Citrix Receiver for Linux OEM's Reference Guide

Citrix Receiver™ for Linux, Version 13.0

Contents

Copyright and trademark notice

Trademark acknowledgments

Chapter 1

About this document

Topics:

The purpose of this document is to support Original Equipment Manufacturers (OEMs) who integrate Citrix® Receiver™ for Linux® into their own or customers' deployments. The document helps you:

- Modify or replace the Receiver for Linux installation
- Customize the Receiver for Linux user interface
- Remove or replace Receiver for Linux libraries

There are two parts to this document: a set of task-based procedures for configuring Receiver, and tables of reference information for command-line utilities, .ini files, and library files.

This document is intended for developers of products that include Receiver for Linux. If you are planning to modify the user interface of Receiver for Linux, Citrix recommends that you read the entire manual.

Citrix eDocs contains the official product documentation for Receiver for Linux. This includes configuration instructions and known issues that may be useful when customizing this component. eDocs is available at http://support.citrix.com/proddocs/topic/receiver/receivers-linux-wrapper.html.

What's new

The following new features are available in this release compared with Version 12.1:

- **Support for XenDesktop® 7 features** - Receiver supports many of the new features and enhancements in XenDesktop 7, including Windows Media® client-side content fetching, HDX™ 3D Pro, HDX RealTime webcam compression, Server-rendered Rich Graphics, and IPv6 support.

  Note: Link-local network addresses are not supported in IPv6 environments. You must have at least one global or unique-local address assigned to your network interface.

- **VDI-in-a-Box support** - You can use Receiver to connect to virtual desktops created with Citrix VDI-in-a-Box.
- **Self-service UI** - A new graphical user interface (UI), like that in other Citrix Receivers, replaces the configuration manager, wfcmgr. After they are set up with an account, users can subscribe to desktops and applications, and then start them.
- **Deprecated and removed utilities** - The pnabrowse command-line utility is deprecated in favor of the new storebrowse command-line utility. The icabrowse and wfcmgr utilities have been removed.
- **StoreFront support** - You can now connect to StoreFront stores as well as Citrix XenApp® sites (also known as Program Neighborhood® Agent sites).
- **UDP audio support** - Most audio features are transported using the ICA® stream and are secured in the same way as other ICA traffic. User Datagram Protocol (UDP) Audio uses a separate, unsecured, transport mechanism, but is more consistent when the network is busy. UDP Audio is primarily designed for Voice over IP (VoIP) connections and requires that audio traffic is of medium quality (that is Speex wideband) and unencrypted.
- **Packaging** - An armhf (hard float) Debian package and tarball are now included in the download packages. In addition, the Debian package for Intel systems uses multiarch (a Debian feature) for installations on 32- and 64-bit systems. 32-bit binaries are also available in RPM packages.
- **System Flow Control** - Video display has been enhanced on low-performance user devices that connect to high-performance servers. In such setups, System Flow Control prevents sessions becoming uncontrollable and unusable.
- **Localization** - Receiver is now available in German, Spanish, French, Japanese, and Simplified Chinese.
- **Platform Optimization SDK** - You can now create plug-in extensions for the client engine (wfica). Plug-ins are built, using the SDK, as shareable libraries that are dynamically loaded by the engine, and are available for various functions. For example, you can create your own JPEG decoding plug-in, based on the supplied ctxjpeg in the SDK. This lets you take advantage of any built-in hardware decoders, either built-in or that you provide, on your platform. The Platform Optimization SDK is different from the Virtual Channel SDK.

- **Keyboard improvements** - You can now specify which local key combination (Ctrl +Alt+End or Ctrl+Alt+Enter) generates the Ctrl+Alt+Delete combination on a remote Windows® desktop. In addition, a new option supports Croatian keyboard layouts.
- **Direct-to-screen bitmap decoding** - Receiver can now decode JPEG-encoded screen tiles directly to screen, avoiding the need to use the bitmap cache on the user device, and providing a performance improvement.
- **Deferred XSync** - While one frame is still on screen, Receiver can now decode tiles for the next frame. This provides a performance improvement compared with previous releases, in which Receiver waited for a frame to finish being displayed before decoding the next frame.
- **Audio and webcam playback improvements** - Various changes are implemented that conserve CPU cycles and reduce latency.
- **Audio settings** - Several new audio settings are now available in module.ini.
- **Performance improvements** - These include ARM-specific performance enhancements using the Advanced SIMD (NEON) instruction set, and enhancements to hardware-accelerated screen tile processing.
- **GStreamer audio** - An experimental feature of this release, you can now use the open-source GStreamer framework to process audio data.

Resources to aid customization

OEMs can make use of the following:

- Receiver for Linux, which is available for download from the Citrix Web site, http://www.citrix.com/.
- Three command-line utilities: storebrowse, wfica, and ctxipc:
  - storebrowse is equivalent to the deprecated pnabrowse utility. It queries Citrix StoreFront for virtual desktops and published applications.
  - wfica is the client engine that creates connections to the server and performs all of the functions of the connections.
  - OEM code uses ctxipc to notify Program Neighborhood Agent when a user inserts or removes a smart card. For more information about this utility, see the *Using Kerberos with Citrix Receiver for Linux Guide*. This is available from Citrix under a non-disclosure agreement.
- A series of .ini configuration files that allow you to customize the behavior of individual connections or users.
- Certain library files (.dll or .so files) that can be added to or removed from the default installation to enable or disable specific functionality.

Tools

If you choose to customize the appearance of Receiver for Linux, Citrix recommends doing so with the GTK or Qt development environment. No other specialized tools are

required. However, the self-service UI requires libwebkitgtk and therefore requires GTK +.

storebrowse and the Authentication Manager and Service Record daemons present UI through the Receiver dialog library, which can be re-engineered in any tool, so you can create an interface using your tool of choice and wrap it around storebrowse.

Receiver for Linux components

This section describes the components that make up Receiver for Linux and describes how developers can configure the client. Typically, such configuration may be required when the user interface of Receiver for Linux is being replaced with a custom version.

About Receiver for Linux

Receiver for Linux is a Linux application that provides access to a session running on a server. When the connection to the server is established, the user can access desktops and applications, and work with files in a way similar to working on a local computer.

Receiver for Linux displays the session on the Linux workstation screen, and is fully integrated with other Linux X applications. The workstation's mouse and keyboard can be used with applications in the usual way, and the user can set up key mappings to enter PC keys that are interpreted locally on the workstation.

Generally, the features in Receiver are performed by software, but with Citrix HDX features you can choose whether these are controlled by hardware or the digital signal processor (DSP), or whether to optimize the running of these by software.

Components used by Receiver for Linux

Receiver for Linux contains the following files:

- **selfservice** - This program replaces the configuration manager, wfcmgr, and allows access to Citrix StoreFront or Program Neighborhood Agent services through the new self-service user interface (UI).
- **storebrowse** - This program is equivalent to the deprecated pnabrowse utility. It queries StoreFront or Program Neighborhood Agent services for virtual desktops and published applications.
- **wfica** - This program is the client engine that creates connections to the server and performs all of the functions of the connections.
- **Configuration files** - These files are designed like Windows .ini files and provide configuration information. The default files are located in the $ICAROOT/config/ directory. A user's .ini files are located in $HOME/.ICAClient.
- **Keyboard mapping files** - These files store the key mappings that allow Receiver for Linux to interpret keystrokes made on keyboards of various types and layouts.
- **Library files** - These shared library files control specific Receiver for Linux features such as security and smart card support.

- **Background processes (daemons)** - These provide functionality for several features such as StoreFront authentication, StoreFront connection, and USB redirection.
- **Helper processes** - These run when features such as HDX MediaStream Windows Media Redirection are active.
- **Utilities** - These are occasionally useful for checking system compatibility (hdxcheck.sh) or collecting information for Citrix Technical Support (lurdump).

