not, the **Drive Mapping** tab does not appear in the **ICA Java Client Settings** dialog box.

Configure client drive mapping by making the **ICA Java Client Settings** dialog box available to users the first time they log on and instructing them to configure drive mapping. The **ICA Java Client Settings** dialog box is described in the Help. See "Copying Client Drive Mapping Configuration" on page 61 for information about how to deploy client drive mapping configuration.

Using Mapped Drives

Once configured, mapped drives are transparent and appear the same as other drives on the server. However, due to the way Java accesses file systems, mapped drives cannot:

- **Lock files**. Applications sometimes lock the files that are in use to prevent other applications from accessing them at the same time and possibly corrupting them. However, the ICA Java Client cannot lock local files. To prevent file corruption, warn users not to access the same file with two or more applications at the same time.
- **Use file attributes**. Attributes cannot be set on files on a mapped drive.
- **Set date and time**. The date and time stamps are not set on files created or edited on mapped drives.
- **Report drive capacity and usage**. Users must use the operating system of their local machine to determine the capacity of mapped drives.

Client Printer Mapping

The ICA Java Client can be configured to map client printers to an ICA session so that client system users can print to their local serial printers or network printers.USB printer mapping is not supported. Printers can be created with the ICA Client Printer Configuration tool on the MetaFrame server, or by configuring auto-created printers either on the HTML page or in the **ICA Java Client Settings** dialog box. Configuring auto-created printers using the **ICA Java Client Settings** dialog box is described in the Help.

To make client printer mapping available to users when using the core archives, you must include the JICA-printer archive. If you do not, the **Printer Mapping** tab does not appear in the **ICA Java Client Settings** dialog box.

Auto-Created Printers

With auto-created printers, users find their local printers mapped to their sessions and ready for use as soon as they connect. You can configure auto-created printers on the HTML page, as described in this section, or with the **ICA Java Client Settings** dialog box. Citrix recommends that you use the **ICA Java Client Settings** dialog box for greatest ease of use. When users log on, their auto-created printers appear in Print Manager or in the Printers folder and are ready for use. When they log off, their printer mappings are deleted from the MetaFrame server.

▸ **To configure auto-created printers**

Specify the printer name, the port name, and the driver, using the following parameters in the HTML page:

```
<param name=”user.localclientprinters” value=”printername”>
<param name=”user.printername.port” value=”portname”>
<param name=”user.printername.driver” value=”drivername”>
```

where:

- ***printername* -** Can be any name by which you want to identify the printer.
- ***portname* -** Specifies a file name, port name, or printer IP address (or network name and print queue).

  When printing to a file, the output file is composed of printer machine code. This file can be sent to a printer using a platform-specific utility. For example, use a command prompt on Windows platforms to send the file to a printer by copying the file to a printer port.

  When printing to a port, specify the port. A typical port on Windows is LPT1: On Linux or UNIX systems, the port is similar to /dev/lp0/. Check your operating system documentation for more information.

When printing to a network printer, specify the printer's IP address or network name and print queue (*ipaddress*:*printqueuename* or *networkname*:*printqueuename*).

The ICA Java Client also allows users to print to standard out or standard error. Specify **stdout** or **stderr** to print to standard out or standard error respectively.

- ***drivername* -** Specify the printer driver. This name is case-sensitive and must exactly match the driver name on the MetaFrame server. To check the driver list on servers running MetaFrame XP Server for Windows:

  1. Double-click **My Computer.**
  2. Double-click **Printers**, then double-click **Add Printer.**
  3. In the Add Printer Wizard, be sure My Computer is selected. Click **Next**.
  4. Select a port, such as LPT1:. Click **Next**.

     The list under Printers contains the printer drivers.

You can also make the printer the default printer for a session by specifying the following parameter:

```
<param name="user.printername.comment" value="WFCDefault">
```

where *printername* is the name of the printer. If the MetaFrame XP server is set to connect only to the user's default printer, this sets the auto-created printer to be the default printer.

Note To auto-create a printer, the MetaFrame server must have the correct printer driver installed, as specified in the **ICA Java Client Settings** dialog box or on the HTML page using the Driver parameter. If the correct driver is not installed, the printer is not auto-created. In this case, you must install the correct printer driver on the MetaFrame server.

You can change the list of drivers that appears in the **ICA Java Client Settings** dialog box by editing the ICAPrinters.dat file. See "Changing the List of Printer Drivers" on page 61 for more information.

Examples: Configuring Individual Printers

The following example demonstrates auto-creating a printer by specifying parameters on the HTML page. This HTML page is suitable for use only with a Windows client system. In this example, the printer’s name is Laser. It is connected to the client’s LPT1: port and has a driver named HP LaserJet 4/4M Plus PS.

```
<html>
 <body>
    <applet code=com.citrix.JICA
      archive=JICAEngN.jar
      width=640 height=480>
      <param name=”cabinets” value=”JICAEngM.cab”>
      <param name=”Address” value=”CitrixServer”>
      <param name=”user.localclientprinters” value=”Laser”>
      <param name=”user.Laser.port” value=”lpt1:”>
      <param name=”user.Laser.driver” value="HP LaserJet 4/4M Plus PS">
    </applet>
 </body>
</html>
```

In the following example, the printer’s name is also Laser and has a driver named HP LaserJet 4/4M Plus PS. The printer is a network printer that exists on a network print server with an IP address of 192.168.1.24 and a print queue named FLOOR2_LJ.

```
<html>
 <body>
    <applet code=com.citrix.JICA
      archive=JICAEngN.jar
      width=640 height=480>
      <param name=”Address” value=”CitrixServer”>
      <param name=”user.localclientprinters” value=”Laser”>
      <param name=”user.Laser.port” value=”192.168.1.24:FLOOR2_LJ”>
      <param name=”user.Laser.driver” value="HP LaserJet 4/4M Plus PS">
    </applet>
 </body>
</html>
```

Example: Configuring Multiple Printers

The following example illustrates configuring multiple printers, one of which is a network printer. The port name of a network printer can be a UNC name or the name of a local port that has a network printer mapped to it. In this example, the printer NetPrinter1 is a network printer mapped to the local LPT2 port and it has a driver named HP Laser Jet 400 Series PS.

```
<html>
 <body>
   <applet code=com.citrix.JICA
     archive=JICAEngN.jar
     width=640 height=480>
     <param name=”Address” value=”CitrixServer”>
     <param name=”user.localclientprinters” value=”Laser,NetPrinter1”>
     <param name=”user.Laser.port” value=”lpt1:”>
     <param name=”user.Laser.driver” value="HP LaserJet 4/4M Plus PS">
     <param name=”user.NetPrinter1.port” value=”lpt2:”>
     <param name=”user.NetPrinter1.driver” value=
       ”HP LaserJet 400 Series PS">
   </applet>
 </body>
</html>
```

Printing From MetaFrame Server for UNIX Operating Systems

Before users can print to a client printer from MetaFrame Server for UNIX Operating Systems, printing must be enabled by the MetaFrame administrator. For further information about printing from MetaFrame Server for UNIX Operating Systems, see the *Citrix MetaFrame Server for UNIX Operating Systems Administrator’s Guide,* and the manual pages.