Command-line utilities

selfservice replaces wfcmgr and is the command-line utility that displays the self-service UI.

storebrowse replaces pnabrowse. The latter is still available and is documented as part of this release, but it is deprecated and does not support the new features in this release. Citrix does not recommend using pnabrowse, unless necessary, to create or customize connections.

icabrowse is no longer available and is not documented as part of this release.

Authentication Manager

Authentication Manager is a new background process for Receiver that manages credentials with StoreFront.

A StoreFront server can at any time request credentials, which can take many forms. Authentication Manager is a long-lived daemon process that runs on the user device and is responsible for communicating with StoreFront. Authentication Manager can launch helper processes, when needed, to gather credentials from user input using a GTK+ interface that is shared with the main Receiver code. The Service Record daemon manages the relationship between stores and Authentication Manager by supplying the latter with configuration information.

storebrowse and selfservice communicate with Authentication Manager using a proprietary protocol.

Related components

Receiver deployments involve other Citrix components. These typically include XenDesktop, XenApp, StoreFront (which replaces Web Interface as the mechanism for publishing applications), and Secure Gateway or NetScaler® Gateway. Configuring and customizing these related components is not covered in this document. For information on each, see eDocs.

Chapter 2

Customize Receiver for Linux

Topics:

This section contains task-based procedures for customizing Receiver for Linux. Where possible, examples and context are provided as well as instructions for developing and configuring Receiver.

The following aspects can be customized:

- Installation
- User interface
- Security
- Multimedia
- Performance

Customize a Receiver for Linux installation

You can customize Receiver configuration before installation by modifying the contents of the Receiver package and then repackaging the files. Your changes will be included in every Receiver installed using the modified package.

Important: Connecting from 64-bit user devices running the self-service user interface or storebrowse to StoreFront servers may cause issues that affect the user experience adversely. See the topic *About this release* in eDocs for more information, including workarounds, for specific issues. If you experience any of these, Citrix recommends using Receiver for Web instead of StoreFront to launch connections.

To customize a Receiver for Linux installation

1. Expand the Receiver package file into an empty directory. The package file is called *platform.major.minor.release.build*.tar.gz (for example, linuxx86.13.0.0.*nnnnnn*.tar.gz for the Linux/x86 platform).
2. Make the required changes to the Receiver package. For example, you might add a new SSL root certificate to the package if you want to use a certificate from a Certificate Authority that is not part of the standard Receiver installation. To add a new SSL root certificate to the package, see the topic *Install root certificates on user devices* in eDocs. For more information on built-in certificates, see the topic *Configure and enable SSL and TLS* in eDocs.
3. Open the `PkgID` file.
4. Add the following line to indicate that the package was modified:

   `MODIFIED=`*`traceinfo`*

   where *traceinfo* is information indicating who made the change and when. The exact format of this information is not important.
5. Save and close the file.
6. Open the package file list, `platform/platform.psf` (for example, `linuxx86/linuxx86.psf` for the Linux/x86 platform).
7. Update the package file list to reflect the changes you made to the package. If you do not update this file, errors may occur when installing your new package. Changes could include updating the size of any files you modified, or adding new lines for any files you added to the package. The columns in the package file list are:
   - File type
   - Relative path
   - Sub-package (which should always be set to cor)
   - Permissions

- Owner
- Group
- Size

8. Save and close the file.
9. Use the `tar` command to rebuild Receiver package file, for example:
   `tar czf ../newpackage.tar.gz *`
   where *newpackage* is the name of the new Receiver package file.

User interface

This topic guides you through the steps for customizing the Receiver user interface (UI) and Receiver connections. This might require you to modify configuration files, run command-line utilities with options that you specify, or develop plug-ins.

In addition to the information presented here, consult the *User experience* topics in the Receiver for Linux section of eDocs.

Citrix provides a set of graphics assets that you can use to modify the Receiver UI in this release. To obtain these assets and a specification to help with the modifications, contact the Citrix Ready team.

Configuration files

About the configuration files

To change advanced or less common settings, you can modify Receiver's configuration files. These are read each time wfica starts. You can update various different files depending on the effect you want the changes to have.

Be aware that, if session sharing is enabled, an existing session might be used instead of a newly reconfigured one. This might cause the session to ignore changes you made in a configuration file.

Apply changes to all Receiver users

If you want the changes to apply to all Receiver users, modify the module.ini configuration file in the $ICAROOT/config directory.

Note: You do not need to add an entry to All_Regions.ini for a configuration value to be read from module.ini, unless you want to allow other configuration files to override the value in module.ini. If an entry in All_Regions.ini sets a default value, the value in module.ini is not used.

Apply changes to new Receiver users

If you want the changes to apply to all future new Receiver users, modify the configuration files in the $ICAROOT/config directory. For changes to apply to all connections, update wfclient.ini in this directory.

Apply changes to all connections for particular users

If you want the changes to apply to all connections for a particular user, modify the wfclient.ini file in that user's $HOME/.ICAClient directory. The settings in this file apply to future connections for that user.

Validate configuration file entries

If you want to limit the values for entries in wfclient.ini, you can specify allowed options or ranges of options in All_Regions.ini. See the All_Regions.ini file in the $ICAROOT/config directory for more information.

Note: If an entry appears in more than one configuration file, a value in wfclient.ini takes precedence over a value in module.ini.

About the parameters in the files

The parameters listed in each file are grouped into sections. Each section begins with a name in square brackets indicating parameters that belong together; for example, [ClientDrive] for parameters related to client drive mapping (CDM).

Defaults are automatically supplied for any missing parameters except where indicated. If a parameter is present but is not assigned a value, the default is automatically applied; for example, if InitialProgram is followed by an equal sign (=) but no value, the default (not to run a program after logging in) is applied.

Precedence

All_Regions.ini specifies which parameters can be set by other files. It can restrict values of parameters or set them exactly. If you want changes to apply to all Receiver users, modify module.ini.

For any given connection, the files are generally checked in the following order:

1. All_Regions.ini. Values in this file override those in:
   - The connection's .ica file
   - wfclient.ini
2. module.ini. Values in this file are used if they have not been set in All_Regions.ini, the connection's .ica file, or wfclient.ini but they are not restricted by entries in All_Regions.ini.

If no value is found in any of these files, the default in the Receiver code is used.

Note: There are exceptions to this order of precedence. For example, the code reads some values directly from wfclient.ini for security reasons, to ensure they are not set by a server.

Customize Receiver using storebrowse

You can customize Receiver by scripting the storebrowse command-line utility.

When used with Citrix StoreFront, storebrowse is equivalent to the deprecated pnabrowse utility. storebrowse takes options on the command line and returns results to its standard output, launches sessions, and so on.

storebrowse requires the libcurl package and libxml2 packages.

There is one mandatory argument, the URL of the store to connect to.

storebrowse uses the concept of a *resource name*. Unlike an application's display name, which can be duplicated, a resource name is unique. For example, there could be a Microsoft Outlook® display name in both an Office 2010 folder and an Office 2007 folder. Therefore, all operations such as launch take the resource name as the argument, and icons are stored with the resource name as the root of the file name. Resource names are long and not human-readable, but result in efficient scripts.

When entering a server address, you can omit the https:// or http:// prefix. storebrowse first tests the supplied URL as an HTTPS address and then, if that fails, as an HTTP address. StoreFront servers are not supported with the http:// prefix.

You can use an IP address instead of a FQDN for HTTP connections.

You can enter the FQDN if your store setup is a default one (if your StoreFront address is <storename>/Citrix/Store/discovery or if config.xml in a Program Neighborhood Agent setup is located in <storename>/Citrix/PNAgent/). You must enter the full URL in if your store setup is non-default.

For information on how storebrowse uses credentials, see Security on page 31.

To understand the command-line options that you can use with storebrowse, see the *Reference information* section of this document.

Using storebrowse with PNA servers

When connecting to a Program Neighborhood Agent (PNA) server, you can use storebrowse as a replacement for pnabrowse. storebrowse differs from pnabrowse in the following respects:

- Support for Kerberos passwords is withdrawn; the -k option is no longer accepted.
- Support for the old icabrowse utility has been removed. That is, the -A, -u, -p, and -c options are no longer accepted. The -S option is accepted but is now used to show subscribed applications on StoreFront servers.
- The -U, -P, and -D options are deprecated and may be removed in future releases. They work with Program Neighborhood Agent sites but are ignored by Storefront sites. Citrix recommends that you do not use these options and instead let the system prompt users for their credentials:
  - storebrowse launches a daemon process so that PNA credentials can be stored between calls. By default, this process terminates after one hour of the last call to storebrowse, at which point the credentials are deleted.
  - To configure a different timeout, create the file $ICAROOT/config/storebrowse.conf containing the required timeout in seconds followed by a newline. If the value zero is used, credentials are not stored for PNA sites (but the daemon process still runs).

    - You can terminate the daemon process early by calling storebrowse --killdaemon.
- Long versions of each option are now available. This allows scripts to be more readable. For example, --enumerate can be specified instead of -E.
- The -r option (long option --icaroot) now specifies the root directory of the Receiver installation.
- Only new icon syntax is supported, for example -i32x32 fetches 32-bit square icons at 32-bit depth.

Migrate to storebrowse

If you are migrating from a pnabrowse environment to a storebrowse one, the following information may help with any customizations that you make using that command-line utility:

- Adding and removing StoreFront stores is easy:
    - To add a store, users enter the URL of the StoreFront server or, if email-based account discovery is configured, they enter their email address. For information on email-based account discovery, see the StoreFront documentation in Citrix eDocs. The --addstore command returns the full store path that is used by storebrowse.
- StoreFront stores can now be used as sources of applications and desktops. Users can perform all of these tasks with storebrowse. These are new except for the -E and -L options that were also present in pnabrowse:
    - Add (using -a), delete (-d), and list (-l) stores.
    - List all of the desktops and applications in a store (using -E), and list all of these that the user has subscribed to (-S).
    - Subscribe to an application (using -s), and launch it (-L).
    - Change a store's default gateway (using -g).
- Subscribing to an application or desktop gives users control and reduces administration:
    - Once users are connected to the store, they can subscribe to desktops and applications in it; administrators do not have to handle subscriptions.
    - Subscriptions are stored locally, in Receiver, when connecting to a Program Neighborhood Agent server but remotely when connecting to a StoreFront server.
- Logons are handled differently with storebrowse:
    - Unlike pnabrowse, storebrowse lets Authentication Manager process logon prompts. For this reason, the -U, -P, and -D options are deprecated.
    - Authentication Manager prompts for credentials when necessary.

storebrowse examples

Add a store

The following command lines are alternative ways of adding a store.

```
./util/storebrowse -a 'my.examplestore.net'
```

```
./util/storebrowse --addstore 'https://
my.secondexamplestore.net/Citrix/Second/discovery'
```

Adding stores with storebrowse serves two purposes: it defines which stores can be used by the selfservice command, and it allows Service Record daemon, which is responsible for gateway management, to function correctly.

List stores

The following command lines list stores.

```
./util/storebrowse -l
```

```
./util/storebrowse --liststores
```

The output from both of these list commands is identical and might be as follows.

```
'https://my.examplestore.net/Citrix/Store/discovery'  'Store'
'149397992'  '"My Default GW",https://my.defaultgateway.com'
'"Alternative Gateway",https://
my.alternativegateway.com,"Alternative Gateway
2",my.alternativegateway2.com'
'https://my.secondexamplestore.net/Citrix/Second/discovery'
'Second'  '401460086'  '"Alternative Gateway",https://
my.alternativegateway.com'  '"My Default GW",https://
my.defaultgateway.com,"Alternative Gateway
2",my.alternativegateway2.com'
```

storebrowse lists stores in the following format, where \t is a Tab character.

```
'<store URL>'\t'<Store Name>'\t'<Unique Store ID>'\t'"<Current
Gateway Name>",<Current Gateway URL>'\t'"<Alternative Gateway
1 Unique Name>",<Alternative Gateway 1 URL>, … "<Alternative
Gateway n Name>",<Alternative Gateway n URL>'
```

Delete a store

The following command lines are alternative ways of deleting a store.

```
./util/storebrowse -d
'https://my.examplestore.net/Citrix/Store/discovery'
```

```
./util/storebrowse --deletestore
'https://my.examplestore.net/Citrix/Store/discovery'
```

Set a default gateway

The following example specifies the default gateway for a store. Gateways are points at which users outside an organization's firewall access a store. storebrowse (and the self-service UI) let you define the default gateway for a machine. For example, machines in two locations might access the same store through two different gateways.

```
./util/storebrowse --storegateway "Alternative Gateway"
'https://my.examplestore.net/Citrix/Store/discovery'
```

Enumerate resources on a Program Neighborhood Agent server

The following example command line enumerates all of the available resources on a Program Neighborhood Agent server. The server's URL is specified in the final argument. The command line outputs the default information and saves the 48-bit icon associated with the resource. The file name is part of the output.

```
storebrowse --enumerate --icons 48x https://my.example.net/
Citrix/Store/PNAgent/config.xml
```

Customize the self-service UI

You can customize the appearance of the self-service user interface (UI) in Receiver. Because this is based on the Receiver for Web, you can use that component's customization interface to modify the UI. For example, you can rebrand the UI by creating a new skin based on an alternative CSS and your own images.

Note: You cannot customize the logon dialog boxes in this way. Use the Receiver dialog library instead. For more information, see Dialog library on page 28.

Typically, you customize the contents of the following subfolders of $ICAROOT/site. These contain the Receiver for Web code, which is rendered by the self-service UI as its interface:

- **/contrib** - Customizable JavaScript and CSS files, which are documented in the comments of each file
- **/media** - Icons and other graphics

The following subfolders also exist, but you are unlikely to need to customize these:

- **/scripts** - Third-party JavaScript files, an obfuscated JavaScript file, and localized strings.
- **/css** - Third-party CSS files and an obfuscated CSS file. You cannot edit the files named Default_*.*.
- **/uiareas** - Site images.

To help modify the self-service UI, you can run the underlying web code in a standalone mode using a web browser. This lets you use standard web tools (for example, Firebug for Firefox) to inspect and modify the site. To run it in standalone mode, load the site $ICAROOT/site/selfservice.html?standalone or $ICAROOT/site/selfservice.html?standalonelogin in a browser. The former displays the main Self Selection view; the latter displays the logon screen for Shared User Mode.

For other information on customizations based on Receiver for Web, see CTX134791 and http://blogs.citrix.com/2012/06/06/customizing-receiver-for-web/.

Preferences

The Preferences UI in Receiver is implemented as a separate binary, $ICAROOT/util/configmgr, which edits the configuration files, and gets and sets values using storebrowse. For complex customizations, you can replace configmgr.

Alternatively, to make only small changes to the UI (for example, to limit the drive mappings that appear on the File Access page), you can edit rather than replace configmgr. You can modify the following pages in the Preferences dialog box by opening configmgr and editing the stated configuration options or storebrowse commands.

Note: Many of the configuration options were available in wfcmgr, which is no longer available. For more information on them than is provided here, consult an earlier version of this document.

General page

The General page uses the `UseFullScreen=True/False` setting in the [Thinwire3.0] section of wfclient.ini, and the following storebrowse commands.

```
--configselfservice ReconnectOnLogon=True/False
```

The setting ReconnectOnLogon corresponds to the **Reconnect apps and desktop: When I start Receiver** preference, and determines whether the self-service UI tries to reconnect to all sessions, for a given store, immediately after logon to that store.

```
--configselfservice ReconnectOnLaunchOrRefresh=True/False
```

The setting ReconnectOnLaunchOrRefresh corresponds to the **Reconnect apps and desktop: When I start or refresh apps** preference, and determines whether the self-service UI tries to reconnect to all sessions when an application is launched or the store is refreshed.