Client Audio Mapping

Client audio mapping enables applications running on the MetaFrame server to play sounds through a sound device installed on the client computer. To make client audio mapping available to users, you must enable it using a parameter on the HTML page.

Starting the ICA Java Client while an audio application is running on your desktop can disable audio mapping. Do not run audio applications while starting the ICA Java Client.

Note Java Plug-in 1.4.1 is required for audio to work on Linux platforms. The ICA Java Client explicitly turns off audio when running on Linux platforms with earlier JVMs because these JVMs can crash or hang on Linux when audio is enabled.

▶ **To enable client audio mapping**

Add the following parameter to the HTML page:

```
<param name=”ClientAudio” value=”on”>
```

You control the amount of bandwidth used by client audio mapping by configuring the ICA Settings on the MetaFrame server.

Note If the MetaFrame server is set to use **Low** quality, client audio mapping for the ICA Java Client is disabled.

Integrating the ICA Java Client with Security Solutions

You can integrate the ICA Java Client with a range of security technologies, including proxy servers, firewalls, and SSL/TLS based systems. This section describes:

- Making a connection through a SOCKS proxy server or Secure proxy server (also known as Security proxy server, HTTPS proxy server, or SSL tunneling proxy server)
- Integrating the client with the Secure Gateway for MetaFrame or SSL Relay solutions with Secure Sockets Layer (SSL) and Transport Layer Security (TLS) protocols
- Connecting to a server across a firewall using alternate addressing
- Using ICA encryption to help protect data on local networks

Connecting Through a Proxy Server

Proxy servers are used to limit access into and out of your network, and to handle connections between ICA Clients and the MetaFrame servers. The ICA Java Client supports both SOCKS and Secure proxy protocols.

Configure the ICA Java Client to work with a proxy server by specifying parameters on the HTML page, through the **ICA Java Client Settings** dialog box, or through the Web Interface. This section describes how to specify a proxy server on the HTML page. Performing this task using the **ICA Java Client Settings** dialog box is described in the Help. Configuring the ICA Java Client through the Web Interface is described in the *Web Interface for MetaFrame XP Administrator’s Guide*.

Some proxy servers require authentication when a connection is requested. Specify a proxy logon name and password on the HTML page when configuring the ICA Java Client. If authentication is required and you do not specify the details on the HTML page, users are prompted to enter the logon name and password when they open an ICA connection.

You can configure the ICA Java Client to use a proxy server in any of the following three ways:

- Enabling auto client proxy detection
- Using a PAC file to automate proxy configuration
- Manually specifying proxy server details

Enabling Auto Client Proxy Detection

If you are deploying the ICA Java Client in an organization with many proxy servers, consider using auto client proxy detection. Auto client proxy detection obtains proxy details from the local Web browser settings. It is also useful if you cannot determine which proxy server will be used when you configure the client. Auto client proxy detection can be used with:

- Internet Explorer 4.0 or later for Windows
- Netscape 4.x or later for Windows, UNIX, and Linux

► **To automatically detect the proxy server settings used by the Web browser (auto client proxy detection)**

Specify the following parameter in the HTML file:

```
<param name=”ProxyType” value=”auto”>
```

Using a PAC File to Automate Proxy Configuration

You can also automate proxy configuration by using a PAC file. A PAC file is a JavaScript file that is served from a local Web server and used to automatically configure the proxy settings of Web browsers.

▶ To obtain the proxy server settings from a PAC file

Specify the following parameters in the HTML file:

```
<param name="ProxyType" value="script">
<param name="ProxyAutoConfigURL"
    value="http://webserver.mycompany.com/myproxies.pac">
```

where http://*webserver.mycompany.com/myproxies.pac* is the URL for the PAC file.

Manually Specifying Proxy Server Details

If you are manually specifying the proxy server, you need to know its address. You also need to know its port number if it is not set to 1080 for a SOCKS proxy server or 8080 for a Secure proxy server.

▶ To specify the proxy server details manually

Note If you are configuring the proxy manually, confirm these details with your security administrator. ICA connections cannot be made if these details are incorrect.

Specify the following details using parameters in the HTML file:

- The address of the proxy server
- The port number of the proxy server (if not 1080 for SOCKS or 8080 for Secure proxy)
- The protocol of the proxy server: SOCKS proxy or Secure proxy
- In the case of a SOCKS proxy, the protocol version number

- If the proxy requires authentication and you are supplying it through the HTML page, the logon name and password.

Parameter	Description
ProxyType=none | auto | socks | socksv4 | socksv5 | secure | script	none - no proxy auto - use the Web browser’s settings socks - use SOCKS - automatically detect the version socksv4 - use SOCKS version 4 socksv5 - use SOCKS version 5 secure - Use Secure proxy script - use a PAC file (specified by ProxyAutoConfigURL)
ProxyHost=*address:port*	Address and port (if required) of the proxy server
ProxyUsername	Proxy Username
ProxyPassword	Proxy Password
ProxyExcludeList=*address1*; *address2*; etc...	A semicolon-separated list of addresses of MetaFrame servers that the client must connect to directly—not through the proxy server.
ProxyAutoConfigURL	The URL for the PAC file. Use with the parameter ProxyType.

For example, to connect to a server named Norbert using a SOCKS proxy server, Version 5, at the IP address 10.45.1.3 and port 1080, use an HTML page like the following:

```
<html>
 <body>
    <applet code=com.citrix.JICA
        archive=JICAEngN.jar
        width=640 height=480>
        <param name=”cabinets” value=”JICAEngM.cab”>
        <param name=”Address” value=”Norbert”>
        <param name=”ProxyType” value=”socksv5”>
        <param name=”ProxyHost” value=”10.45.1.3:1080”>
        <param name=”ProxyUsername” value=”dentres”>
        <param name=”ProxyPassword” value=”sangle”>
    </applet>
 </body>
</html>
```

Integrating the ICA Java Client with the Secure Gateway for MetaFrame or SSL Relay

Note The SSL and TLS protocols cannot be used with Netscape 4.x or earlier because the key generation is not sufficiently random for a secure connection. The one exception to this is Netscape 4.61 on OS/2: you can use SSL and TLS protocols so long as you disable the JIT (Just In Time compiler) by adding -nojit to the command line options for Java in the Netscape configuration.