Accounts page

The Accounts page uses the following storebrowse commands to add, remove, and edit stores.

```
--addstore <store URL>
```

```
--deletestore <store URL>
```

```
--storegateway <gateway name>
```

If you have multiple stores, use the following command to define which one is displayed when the user first starts Receiver.

```
 ./util/storebrowse --configselfservice DefaultStore=<store
URL>
```

If users enter a partial store address (for example, my.store.net), Receiver tries to match it to one of the standard store address formats.

File Access page

The File Access page uses the following settings in the [WFClient] section in wfclient.ini to add, remove, and change read-write access to mapped drives. Replace the ? (question mark) with the letter of the drive that you want to map.

Setting	Description
CDMAllowed=True/False	Enables the client drive mapping feature. Mapped drives only appear in a session if this setting is enabled.
DrivePathKey?=/a/path	Sets the path (including drive) that you want to map. For example, to map P:\my \directory, configure this setting as follows: DrivePathKeyP=/my/directory
DriveEnabledKey?=True/False	Enables the specified drive.
DriveReadAccessKey?=0/1/2	Gives read access to the specified drive.
DriveWriteAccessKey?=0/1/2	Gives write access to the specified drive.

Mic & Webcam page

The Mic & Webcam page uses the setting `AllowAudioInput=True/False` in the [WFClient] section in wfclient.ini.

Flash page

The Flash page uses the `HDXFlashUseFlashRemoting` setting in the [WFClient] section in wfclient.ini.

Customize connections using the Platform Optimization SDK

Receiver connections can be customized by creating plug-ins to perform one or more of the following functions:

- Provide accelerated decoding of JPEG and H.264 data used to draw the session image

- Control the allocation of memory used to draw the session image
- Improve performance by taking control of the low-level drawing of the session image
- Provide graphics output and user input services for OS environments that do not support X11

You can develop plug-ins for decoding independently of the other types listed, unless they also need to control memory allocation. To test any plug-ins that you develop, you may need to rename them and you must copy them to the Receiver installation directory.

Receiver supports additional plug-ins for accelerated audio and video codecs, but no SDK is provided for these in this release. Receiver can also be configured to use GStreamer for webcam and multimedia functions. These plug-ins are standard GStreamer components and are not covered in this document.

Important: Plug-in development in a non-X-Window system might require a specialized toolkit and customization of the dialog library in the Receiver.

The following tables describe the files that you should be aware of when developing plug-ins with the Platform Optimization SDK. If Receiver cannot locate or use a file, the fallback file (where available) is used instead.

You can develop custom plug-ins from the following source files, which are supplied in this release.

File	Purpose	Fallback file	Notes
ctxjpeg.so	Citrix decoder for JPEG images	libjpeg Version 6: ctxjpeg_fb.so libjpeg Version 8: ctxjpeg_fb_8.so	The fallback decoder files are used only in ARM environments; the Receiver provides its own built-in fallback JPEG decoder in x86 environments. If you develop your own decoder, you must call it ctxjpeg.so.
ctxh264.so	Citrix decoder for H.264 images	ctxh264_fb.so	ctxh264.so decodes H.264 graphics only; HDX MediaStream for Windows Media and HDX MediaStream for Flash use different mechanisms to display H.264 video and movie content.
KVMEPlugin.so	Memory allocation	SOCX11plugin_COMPAT.so	The binary fallback file is only provided for ARM deployments. For x86 deployments, the source is

File	Purpose	Fallback file	Notes
			available and can be compiled. **Note:** KVMEPlugin.so is also used for screen drawing.

You can enable or configure some plug-ins using the following files (and additional system components). In these cases, no fallback files are employed and source files, for plug-in development, are not supplied.

File	Purpose	Notes
KVMEPlugin.so	Screen drawing	No fallback file is available, but a sample, SOCX11_plug.c, is included in this release. You can use this to develop a custom OpenGL implementation, for example. **Note:** KVMEPlugin.so is also used for memory allocation.
VORBIS.DLL	Decoder for non-speech audio data	You can use these files for standard audio (not HDX MediaStream Windows Media or HDX MediaStream for Flash). **Important:** Do not replace these files. When customizing the standard audio decoder, replace the libvorbis.so or libspeex.so library files instead. Any replacements must be API-compatible.
SPEEX.DLL	Decoder for speech audio data	
gst_read	A GStreamer utility required for HDX Realtime Webcam Video Compression	**Important:** Do not replace these files. For information on customizing these HDX features, see HDX RealTime Webcam Video Compression on page 39.
gst_play	A GStreamer utility required for HDX SpeedScreen Multimedia Acceleration	

File	Purpose	Notes
FlashContainer.bin	Provides support for HDX MediaStream Flash Redirection	For information on customizing this HDX feature, see Flash on page 35.

Plug-ins for H.264-based session graphics

For XenDesktop 7 and later, the preferred protocol for presenting the remote session's graphics uses a combination of H.264 and proprietary lossless graphics encoding. For maximum flexibility in exploiting on-chip decoders and hardware rendering support, plug-ins take full control of the decoding, overlay, and rendering process.

The details of the interface for these plug-ins are documented as comments in the associated header file, H264_decode.h. An unaccelerated sample implementation is included in the H264_sample directory.

Plug-ins for accelerated JPEG decoding

All currently supported versions of XenDesktop and Citrix XenApp® for UNIX® can use JPEG to compress portions of the session image. Plug-ins that support hardware-accelerated JPEG decoding can improve graphics performance for sessions not using H. 264 session graphics.

The details of the interface for these plug-ins are documented as comments in the associated header file, jpeg_decode.h. The sample code jpeg_sample demonstrates how wfica falls back when no accelerated plugin is available. It builds a plug-in called ctxjpeg_fb.so.

JPEG fallback is employed if necessary to ensure images are displayed efficiently on the user device. The following decoders are used in this order:

- On ARM platforms:
  a. ctxjpeg.so
  b. ctxjpeg_fb_8.so if Version 8 of libjpeg is present
  c. ctxjpeg_fb.so if Version 6 of libjpeg is present
- On x86 platforms:
  a. ctxjpeg.so
  b. The built-in decoder

Plug-ins for memory allocation

The following information may be useful if you want to hardware accelerate JPEG decoding, H.264 decoding, or screen drawing.

Hardware-accelerated plug-ins for H.264 or JPEG decoding may need to allocate memory buffers with special characteristics, for example using physically contiguous pages. A single plug-in component, KVMEPlugin.so, is used for both standard memory allocation and for drawing the session image. If you are using the plug-in for memory allocation, you must supply only two functions.

The header file for memory allocation plug-ins is mainloop.h. The two entry points that must be implemented are `special_allocate()` and `special_free()`. The example code is in the \allocation_sample directory. Before using this code as a model for your own plug-in, pay careful attention to the comments in the code. Parts of it are present only for backward compatibility with decoder plug-ins that were developed for obsolete versions of Receiver.

Plug-ins for faster drawing in X11 environments

In some environments using X11, drawing methods might be faster than the calls to `XShmPutImage()` that are used by default. You can implement KVMEPlugin.so using an alternative drawing method by providing the `draw()` entry point, which is used to send the session image to the screen. You can also provide the optional `draw_complete()` entry point. When these alternative entry points are used, you do not additionally have to implement the memory allocation functions.

The example code in the \allocation_sample directory includes an implementation that is almost identical to the default drawing code.

Plug-ins for non-X11 environments

The Platform Optimization SDK includes a separate version of the Receiver engine called wfica_plugin. This is not linked with any X11 libraries. The program requires a version of KVMEPlugin.so that provides video output, mouse and keyboard input, and timer and event detection services. The following features of the X11 version are not yet available in the separate version: clipboard, seamless windows, and multimedia and Flash support.

Two example plug-in implementations are included:

- SDL_plugin contains an implementation based on the SDL library.
- FB_plugin contains a version based on Linux system calls and device files. It uses the raw frame buffer for display.

Support for environments that use Simple DirectMedia Layer (SDL) depends on how the library is built. Usually, X11 and frame buffer graphics are supported. To use frame buffer graphics, run the program from a text console as a superuser, or change the permissions on the /dev/fb0 and /dev/mice files and then run it. The frame buffer plug-in needs access to these device files.

Dialog library

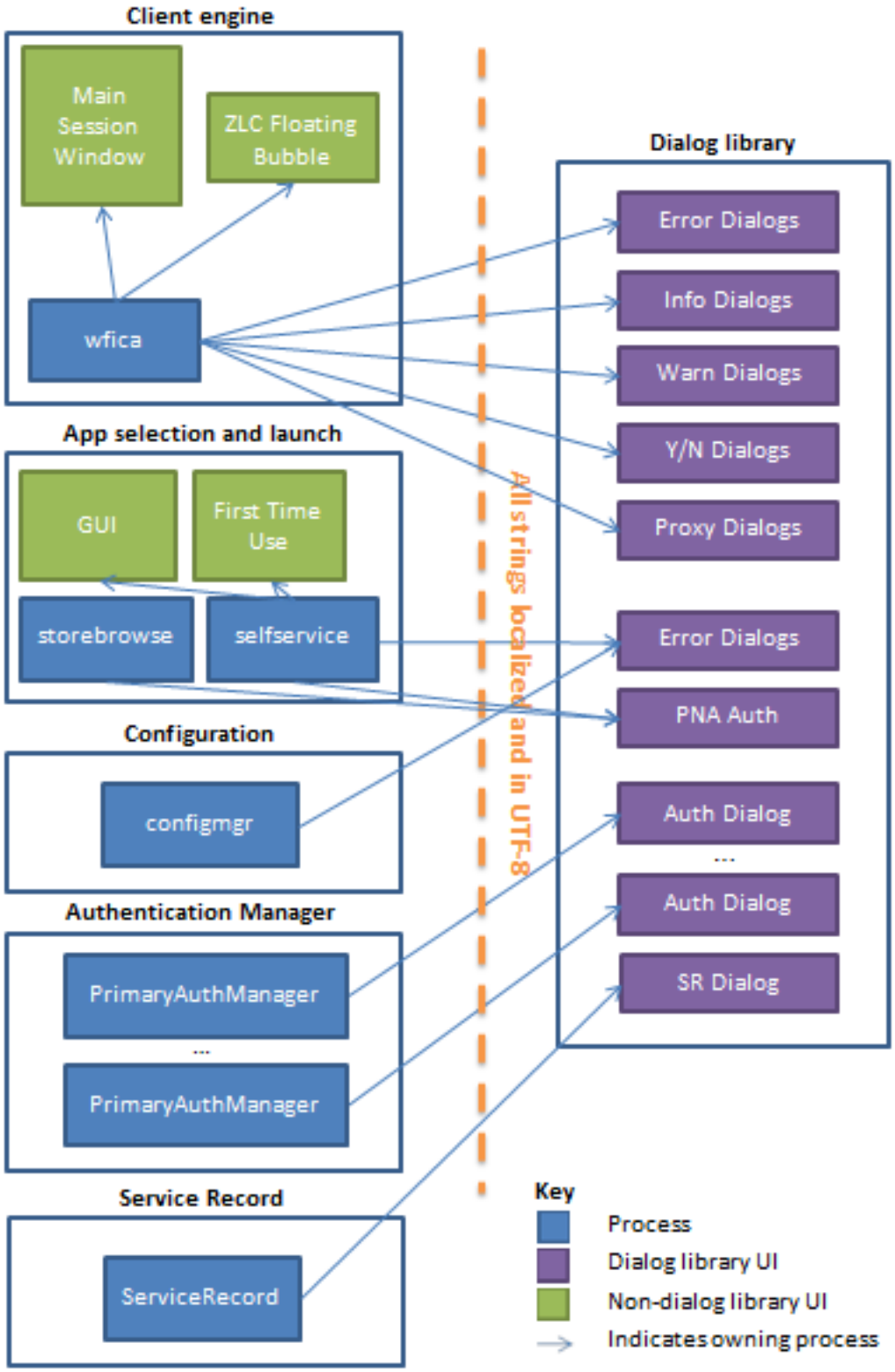
For alternative windowing systems to X Windows and their toolkits, you can develop customized dialogs using the Receiver for Linux dialog library described in this topic. The library is a C interface that can represent dialogs containing a selection of widgets: labels, text boxes, check boxes, radio buttons, combo boxes, multi-select combo boxes, buttons, and expanders. The library is loaded as a shared object file (UIDialogLib.so).

The dialog library is used for most of the dialogs that are displayed by Receiver for Linux processes, including the X11-based wfica. The processes storebrowse, AuthManager, PrimaryAuthManager, and ServiceRecord use it for all of their user

interface (UI). By re-implementing the library, you can replace the UI of these essential processes with a toolkit and event loop of your choosing. Except for dialogs, the remaining processes (selfservice, configmgr, and X11 wfica binaries) require GTK+ for other aspects of their UI, and therefore cannot be used with a different implementation of the library than the GTK+ implementation provided with Receiver. However, all of their functionality is available in the storebrowse command-line utility and the configuration files.

The following graphic represents the library's architecture and use by Receiver components.

For further documentation and examples to aid implementation of the API, refer to the Platform Optimization SDK.


Client engine
Main Session Window
ZLC Floating Bubble
wfica
App selection and launch
GUI
First Time Use
storebrowse
selfservice
Configuration
configmgr
Authentication Manager
PrimaryAuthManager
...
PrimaryAuthManager
Service Record
ServiceRecord
All strings localized and in UTF-8
Dialog library
Error Dialogs
Info Dialogs
Warn Dialogs
Y/N Dialogs
Proxy Dialogs
Error Dialogs
PNA Auth
Auth Dialog
...
Auth Dialog
SR Dialog
Key
Process
Dialog library UI
Non-dialog library UI
Indicates owning process

Security

Smart cards

Smart card deployments involving Receiver for Linux must adhere to the PC/SC standard.

The way you set up smart card deployments involving Receiver for Linux depends on the Citrix product.

Smart card authentication is not supported in Receiver for Linux deployments using StoreFront.

XenApp and XenDesktop 7

In XenApp and XenDesktop 7 deployments, users authenticate with one PIN entry to the server components (including Web Interface and Authentication Manager). They then start a session by authenticating with a second PIN entry. You set up these two authentication steps as follows.

To allow the first PIN entry, use the ctxipc command-line utility to notify the server when a user inserts or removes a smart card.

To allow the second PIN entry (once a connection to the server has been made), use the smart card virtual channel to give applications on the server access to a smart card on the Linux user device. Note the following:

- The Receiver library file VDSCARD.DLL contains functionality for smart card support and must be present on the user device.
- In module.ini:
  - The SmartCard setting must be On to enable smart card support
  - The [SmartCard] section lets you configure your smart card virtual driver
- In wfclient.ini:
  - DisableCtrlAltDel must be set to Off to enable smart card logons
  - ReaderStatusPollPeriod defines the smart card status polling period

XenDesktop 5.x and earlier

Smart card support is limited to the legacy use of Kerberos with Program Neighborhood Agent servers. For more information, see the *Using Kerberos with Citrix Receiver for Linux Guide*.

Certificates

By default, StoreFront sites use the HTTPS protocol. This is non-configurable.

Receiver recognizes a certificate as being from the correct certificate authority if a root certificate is installed in the $ICAROOT/keystore/cacerts directory.

To use SSL or TLS, you need a root certificate on the user device that can verify the signature of the Certificate Authority on the server certificate. By default, Receiver supports the following certificates.