You can integrate the ICA Java Client with a Secure Gateway or SSL Relay service. The client supports both SSL and TLS protocols:

- SSL provides strong encryption to increase the privacy of your ICA connections and certificate-based server authentication to ensure that the server you are connecting to is a genuine server.
- TLS (Transport Layer Security) is the latest, standardized version of the SSL protocol. The Internet Engineering Taskforce (IETF) renamed it TLS when they took over responsibility for the development of SSL as an open standard. TLS secures data communications by providing server authentication, encryption of the data stream, and message integrity checks. Because there are only minor technical differences between SSL Version 3.0 and TLS Version 1.0, the certificates you use for SSL in your MetaFrame installation also work with TLS. Some organizations, including US government organizations, require the use of TLS to secure data communications.

For more information about Secure Gateway, see the *Secure Gateway for MetaFrame Administrator's Guide.*

Configuring and Enabling the Client for SSL and TLS

SSL and TLS are configured in the same way, use the same certificates, and are enabled with the same parameter. You configure SSL and TLS using parameters on the HTML page or with the Web Interface.

When SSL and TLS are enabled, each time you initiate a connection the client tries to use TLS first, then tries SSL. If it cannot connect with SSL, the connection fails and an error message appears.

Check that the built-in CA (Certification Authority) root certificates listed below meet the security requirements for your organization. If your organization has a certificate authority and you need to specify your own certificates, see "Configuring the ICA Java Client for Use with Your Security Solution" on page 55.

Certificate	Issuing Authority
Class4PCA_G2_v2.crt	VeriSign Trust Network
Class3PCA_G2_v2.crt	VeriSign Trust Network
BTCTRoot.crt	Baltimore Cyber Trust Root
GTECTGlobalRoot.crt	GTE Cyber Trust Global Root
GTECTRoot.crt	GTE Cyber Trust Root
Pcs3ss_v4.crt	Class 3 Public Primary Certification Authority
SecureServer.crt	Secure Server Certification Authority

▸ To enable SSL and TLS

1. If you have not already done so, include the appropriate cryptoj and ssl archives in the archive parameter in the applet tag on the HTML page. These archives are not included in the complete JICAEng archive and you must include them before SSL or TLS encryption can be used.
2. Configure the Web server so that the HTML page specifying the applet can be delivered to the Web browser only through an SSL/TLS (https://) connection.

CAUTION Security is seriously compromised if this step is omitted.

3. Change the network protocol to SSL/TLS+HTTPS and enable SSL by adding the following parameters to the HTML page:

```
<param name="SSLEnable" value="on">
<param name="BrowserProtocol" value="HTTPonTCP">
<param name="HTTPBrowserAddress" value="myserver.mycompany.com">
```

where *myserver.mycompany.com* is the fully qualified domain name of a MetaFrame server.

▸ To force TLS connections

To force clients to connect only with TLS, you must specify TLS on the Secure Gateway server or SSL Relay service. See the *Secure Gateway for MetaFrame Administrator's Guide* or SSL Relay service documentation for more information.

Configuring the ICA Java Client for Use with Your Security Solution

When you enable the SSL/TLS+HTTPS protocol, the default settings apply. The client attempts to connect using the TLS protocol. If a TLS connection cannot be established, the client tries SSL. If an SSL connection cannot be established, the connection fails and an error message appears.

You must perform further configuration if:

- You are using a Secure Gateway server that is configured to run in Relay mode. If you are not sure what this means, see the *Secure Gateway for MetaFrame Administrator's Guide* or contact your security administrator for more information.
- You are not using the default Cipher Suite. You may need to use the Com or Gov cipher suites to comply with your organization's security regulations. If you are not sure, contact your security administrator.
- Your organization has its own certificate authority. You may need to specify alternative root certificates to comply with your organization's security regulations. If you are not sure, contact your security administrator.

This section describes how to configure these three parameters.

▸ To specify a Secure Gateway server that is configured to run in Relay mode

Add the following parameter to the HTML page:

```
<param name="SSLProxyHost" value="address:port">
```

Where *address* is the fully qualified domain name of the Secure Gateway server and the same domain name specified in the server certificate. If the server port is not 443, specify the port. You can also specify a Secure Gateway server address in the **ICA Java Client Settings** dialog box.

To use a different Cipher Suite

Add the following parameter to the HTML page:

```
<param name=”SSLCiphers” value=”All|Com|Gov”
```

where:

Option	Description
All	SSL_RSA_WITH_RC4_128_MD5 SSL_RSA_WITH_RC4_128_SHA SSL_RSA_WITH_3DES_EDE_CBC_SHA
Com	SSL_RSA_WITH_RC4_128_MD5 SSL_RSA_WITH_RC4_128_SHA
Gov	SSL_RSA_WITH_3DES_EDE_CBC_SHA

To use your own certificates

You can also use your own SSL certificates with the ICA Java Client. To do this, you need to create an archive that contains your certificates, add this archive to the list of ICA Java Client archives, and then use parameters to specify the number of certificates and their names:

1. Download a development kit. Before you can create the archive, you must obtain an appropriate kit.
    - To create .jar files for Netscape and other JVMs, download a copy of the Sun JDK from http://java.sun.com/
    - To create .cab files for Internet Explorer, download the Cabinet Software Development Kit from http://microsoft.com
2. Create the archive as a .jar or .cab file.
    - To make a .jar file containing the certificates A.crt, B.crt, and C.crt, install the Sun JDK tools (and put them on your path), then type the following at the command line:

      ```
      jar -cf MyCertificates.jar A.crt B.crt C.crt
      ```

      This command generates an archive called MyCertificates.jar. Specify the archive in the archive parameter in the HTML page. Use commas to separate multiple archives in an applet tag.

- To make a .cab file containing the certificates A.crt, B.crt, and C.crt, install the Cabinet Software Development Kit (and ensure the CABARC tool is in your path), then type the following at the command line:

  ```
  CABARC n MyCertificates.cab A.crt B.crt C.crt
  ```

  This command generates an archive called MyCertificates.cab. Specify the archive in the cabinets parameter of your HTML page. Use commas to separate multiple archives in a cabinets parameter.

3. Specify which SSL certificates the ICA Java Client should use. Use the following parameters:

   SSLNoCACerts - The number of specified certificates in the archive. The default value of zero indicates that only the ICA Java Client built-in set of certificates will be used.

   SSLCACert0, SSLCACert1, ... SSLCACert<n> - The names of SSL root certificates to use for validating the SSL certificate obtained from the SSL Relay. The number of root certificates you specify must match the number specified in the SSLNoCACerts parameter. With these parameters, you can also specify the ICA Java Client built-in certificates, in addition to your own custom certificates (the former reside in the ssl archive, the latter reside in the archive you created following the steps above).