Certificate	Issuing Authority
Class4PCA_G2_v2.pem	VeriSign Trust Network
Class3PCA_G2_v2.pem	VeriSign Trust Network
BTCTRoot.pem	Baltimore Cyber Trust Root
GTECTGlobalRoot.pem	GTE Cyber Trust Global Root
Pcs3ss_v4.pem	Class 3 Public Primary Certification Authority

You are not required to obtain and install root certificates on the user device to use the certificates from these Certificate Authorities. However, if you choose to use a different Certificate Authority, you must obtain and install a root certificate from the Certificate Authority on each user device.

Important: Receiver does not support keys of more than 4096 bits. You must ensure that the Certificate Authority root and intermediate certificates, and your server certificates, are a maximum of 4096 bits long.

Use a root certificate

If you need to authenticate a server certificate that was issued by a certificate authority and is not yet trusted by the user device, follow these instructions before adding a StoreFront store.

1. Obtain the root certificate in PEM format.

   Tip: If you cannot find a certificate in this format, use the openssl utility to convert a certificate in CRT format to a .pem file.

2. As the user who installed the package:
   a. Copy the file to $ICAROOT/keystore/cacerts.
   b. Run the following command as root:

```
c_rehash $ICAROOT/keystore/cacerts
```

Multimedia

This section contains information on customizing the way that Receiver processes:

- Graphics

- Video
- Audio

Graphics

XenDesktop and XenApp are based on different technologies, send different protocols to Receiver, and therefore require different configurations. Citrix recommends that you test Receiver with both of these products while you develop your solution.

Configure H.264 support

Receiver supports the display of H.264 graphics, including HDX 3D Pro graphics, that are served by XenDesktop 7. This support uses the deep compression codec feature, which is enabled by default. The feature provides better performance of rich and professional graphics applications on WAN networks compared with the JPEG codec.

Follow the instructions in this topic to disable the feature (and process graphics using the JPEG codec instead). You can also disable text tracking while still enabling deep compression codec support. This helps to reduce CPU costs while processing graphics that include complex images but relatively small amounts of text or non-critical text.

Important: To configure this feature, do not use any lossless setting in the XenDesktop **Visual quality** policy. If you do, H.264 encoding is disabled on the server and does not work in Receiver.

To disable deep compression codec support

In wfclient.ini, set **H264Enabled** to **False**. This also disables text tracking.

To disable text tracking

With deep compression codec support enabled, in wfclient.ini set **TextTrackingEnabled** to **False**.

To disable small frames support

The small frames feature allows efficient processing when only a small portion of the screen changes over time (for example, when a cursor flashes on an otherwise stable background). This procedure only works with XenDesktop 7.1 and overrides the equivalent setting in the Receiver for Linux SDK.

In wfclient.ini set **SmallFramesEnabled** to **False**.

Improve graphics performance with the Platform Optimization SDK

Using the Platform Optimization SDK, you can improve graphics performance (by accelerating the decoding of images, by controlling how memory is allocated when drawing an image, and so on). For information on this, see Customize connections using the Platform Optimization SDK on page 24.

Advanced graphic configurations

You can adjust how Receiver is configured to process graphics that are rendered on the server. Typically, these are bitmaps that are encoded using the JPEG protocol.

Input and output color formats

Most JPEGs are sub-sampled in YUV 4:2:0 format. However, the server can also send images in 4:4:4 format. Receiver expects ctxjpeg.so to output decoded JPEGs in 32-bit BGRX format, with the Blue component being the most significant eight bits.

The protocol used by Receiver does not restrict JPEG types, with the following exceptions:

- The protocol does not support JPEG2000
- The protocol does not use lossless JPEG
- The protocol does not use arithmetic encoding unless your ctxjpeg.so plugin indicates support for this in the decoder structure

The protocol use sequential encoding, rather than progressive or hierarchical encoding.

Citrix recommends sequential encoded, Huffman-compressed YUV 4:2:0 or YUV 4:4:4 images for hardware or DSP acceleration.

You can operate in the correct color format while decoding, to avoid the need to carry out color space conversion. However, this can be CPU-intensive. Alternatively, you can carry out the color space conversion in the hardware or DSP.

Custom memory allocation

You can adjust the memory allocation for graphics processing in:

- JPEG output buffers
- JPEG input buffers (also known as the *compressed image cache*)
- The session LVB
- Off-screen surfaces

If you develop a custom allocation mechanism, it replaces shared memory.

A sample, SOCX11_plug.c, is included in this release.

Sending decoded bitmaps to Xserver

You can hook the LVB allocation (source image data) function. When a frame is ready to be displayed, Receiver uses XShmPutImage to copy the LVB to screen. You may also need to hook the XShmPutImage function. If this is not convenient, alternative solutions (for example, using a non-atomic display) are available but they might degrade performance.

Calls to the Receiver constructor body

You can use the function pointer initialization for entry functions. This is in jpeg_decode.h. If GCC used, the Receiver library can use the

`__attribute__((constructor))` attribute to perform initialization. An example implementation of the JPEG SDK, defined in jpeg_decode.h, is available on request.

Advantages of CTXJPEG abstraction

In addition to hardware acceleration, abstracting CTXJPEG has these advantages:

- You can fully optimize JPEG decoding.
- You can allocate *special memory* for decoding purposes, which eliminates unnecessary memory copies and increases performance.
- You can save CPU. If you do not implement CTXJPEG, Receiver uses CTXJPEG_FB which in turn uses libjpeg, or libjpeg-turbo if NEON is available, to decode bitmaps. This means that JPEGs are decoded using software, which can be CPU intensive and can reduce performance (unless you provide API-compatible hardware replacements for either library).

Video

Flash

Citrix recommends that you develop your own Adobe Flash plug-in and that Flash files are played on an X Window system. For the ARM platform, you can obtain the necessary Flash libraries optimized from your Adobe scaling partner. Contact Adobe for more information on this.

HDX MediaStream Flash Redirection

The Citrix feature HDX MediaStream Flash Redirection uses a Citrix plug-in to send Flash content on websites to user devices. This lets Flash content run locally provided that Adobe Flash Player is installed on the device.

Important: This feature has not been tested on the ARM hard float (armhf) platform because, at the time of writing, no armhf platform with a Flash plugin is available.

The requirements for this feature are as follows:

- The NPAPI Flash plug-in and its dependent libraries must be present on the user device. A browser is not required but might be a convenient if it includes these plug-in and libraries.
- All NPAPI functions in the Flash plug-in must be Version 0-22 or earlier.
- The standard Flash function NPError Flash_EnforceLocalSecurity is required. A dummy function implementation which only returns NPERR_NO_ERROR should suffice as a minimum.
- Flash videos with resolutions less than 250 pixels in either the x or y dimension are rendered on the server by design.
- In some cases, HDX MediaStream Flash Redirection might only work when gliblc 2.10 is installed on the user device

Receiver searches in the following locations for the Citrix Flash plug-in, libflashplayer.so:

- /usr/lib/browser-plugins/
- /usr/lib/flashplugin-installer/
- /usr/lib/adobe-flashplugin/
- /usr/lib/mozilla/plugins/
- /usr/lib/opera/plugins/
- /usr/lib/flash-plugin/
- /usr/lib/firefox/plugins/
- /usr/lib/flashplugin-nonfree/
- $ICAROOT

If the plug-in is found in multiple locations, the plug-in with the latest version number is used by the HDX MediaStream Flash Redirection feature. If the plug-in is present in a different location, you can create a link to the location at $ICAROOT (the directory where Receiver for Linux is installed by default) using this command:

```
ln –s <target flash plugin location> libflashplayer.so
```

FlashContainer.bin runs on the device when the feature is active.

Test your Flash plug-in

Test your plug-in in the environment in which it will be used.

To check that Flash content is being rendered correctly on the user device, right-click in the Flash window. The Flash context menu displayed should appear similar to the native Linux Flash context menu.