Here is an example of an applet tag that enables SSL with the ICA Java Client and uses the custom root certificates A.crt and B.crt, in addition to the ICA Java Client built-in root certificates, Verisign Class 4, and Baltimore.

```
<applet code=com.citrix.JICA width=640 height=480>
    <param name="cabinets"
    value="JICA-coreM.cab,sslM.cab,cryptojM.cab,MyCertificates.cab">
    <param name="Address" value="myserver.mycompany.com">
    <param name="SSLEnable" value="on">
    <param name="BrowserProtocol" value="HTTPonTCP">
    <param name="SSLNoCACerts" value="4">
    <param name="SSLCACert0" value="Class4PCA_G2_v2.crt">
    <param name="SSLCACert1" value="BTCTRoot.crt">
    <param name="SSLCACert2" value="A.crt">
    <param name="SSLCACert3" value="B.crt">
</applet>
```

Connecting to a Server Across a Firewall

Network firewalls can allow or block packets based on the destination address and port. If you are using the ICA Java Client through a network firewall that employs IP address translation, you must specify the following parameters:

- **UseAlternateAddress**: Use an alternate address across a firewall (specified by the parameter HTTPBrowserAddress or TCPBrowserAddress). The values are **0** (default; actual address is used) and **1** (alternate address is used). If this parameter is set to **1**, the parameter HTTPBrowserAddress or TCPBrowserAddress must also be specified. Setting this parameter to **0** is the same as not using the parameter.

  You can also enable alternate addressing using the **ICA Java Client Settings** dialog box.
- **HTTPBrowserAddress** or **TCPBrowserAddress**: The external Internet address of a MetaFrame server.

Note All MetaFrame servers in the farm must be configured with their alternate (external) address. See the *MetaFrame XP Server Administrator's Guide* for more information.

For example, to connect to a server across a firewall in applet mode and use an alternate address for the server **Fountain**, make an HTML page like the following:

```
<html>
 <body>
   <applet code=com.citrix.JICA
     archive="JICAEngN.jar"
     width=640 height=480>
     <param name="cabinets" value="JICAEngM.cab">
     <param name="Address" value="Fountain">
     <param name="HTTPBrowserAddress" value="177.17.1.7">
     <param name="user.wfclient.UseAlternateAddress" value="1">
   </applet>
 </body>
</html>
```

Using ICA Encryption

Note ICA encryption cannot be used with Netscape 4.x or earlier because the key generation in these versions is insufficiently random for a secure connection.

ICA encryption is included in MetaFrame XP for Windows Feature Release 1 or later. The default level is **Basic**.

To enable ICA encryption levels higher than **Basic**:

1. Ensure the MetaFrame server is configured to allow the selected encryption level or greater. To enable encryption levels higher than **Basic**, the MetaFrame server must support RC5 encryption. This support is included with SecureICA Services, MetaFrame 1.8 Feature Release 1, and MetaFrame XP.
2. If you are using the core archive, include the cryptoj and JICA-sica archives in the archive attributes or cabinets parameter on the HTML page, if you have not already done so. If you are using a complete archive, only include cryptoj.
3. Set the appropriate encryption level.

Note Selecting an encryption level higher than **Basic** disables automatic logon to the MetaFrame server.

▸ To specify ICA encryption

Use the following parameter on an HTML page:

```
<param name=”EncryptionLevel” value=”0|1|2|5”>
```

The number you enter corresponds to the encryption level required as follows:

Level	Description
0	No encryption (needs MetaFrame server setup; default is Basic).
1	Basic encryption. This is the default.
2	RC5 128-bit encryption during authentication. After the logon process is successfully completed, the encryption level changes to Basic.
5	RC5 128-bit. This is intended for users who are dealing with sensitive data and need a high level of privacy and integrity.

To create a connection to a MetaFrame server called CitrixServer using 128-bit ICA encryption using Netscape 6, create an HTML page like the following:

```
<html>
 <body>
    <applet code=com.citrix.JICA
      archive=”JICA-coreN.jar,cryptojN.jar,JICA-sicaN.jar”
      width=640 height=480>
      <param name=”cabinets” value=”JICA-coreM.cab,cryptojM.cab,
                                             JICA-sicaM.cab”>
      <param name=”Address” value=”CitrixServer”>
      <param name=”EncryptionLevel” value=”5”>
    </applet>
 </body>
</html>
```

Advanced Configuration

Precedence of Configuration Locations

This section describes the different places parameters can be specified and the precedence of each place.

The ICA Java Client configuration is stored in three places:

- The HTML page, stored on a Web server
- The ICA file, served from a server running the Web Interface or from a standard Web server
- The appsrv.ini file, stored on the user’s local device

Parameters specified on the HTML page take precedence over those specified in the ICA file. Similarly, parameters specified in the ICA file take precedence over those in the appsrv.ini file.

The location of the appsrv.ini configuration file depends on the client operating system and Web browser. The locations for some common Web browser and operating system combinations are as follows:

Sun Plugin 1.3.x and 1.4x (Win32):

```
%USERPROFILE%\Citrix
```

Microsoft VM (Win32):

```
%SYSTEMROOT%\Java\Citrix
```

Netscape 4.x (Win32):

```
<Netscape user directory>\Citrix
```

Netscape 6.x and later (Win32):

```
%USERPROFILE%\Citrix
```

Netscape 4.x (UNIX):

```
$HOME/.netscape/.citrix
```

Netscape 6.x and later (UNIX):

```
$HOME/.citrix
```

Internet Explorer (Mac OS X):

```
<User's home directory>/.citrix
```

Netscape 7 (Mac OS X):

```
<User's home directory>/.citrix
```

Copying Client Drive Mapping Configuration

If you are deploying the ICA Java Client to multiple users, you typically configure all common settings through the HTML page or the ICA file. However, client drive mapping settings can be configured only through the **ICA Java Client Settings** dialog box and are stored only in the appsrv.ini file. They cannot be configured through the ICA file or the HTML page.

You can deploy client drive mapping settings by configuring client drive mapping on one device and copying the appsrv.ini file to the correct location on the users' devices.

Changing the List of Printer Drivers

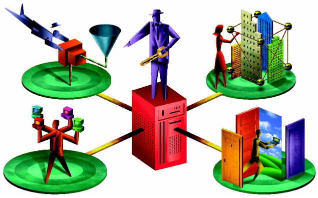
When you configure auto-created printers using the **ICA Java Client Settings** dialog box, you set the driver by choosing one from a list.

You can change the list of drivers to match the printer drivers that are available on the MetaFrame server by editing the ICAPrinterDrivers.txt file. This plain text file is included in the ICA Java Client package and is located in the same directory as the client archives. The contents of the default file are listed in "ICAPrinterDrivers.txt File" on page 83.