You can also run the following command on the device to verify Flash content is being correctly rendered:

```
ps –ef |grep –i FlashContainer
```

Output similar to the following should be displayed:

```
 1000 6272 6240 0 15:41 pts/6 00:00:00 sh - c
/home/user/installation/icaclient/FlashContainer.bin
/tmp/Ctx1504387638977556438624O /tmp/Ctx5646687127620733126240
6240 0
1000 6273 6272 8 15:41 pts/6 00:00:02
/home/user/installation/icaclient/FlashContainer.bin
/tmp/Ctx1504387638977556438624O /tmp/Ctx5646687127620733126240
6240 0
```

Troubleshoot your Flash plug-in

You can collect trace logs to help debug your Flash plug-in. Run the following command and then test the feature using Receiver:

```
cat > $HOME/HDXFlash.ini <<EOM
[Tracing]
# enable/disable file tracing
File=1
# hex value
Flags=0x0FFFFFFF
# dec value
Level=9
EOM
```

The following logs are created in the /tmp/directory:

- CtxFlash_FlashContainer.bin_<*PID*>.log for the FlashContainer.bin process
- CtxFlash_wfica_<*PID*>.log for the wfica process

For more information on troubleshooting Flash, refer to CTX134786. If necessary, consider using HDX Windows Media Redirection instead of Flash. This is robust in different environments.

HDX MediaStream Windows Media Redirection

The HDX Mediastream Windows Media Redirection feature redirects audio and video content from the Microsoft® Media Foundation platform on the server to a local media player on the user device. Receiver uses the GStreamer pipeline to run streamed multimedia content on the device.

If a video codec is not available on the device or is not supported by HDX MediaStream Windows Media Redirection, it is processed by the server's media player. In these cases, video is delivered as server-rendered bitmaps through the graphics virtual channel. Depending on the audio quality settings, if an audio codec is not available on the device or is not supported by this feature, it is encoded on the server and sent to the device through the audio virtual channel.

If any of the following are missing, rendering takes place on the server:

- **On the server** - DirectShow or MediaFoundation components
- **On the user device** - GStreamer components
- **On the user device** - Appropriate entries in MediaStreamingConfig.tbl

HDX MediaStream Windows Media Redirection supports flow control and frame dropping because Receiver uses the GStreamer flow control mechanism for connections to XenDesktop.

Supported media players and formats

Supported media players, container formats, video codecs, and audio codecs are documented in CTX125211.

In addition, MediaStreamingConfig.tbl is a configurable text-based translation table that is located in $ICAROOT/config in the installation directory. This lists supported formats. Edit MediaStreamingConfig.tbl to add or remove support for client-side

rendering of media formats using the HDX MediaStream Windows Media Redirection feature. To locate the GUID of a media format in MediaStreamingConfig.tbl, use the verbose option `SpeedScreenMMAVerbose=True` in the [WFClient] section of wfclient.ini or in All_Regions.ini, and collect output from stdout for wfica.

Configure HDX MediaStream Windows Media Redirection

The following settings are located in module.ini in this release.

Item	Description
SpeedScreenMMAClosePlayerOnEOS=*Boolean*	Closes gst_play at the end of a media clip. This ensures only one gst_play process runs at a time. Default=False.
SpeedScreenMMAGstPlayKillAtExit=*Boolean*	Lets Receiver stop any gst_play processes that do not exit within a specified timeout period. Default=True.
SpeedScreenMMAGstPlayExitTimeout=*integer*	Period of time, in seconds, allowed for gst_play processes to exit before being terminated. Default=20.
SpeedScreenMMARebaseTimestampsOnSeek=*Boolean*	Enables rebasing of timestamps to a positive value following seek. Default=True.
SpeedScreenMMAStopOverlayHandlingEvents=*Boolean*	If set to False, fixes potential issues with videos not playing in the correct location or at the correct size, not resizing properly, or with the video window remaining black, but causes an issue where, after the mouse pointer has disappeared in full-screen Windows Media Player, it does not return when the mouse is moved. If set to True, corrects the mouse-pointer issue. Default=False.

Configure flow control

You can enable or disable flow control for HDX MediaStream Windows Media Redirection using XenDesktop policies. Flow control is enabled by default on the user device. To disable flow control on the device, set `SpeedScreenMMAFlowControlV3=False` in All_Regions.ini. This also disables frame dropping.

Troubleshoot HDX MediaStream Windows Media Redirection

To debug this feature on the user device, set SpeedScreenMMAVerbose=On in the [WFClient] section of the appropriate .ini file. To debug GStreamer behavior, see http://gstreamer.freedesktop.org/data/doc/gstreamer/head/gstreamer/html/gst-running.html.

Tip: GStreamer logging can adversely affect performance. Try finding a GStreamer trace that provides the necessary logging information, and then limit logging to that trace.

For information on troubleshooting this feature, see CTX104912.

HDX RealTime Webcam Video Compression

HDX RealTime Webcam Video Compression is the default mechanism for video conferencing applications. The video input is provided by the webcam to the user device and the application runs on the server. This feature lets webcam input on the device communicate with the application on the server.

You can specify how Receiver encodes webcam data. Both H.264 and Theora codecs are supported. By default, Theora encoding is enabled.

Important: To ensure this feature works, install any appropriate webcam drivers on the user device.

Theora encoding

Receiver uses GStreamer to encode webcam output on the user device using the Theora codec. This is theoraenc and is included in GStreamer's Base Plugins collection.

The following GStreamer pipeline is used for Theora encoding with HDX RealTime Webcam Video Compression:

v4l2src > ffmpegcolorspace > videoscale > capsfilter > theoraenc > appsink

By default, the resolution for the webcam output window is set to CIF/SIF(625): 352 × 288 and the frame rate is set to 15.

H.264 encoding

Receiver encodes webcam output in the H.264 format by choosing a pipeline in this order:

1. **HDXH264CaptureBin > appsink** - Receiver uses this option if you create and configure an HDXH264CaptureBin plug-in that is responsible for capturing and transcoding the webcam data. You might want to do so if the performance of GStreamer is unacceptable or if your chip has video acceleration capabilities.
2. **appsrc > appsink** - Receiver uses this option if the webcam supports H.264 and outputs H.264 data directly. It also requires HDXH264EnableNative to be set.
3. **v4l2src > encodebin > appsink** - Receiver uses this option if the webcam produces uncompressed output. The GStreamer elements that process this include v4l2src, which obtains data from the webcam's video driver, and encodebin, which

constructs a GStreamer pipeline for the H.264 encoder element that is present on the user device.

4. **v4l2src > jpegdec > encodebin > appsink** - Receiver uses this option if the webcam produces JPEG output rather than H.264 or another uncompressed format. This pipeline is not very efficient because it adds a decode step, jpegdec.

In each case, GStreamer Version 0.10.31 or any later release in the 0.10 series must be available on the user device.

If you choose a pipeline that uses encodebin and this cannot find the H.264 encoder, Theora encoding is used.

To configure H.264 support

1. If required, create an HDXH264CaptureBin.
2. In the [WFClient] section of the appropriate configuration file, set the following:
   - **HDXH264InputEnabled** - Set to **True**. By default, this is **False**, which enables Theora encoding.
   - **HDXH264CaptureBin** - If you created a plug-in, enter its name. By default, this is empty.
   - **HDXWebCamWidth** and **HDXWebCamHeight** - Set the width and height that define the webcam resolution. By default, HDXWebCamWidth is 352 pixels and HDXWebCamHeight is 288 pixels.
   - **HDXWebCamFramesPerSec** - Specify the preferred frame rate. By default, this is 15 frames per second.
   - **HDXWebCamDevice** - Enter the webcam name. By default, this is /dev/video0.