Use a text editor such as Notepad to open the file. Add or remove driver names by deleting or adding names to the file, one driver name per line. You can add the driver names in any order.

CHAPTER 5

Improving the Performance of the Citrix ICA Java Client



This chapter describes how you can improve the performance of the ICA Java Client using:

- Session sharing
- Data compression
- Data caching
- Queuing mouse movements and keystrokes
- SpeedScreen Latency Reduction

Advice is also provided on how to improve performance over a low-bandwidth connection.

Session Sharing

If you are not using seamless windows, consider moving to this mode of working in order to gain the performance benefits of session sharing. These include a reduction in system overhead and quicker application launching. For more details of seamless windows, see “Seamless Support” on page 14.

Data Compression

Data compression reduces the amount of data that needs to be transferred over the network. However, additional processor resources are required to compress and decompress the data. If your connection is bandwidth-limited, enabling data compression improves performance.

You can specify two levels of data compression, standard and maximum. Maximum data compression uses more processor power and memory and may reduce performance on slow devices. Data compression is enabled by default.

▶ To enable maximum data compression

Specify the following parameters on the HTML page:

```
<param name=”Compress” value=”on”>
<param name=”Maximumcompression” value=”on”>
```

▶ To disable data compression

Specify the following parameter on the HTML page:

```
<param name=”Compress” value=”off”>
```

Bitmap Caching

Bitmap caching stores commonly used images on a local disk. If the connection is bandwidth-limited, using bitmap caching increases performance. If the client is on a high-speed LAN, you do not need bitmap caching.

You can configure bitmap caching using parameters on the HTML page or the user can do it through the **ICA Java Client Settings** dialog box.

▶ To enable bitmap caching for a connection

Specify the following parameter on the HTML page:

```
<param name=”PersistentCacheEnabled” value=”on”>
```

The bitmap cache directory is stored in the directory specified by the standard Java system property user.home. The location of the user.home directory depends on the JVM or Web browser you are using. If necessary, use the **ICA Java Client Settings** dialog box to change the bitmap cache directory.

Queuing Mouse Movements

When queuing is enabled, the client sends mouse movement updates less frequently to the MetaFrame server. This prevents the connection becoming overburdened with excessive mouse events, which would degrade performance. Disabling queuing makes the ICA session more responsive to mouse movements. By default, mouse movement updates are sent every 150 milliseconds.

▶ To change the update period for mouse movement queuing

Specify the following parameter on the HTML page:

```
<param name=”MouseTimer” value=”period”>
```

Where *period* is the update period you are using, in milliseconds. To disable mouse movement queuing, set the period to zero.

Using SpeedScreen Latency Reduction

SpeedScreen Latency Reduction improves performance over high latency connections by providing instant feedback to the user in response to typed data or mouse clicks.

Note SpeedScreen Latency Reduction works only when it is enabled on the MetaFrame server to which you are connecting. See the MetaFrame server documentation for more details.

▶ To enable mouse click feedback

Add the following parameter to the HTML page:

```
<param name=”ZLMouseMode” value=”0|1|2”>
```

where **0** is off, **1** is on, and **2** is automatic. When mouse click feedback is set to automatic, the client enables it when required.

▶ To enable local text echo

Add the following parameter to the HTML page:

```
<param name=”ZLKeyboardMode” value=”0|1|2”>
```

where 0 is off, 1 is on, and 2 is automatic. When local text echo is set to automatic, the client enables it when required.

Improving Performance over a Low-Bandwidth Connection

If you are using ICA over a low-bandwidth connection, such as a modem or cellular telephone, there are a number of changes that you can make to your client configuration and the way you use the client to improve performance:

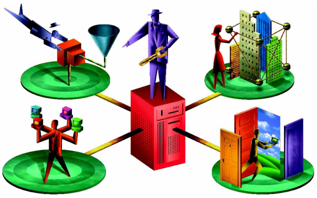
- **Reduce the size of the ICA Java Client applet.** If you do not require the entire functionality of the ICA Java Client, specify only the archives necessary to provide the functions you require. Chapter 3 of this guide explains how to specify component archives. Reducing the size of the applet can greatly reduce the download time.

- **Change the ICA Client configuration.** On devices with limited processing power, or where limited bandwidth is available, there is a trade-off between performance and functionality. The ICA Client provides both user and administrator with the ability to choose an acceptable mixture of rich functionality and interactive performance. Making one or more of the following changes can reduce the bandwidth that the connection requires and improve performance:
    - **Enable maximum data compression.** Compression reduces the size of the data that is transferred over the ICA connection (see“Data Compression” on page 63).
    - **Enable the bitmap cache.** Bitmap caching stores commonly used images locally on the client so that they do not have to be transferred over the ICA connection every time they are needed (see “Bitmap Caching” on page 64).
    - **Queue mouse movements.** When queuing is enabled, the client sends mouse updates to the MetaFrame server less frequently. Increasing the queuing period may improve performance on a low-bandwidth connection (see “Queuing Mouse Movements” on page 64).
    - **Enable SpeedScreen Latency Reduction.** SpeedScreen Latency Reduction improves performance over high latency connections by providing instant feedback to the user in response to typed data or mouse clicks (see “Using SpeedScreen Latency Reduction” on page 65).
    - **Reduce the window size.** Change the window size to the minimum size users can comfortably use (see “Editing desktop.html” on page 23).
    - **Reduce the number of colors.** Reduce the number of colors to 256 (see “Window Properties” on page 37).
    - **Disable client audio mapping.** If your users do not need sound, disable client audio mapping (see “Client Audio Mapping” on page 48), or remove the JICA-audio archive.
    - **Disable clipboard mapping**. If users do not need to copy and paste text, do not include the JICA-clipboard archive.
- **Change the way you use the client.** ICA technology is highly optimized and typically does not have high CPU and bandwidth requirements. However, users on a low-bandwidth connection should consider the following to preserve performance:
    - **Avoid accessing large files using client drive mapping.** When you access a large file with client drive mapping, the file is transferred over the ICA connection. On slow connections, this may take a long time.

  - **Avoid printing large documents on local client printers.** When you print a document on a local client printer, the print file is transferred over the ICA connection. On slow connections, this may take a long time.
  - **Avoid playing multimedia content.** Playing multimedia content uses a lot of bandwidth and can cause reduced performance.
- **Use the latest MetaFrame server and ICA Clients.** Citrix is continually enhancing and improving ICA performance with each release, and many performance features require the latest client and server software in order to function.

CHAPTER 6

Limitations of the Citrix ICA Java Client



This chapter describes some known limitations of the ICA Java Client and suggests workarounds where applicable.

Seamless Windows

Seamless windows are not supported in Netscape 4.x on any UNIX/Linux platform, or on Mac OS 9 or Mac OS X. If the ICA Java Client is configured for seamless mode and run on these platforms, a non-seamless session is launched.

The ‘outline’ drag-box that is displayed when you move or resize seamless windows can cause unattractive repaint effects on some platforms. If you experience this, you can either specify a solid drag-box type or eliminate the drag-box entirely by setting the parameter TWIDragBoxType to **solid** or **none** respectively.

COM Port Mapping

COM port mapping is no longer supported; it requires third party libraries to be installed on the client device and is therefore not appropriate for zero-install applet mode deployment. If you require this functionality, Citrix recommends that you use the Version 6.20 application mode ICA Java Client.

Server Names with non-ASCII Characters

For servers to be contactable by client operating systems, it is generally necessary for the server’s host name and domain name to be compliant with the Domain Naming System (DNS). That is, the names should contain only upper- and lowercase ASCII a to z, digits 0 to 9, and dash (-). This may not be necessary when client operating systems and the DNS server support multinational characters.

Published application names can comprise any characters because these are always resolved by the ICA browser. Note that any ICA browser addresses (HTTPBrowserAddress or TCPBrowserAddress) have the same DNS restrictions as the server names described above.

On some systems, you may not be able to connect to a server if its name does not conform to this requirement (that is, a server with a Japanese name or a name containing other non-ASCII characters).

To avoid this, either:

- Conform to the DNS requirement and use only ASCII characters in server names.

—Or—

- Specify ICA browser addresses as IP addresses and also do not enable DNS resolution for the MetaFrame server. If you do both of these things, the MetaFrame servers can have names that do not comply with the DNS requirement. However, note that SSL will not work in this scenario because DNS names are required for certificate validation, and, in addition, the Web Interface will not work because DNS is used to resolve the hostname part of the Web Interface URL.

Client Drive Mapping

A remote application generates an error message because it cannot lock a file.

Due to Java limitations, Java programs cannot lock files. This means that files cannot be locked on mapped local drives.

To avoid generating error messages, disable the Safelock option on the HTML page:

```
<param name=”Safelock” value=”off”>
```

When Safelock is disabled, if a remote application attempts to lock a file on a mapped drive, the ICA Java Client will report success. However, this does not mean that the file is locked; you must take precautions to ensure that another application does not write to or delete the same file, because this could result in data corruption.

Client Printer Mapping

When using the ICA Java Client on the following Web browser and operating system combinations, the list of printer drivers in the **ICA Java Client Settings** dialog box does not appear:

- Netscape 4.x on any platform

- Internet Explorer on Mac OS 9.x
- Internet Explorer on Mac OS X

To solve this problem, you need to add the ICAPrinterDrivers.txt file to an archive and add this archive to the list of Java Client archives specified on the HTML page. The ICAPrinterDrivers.txt file is a plain text file, included in the ICA Java Client package and located in the same directory as the client archives.

To add the ICAPrinterDrivers.txt file to an archive

1. Before you can create the archive, you must obtain an appropriate development kit. To create .jar files, download a copy of the Sun JDK from http://java.sun.com/.
2. To make a .jar file containing ICAPrinterDrivers.txt, install the Sun JDK tools (and put them in your path), then type the following at the command line:

   ```
   jar -cf PrinterDrivers.jar ICAPrinterDrivers.txt
   ```

   This command generates an archive called PrinterDrivers.jar.

Specify the archive on the HTML page. Note that multiple archives in an applet tag must be separated by commas.

Clipboard Support on X11

- ICA Java Client clipboard support works only with applications that use the X11 CLIPBOARD selection. Specifically, Motif and Gnome applications are fine, but KDE applications do not work. Xterm can be configured with X11 resources to use the CLIPBOARD selection, for example:

  ```
  XTerm*VT100.Translations: #override \n\
  ~Ctrl ~Meta <Btn2Up>:\
  insert-selection(PRIMARY,CLIPBOARD,CUT_BUFFER0)\n\
  <BtnUp>: select-end(PRIMARY,CLIPBOARD,CUT_BUFFER0)\n
  ```

  GNU Emacs can also be configured to use CLIPBOARD with the following Emacs lisp:

  ```
  (setq x-select-enable-clipboard t)
  ```

- The xclipboard utility (present on most X11 systems) may be of use in transferring data between the PRIMARY and CLIPBOARD selections.

Mouse Pointer Support

Java 1.1 supports only 14 different mouse pointers, so in these environments the ICA Java Client makes a best effort attempt to map remote pointer shapes to one of these 14.

Java 2 supports arbitrary mouse pointers, so in these environments the ICA Java Client can display a correct mouse pointer. You can obtain a Java 2 applet environment by installing Sun’s Java Plug-in.

Arbitrary mouse pointers are not supported on Mac OS X.

In shadowed sessions, the mouse pointer does not move when the shadowing user moves his or her mouse pointer.

Mouse Wheel Support

Java 1.1 does not provide any mouse wheel support, so using the mouse wheel in these environments has no effect in the remote session.

J2SE Version 1.4 environments do provide mouse wheel support, so in these environments the ICA Java Client propagates mouse wheel events to the remote session. You can obtain a Java 2 applet environment by installing Sun’s Java Plug-in.

Internet Explorer

- The browser does not correctly route some ALT key events. For example, pressing ALT+F when an application running in the ICA session has focus sends the event to both the application and to the browser.
- When Web browser scroll bars are in use, the session is sometimes repainted incorrectly.
- To display dead key characters, you must press the chosen dead key and then press the space bar twice.
- When using Internet Explorer on Windows with Microsoft's Virtual Machine for Java, status messages are displayed either in the Web browser's status bar or (if present) in the ICA Java Client status bar. However, status messages displayed in the Web browser's status bar are not permanently visible; instead, they flash on and off as the mouse pointer enters and exits the applet. This means that indications of SSL/TLS encryption are not always visible, and on some devices these messages may be impossible to see. To avoid this, either ensure that the ICA Java Client status bar is used, or use the Sun Java Plug-in.

Netscape

- The ICA Java Client does not support ICA Encryption or SSL/TLS protocols on Netscape 4.x. If you require encryption, upgrade to Netscape 6.x/7.x.
- Netscape 4.x for Mac OS 9 does not fully support Java 1.1 applets. Netscape 4.76 and later using the MRJ Plug-in does work; it has not been extensively tested.
- The Java Environment under Netscape 4.x does not correctly report the system's locale and the Java Client applet always displays English text. To work around this problem, you need to explicitly specify the language as an applet parameter, for example:

```
<applet code=com.citrix.JICA
 archive=”JICAEngN.jar” width=640 height=480>
 <param name="Language" value="es">
</applet>
```

- Auto-locate with UDP broadcast is not supported in the Netscape 4.x JVM. To work around this limitation, specify a server location. See “Server Location” on page 33.

Mac OS 9 and Mac OS X

- To right-click when connected to a MetaFrame server with a Macintosh, hold down the command key and click the mouse button.
- Characters generated using the Option keys on Macintosh may not be supported by the current Windows font in your ICA session. If the character produced is not the expected character, choose a Windows font in the ICA session that supports the character. After producing the desired character, you can return to the usual font.

Mac OS 9

- When using Internet Explorer on Mac OS 9, the TAB key cannot be used in a MetaFrame session. Internet Explorer uses the TAB key to move keyboard focus within the Web browser and consequently the event does not get sent to the MetaFrame server. This is a limitation of Internet Explorer.
- CTRL keys on Mac OS 9 cannot be correctly detected.

Mac OS X

The ICA Java Client cannot load when the Java archive files are hosted on a Web server that requires cookies or authentication before serving the files. Ensure that the Web server does not require authentication to serve these files.

This issue also affects the HTML help and .ica files.

In particular, asp or jsp applications that manage session state using cookies are affected.

OS/2 Session Sharing

On OS/2 platforms a separate Connection Center is instantiated for each seamless application when using the ICA Java Client with the Web Interface or with custom HTML containing the MAYSCRIPT attribute in the applet tag. Consequently session sharing does not take place.

To get session sharing functionality on OS/2, you must remove the MAYSCRIPT attribute from the Web Interface file **\Citrix\NFuse\site\include\appembed.inc** or from the applet tag in your HTML. However, doing this means that proxy auto-configuration (PAC) files cannot be executed. In other words, on OS/2 you can have either session sharing or support for PAC files, but not both.

Miscellaneous

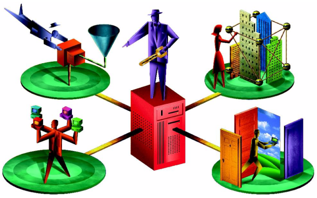
- When connecting to a MetaFrame server, the ICA Java Client cannot detect the state of the Scroll Lock, Number Lock, or Caps Lock keys when the client receives focus.
- Persistent caching for high resolution or color depth is available only with MetaFrame XP 1.0 Feature Release 1 or later and MetaFrame for UNIX 1.0 Feature Release 1 or later.
- In rare circumstances, typing at the keyboard can produce no effect. If this happens, explicitly setting the Keyboard Layout should resolve the problem. See “Keyboard and Mouse” on page 39.
- SSL/TLS is not supported when using key lengths greater than 2048 bits.

Reporting Issues

If you have any issues with the ICA Java Client that are not described in this chapter, please take the following steps:

1. Make sure you are running the latest version of the ICA Java Client.
2. Read the information in the Readme file, located in the root installation directory.
3. Check the relevant section of this guide.
4. If the problem still persists, please report the problem to your Citrix support representative and provide all the following information:
    - The client operating system platform.
    - The Web browser and version number and, if you are using a JVM Plug-in, the Plug-in version number.
    - The complete applet tag from the Web browser's **View Source** (or equivalent) menu. This shows the resulting HTML after any client- or server-side scripting.
    - Any error messages displayed in the Java console.
    - If applicable, any Java stack trace that is displayed after clicking **Details** in an error dialog box.
    - The MetaFrame server platform and MetaFrame server version.
    - The Web Interface version number, if applicable.
    - Any relevant security information; for example, whether you are using ICA encryption or SSL.
    - An explanation of how to reproduce the problem.

Citrix ICA Java Client Parameters and File Descriptions



ICA Java Client Parameters

The following is a complete list of the parameters that can be specified on the HTML page to provide additional features and customization. The Address parameter is the only required parameter; all other parameters in these lists are optional.

General Configuration

Parameter	Description
Address	The address of the MetaFrame server or the name of the published application. If a published application is entered as an address value, the InitialProgram parameter must also be specified.
ICAPortNumber	The default ICA port number is 1494. A different port number can be specified using this parameter or by appending the port number to the address value; for example, **CitrixServer:1495**.
InitialProgram	The name of the initial program to run after connecting to the MetaFrame server. If you are connecting to a published application, add a pound sign or hash mark (#) before the program name.
Param	Passes a parameter such as a file name to a published application.

Parameter	Description
HTTPBrowserAddress	The address of a MetaFrame server. This parameter is used when TCP/IP+HTTP or SSL/TLS+HTTPS browsing has been specified. This parameter is also used to designate groups of primary and backup servers. Note that when the XML Service on the MetaFrame server is not configured to use the default port 80, you must append :<*port number*> to this parameter, substituting <*port number*> with the port number your server's XML Service is configured to use.
TCPBrowserAddress	The address of a MetaFrame server. This parameter is used when the client is on a different subnet from the MetaFrame server. This parameter is also used to designate groups of primary and backup servers. Use with the parameter BrowserProtocol=UDP.
BrowserProtocol	The protocol to use to locate the ICA browser.
WorkDirectory	The path of the working directory where the initial program is run after the user connects to the MetaFrame server.
UseHostname	Specifies using the hostname as the client name. Use in the form client.wfclient.UseHostname.
Clientname	The client name. Use in the form client.wfclient.Clientname. Use with the parameter client wfclient.UseHostname=off.
Username	The user name to use during logon.
Password	The password of the user. The Password parameter cannot be used to specify an encrypted password. To specify an encrypted password, use an ICA file or .ini file that contains an encrypted password.
Domain	The name of the domain for the user name.
Cabinets	Specifies Authenticode signed .cab file archives, used with Internet Explorer for Windows.
Language	Causes the ICA Java Client's user interface components to appear in a language other than the language of the client device.
Icafile	An ICA file for the ICA Java Client to use. The value entered must be a valid URL.
EndSessionTimeout	In Netscape 4.x only, when users navigate away from a Web page containing an open ICA session for more than 300 seconds (five minutes), the session is disconnected. Use this parameter to specify a different value in seconds.

Parameter	Description
CREnabled	Specifies whether or not content redirection is enabled. Values are **yes** or **no**. The default value is **yes**.
TWIDisableSessionSharing	Specifies whether or not session sharing is disabled. Used in conjunction with Connection Center. The default value is **no**.
TWIMode	Enables seamless windows. If you are using seamless windows, you must set this parameter to **on**. If you are not using seamless windows, set it to **off.**

User Interface Parameters

Parameter	Description
Width	The width of the ICA session window. This parameter is specified in the applet tag.
Height	The height of the ICA session window. This parameter is specified in the applet tag.
DesiredColor	The color depth of the ICA session windows. The values are 2 (256 colors), 4 (thousands of colors), and 8 (millions of colors). The default value is 256 colors. 16 color mode is not supported but the client can connect to applications published in 16 color mode, in which case 256 colors are used.
Border	Turns the border around the ICA session in the browser window on or off. Values for this parameter are **on** or **off.** The default value is **off**.
ShowStatusBar	Specifies whether to show the status bar. You must include the complete JICAEng archive or the JICA-config archive to display the status bar. Values for this parameter are **yes** or **no**. The default value is **yes**.
ShowSettingsButton	Specifies whether or not to show the **Settings** button. You must include the complete JICAEng archive or the JICA-config archive to display the **Settings** button. Values for this parameter are **yes** or **no**. The default value is **yes**.
BorderWidth	The border width in pixels. The default value is 6.
Start	Controls the ICA Java Client's behavior when you start a session. The values are **manual** (default) and **auto**.

End	Controls the ICA Java Client's behavior when you terminate a session. The values are **manual** (default), **auto**, **terminate**, and ***URL***.
KeyboardLayout and KeyboardType	The type of keyboard. Use in the form user.wfclient.KeyboardType and user.wfclient.KeyboardLayout.
Hotkey*n*Shift Hotkey*n*Char	Sets hotkeys that can be used to control various client functions. Use in the form user.wfclient.Hotkey*n*Shift or user.wfclient.Hotkey*n*Char where *n* is the number of the hotkey. *n* can be 2, 3, 4, 5, 6, 7, 8, 9, or 10.

Client Device Mapping Parameters

Parameter	Description
LocalClientPrinters	Used for passing information about client printers to the server. Must be specified as <param name=”user.localclientprinters” value=”*printername*”>. To specify more than one printer, separate the printer names with commas.
Port	The printer port. Use in the form user.*printername*.Port, where *printername* is the name allocated with the parameter LocalClientPrinters.
Driver	The printer driver. Use in the form user.*printername*.Driver, where *printername* is the name allocated with the parameter LocalClientPrinters.
ClientAudio	Enables client audio. The values are **on** and **off**. The default is **off**.

Performance Tuning Parameters

Parameter	Description
Compress	Sets data compression. The values are **on** to enable data compression and **off** to disable it. By default, compression is enabled.
MaximumCompression	Sets high level data compression. The values are **on** to enable greater data compression and **off** to select normal data compression. Requires Compress parameter to be enabled.
PersistentCache Enabled	Enables or disables bitmap caching. Must be specified as “user.wfclient.PersistentCacheEnabled”. Values for this parameter are **on** or **off**. The default value is **off**.
PersistentCacheSize	The size of the bitmap cache in bytes.
PersistentCacheMin Bitmap	The size of the smallest bitmap to cache, in bytes.
MouseTimer	The period (in milliseconds) between the mouse movement updates that are sent to the MetaFrame server. Set to 0 to disable queuing.
ZLMouseMode	SpeedScreen Latency Reduction mode. The values are 0 (off), 1 (on), or 2 (auto). The default is 2.
ZLKeyboardMode	SpeedScreen Latency Reduction mode. The values are 0 (off), 1 (on), or 2 (auto). The default is 0.

Security Integration Parameters

Parameter	Description
ProxyType	The type of proxy server.
ProxyHost	The location and port of the proxy server.
ProxyUsername ProxyPassword	Proxy server logon credentials.
ProxyExcludeList	A semicolon- or comma-separated list of addresses of MetaFrame servers that the client must connect to directly—not through the proxy server.
ProxyAutoConfigURL	The location of a Proxy Auto Configuration (PAC) file for automatic proxy configuration.
SSLEnable	Enables SSL and TLS encryption protocols. Used with the BrowserProtocol parameter.
SSLProxyHost	A Secure Gateway (Relay Mode) address.
SSLCiphers	An alternative cipher suite.
SSLNoCACerts	The number of additional certificates.
SSLCACert0... SSLCACert*	The name of an additional certificate.
UseAlternateAddress	Specifies whether or not to use an alternate server address across a firewall. Used in the form user.wfclient.UseAlternateAddress.
EncryptionLevel	The level of ICA encryption to use for an ICA connection.

ICAPrinterDrivers.txt File

The contents of the default ICAPrinterDrivers.txt file are as follows:

```
HP LaserJet
HP DeskJet
HP OfficeJet
HP LaserJet Series II
HP LaserJet IIIHP LaserJet 4
HP LaserJet 5
HP LaserJet 4000 Series PCL
HP LaserJet 4000 Series PS
HP LaserJet 4050 Series PCL
HP LaserJet 4050 Series PS
HP LaserJet 5000 Series PCL
HP LaserJet 5000 Series PS
HP LaserJet 8000 Series PCL
HP LaserJet 8000 Series PS
HP LaserJet 8100 Series PCL
HP LaserJet 8100 Series PS
Canon Bubble-Jet BJC-70
Canon Bubble-Jet BJ-200ex
Canon Bubble-Jet BJC-600
Canon LBP-4
Canon LBP-8II
Canon LBP-8III
Epson Stylus Pro ESC/P 2
Epson Stylus COLOR ESC/P 2
Epson Stylus Photo ESC/P 2
Epson EPL-3000
Epson EPL-4000
Epson EPL-5000
Epson EPL-6000
Epson EPL-7000
Epson EPL-8000
Epson EPL-9000
Lexmark Optra
```

Supported Keyboards

The keyboards supported by Citrix MetaFrame are:

```
"(Server Default)"
"Albanian"
"Belarusian"
"Belgian Dutch"
"Belgian French"
"Brazilian (ABNT)"
"British"
"Bulgarian (Latin)"
"Bulgarian"
"Canadian English (Multilingual)"
"Canadian French (Multilingual)"
"Canadian French"
"Croatian"
"Czech (QWERTY)"
"Czech"
"Danish"
"Dutch"
"Estonian"
"Finnish"
"French"
"German (IBM)"
"German"
"Greek (220) Latin"
"Greek (220)"
"Greek (319) Latin"
"Greek (319)"
"Greek Latin"
"Greek"
"Hungarian 101-Key"
"Hungarian"
"Icelandic"
"Irish"
"Italian (142)"
"Italian"
"Japanese"
"Korean"
"Latin American"
"Latvian (QWERTY)"
"Latvian"
"Lithuanian"
"Norwegian"
"Polish (214)"
"Polish (Programmers)"
"Portuguese"
"Romanian"
"Russian (Typewriter)"
"Russian"
"Serbian (Cyrillic)"
"Serbian (Latin)"
"Slovak (QWERTY)"
```

```
"Slovak"
"Slovenian"
"Spanish Variation"
"Spanish”
"Swedish"
"Swiss French"
"Swiss German"
"Taiwan"
"Turkish (F)"
"Turkish (Q)"
"Ukrainian"
"United Kingdom"
"US"
"US-Dvorak for Right hand"
"US-Dvorak for left hand"
"US-Dvorak"
"US-International"
```

Index