About the HDXH264CaptureBin plug-in

HDXH264CaptureBin is the customized plug-in that captures and transcodes webcam data, and that you create. The plug-in sends data to the GStreamer appsink plug-in, which has its capabilities set as follows:

```
caps_h264 =gst_caps_new_simple ("video/x-h264",
            "stream-format", G_TYPE_STRING, "byte-stream",
   "width", G_TYPE_INT, width,
   "height", G_TYPE_INT, height,
   "framerate", GST_TYPE_FRACTION, rate_num, rate_denom,
   "bpp", G_TYPE_INT, 16,
   "depth", G_TYPE_INT, 16,
   "endianness", G_TYPE_INT, G_BYTE_ORDER, NULL);
gst_app_sink_set_caps(GST_APP_SINK(appsink), caps_h264);
```

where `rate_num` is the value of **HDXWebCamFramesPerSec** in the configuration file, and `rate_denom` is fixed at 1.

If you create a plug-in, its capabilities must match these.

The plug-in must support a readable property, source, that returns the source element v4l2src. If multiple webcams are connected, this requirement ensures that a specific one can be selected.

The plug-in must support the properties device, num-buffers, and do-timestamp, as follows:

```
GObject *source;
/* get the source element from CaptureBin*/
g_object_get (G_OBJECT(CaptureBin),
"source", &source,
NULL);
// Set device properties on source i.e. v4l2src
g_object_set(source,
"device", device,
"num-buffers", num_buffers,
"do-timestamp", TRUE,
NULL);
g_object_unref (source);
```

For all other information on HDX RealTime Webcam Video Compression, see CTX132764.

Troubleshoot HDX RealTime Webcam Video Compression

To help debug the HDX RealTime Webcam Video Compression feature, you can wrap gst_read. The resulting script captures the standard output and error streams, stdout and stderr, and places them in /tmp/gst_read.log.

Run the commands in this procedure as the user who installed the client (usually, root).

1. From the util directory run the following command:

   ```
   mv gst_read gst_read.bin
   ```

2. Create a new file gst_read with the following lines:

   ```
   #!/bin/bash
   $ICAROOT/util/gst_read.bin -d $@ 2>&1 >/tmp/gst_read.log
   ```

   Important: Set $ICAROOT here even if you use the default location /opt/Citrix/ICAClient. If you do not, the script fails.

3. Set the file to be executable by running the following command:

   ```
   chmod +x gst_read
   ```

Apply custom properties to GStreamer elements for H.264 webcam support

In some configurations, you might need to apply custom properties to elements in the GStreamer pipeline. In these cases, Receiver tries to load a GStreamer preset called `Profile Citrix HDXH264WebCam` from .prs files that are stored in $ICAROOT/config/gstpresets (for GStreamer 0.10.36 or later) or in the default GStreamer location (for earlier versions).

For details of the .prs files' format, refer to your GStreamer documentation.

Webcams with native H.264 support

Because of the high bandwidth that is generated with the default settings on some webcams, native H.264 is turned off by default in Receiver. To enable support, configure the following setting in wfclient.ini:

```
HDXH264EnableNative=True
```

Audio

Audio input and output

Audio input consists of audio coming from the microphone on the user device that is redirected to an application on the server. This is mainly used with Voice-over-Internet-Protocol (VoIP) applications.

Audio output consists of any audio that is not redirected to the user device, for example audio from a server-rendered application such as Microsoft Outlook or audio from server-rendered media.

Configure Speex or Vorbis

If you are using standard audio (not HDX MediaStream Windows Media or HDX MediaStream for Flash), you can configure Receiver to process audio data using either the Speex or Vorbis codec. Speex is designed for speech audio data. Vorbis is designed for other types of audio data. Receiver uses the SPEEX.DLL library file to process Speex data and VORBIS.DLL to process Vorbis data.

When connections to virtual resources are negotiated (after installation but before runtime), the server sends one of the codecs to Receiver. The codec that is sent depends on your configuration of the AudioBandwidthLimit setting. This specifies the audio bandwidth limit and, by extension, the audio quality for the connection.

To configure Receiver to use Speex or Vorbis

Set AudioBandwidthLimit in the [WFClient] section of the appropriate .ini file or in the ICA file as follows:

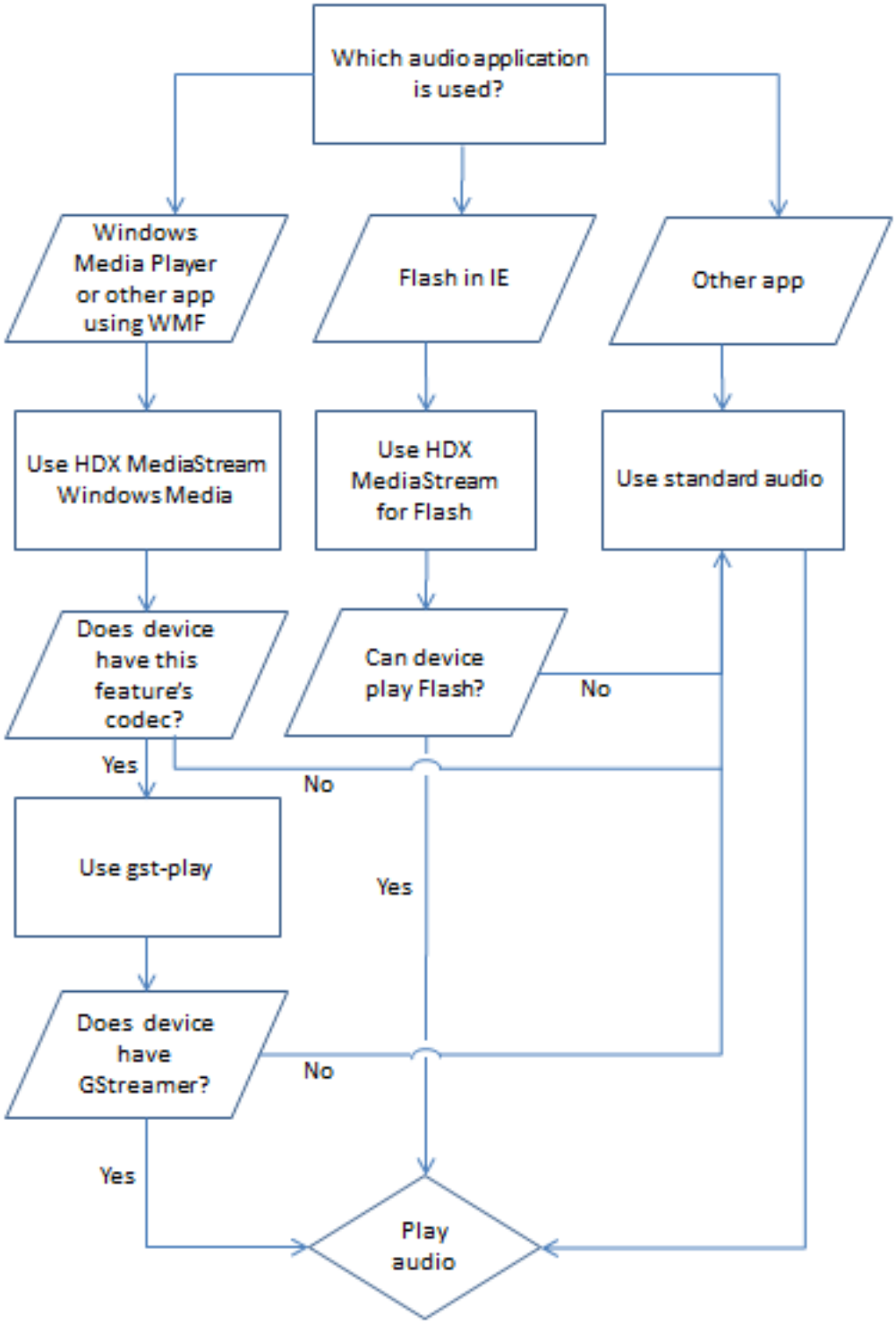
- 0 specifies the bandwidth as high and means the Vorbis codec is used
- 1 specifies the bandwidth as medium and means the Speex codec is used
- 2 specifies the bandwidth as low and means the Speex codec is used

Tip: You can override the configuration specified by the .ini file or ICA file by adding this setting to module.ini.

Which audio feature is used at runtime

The following diagram illustrates how different audio features are used at runtime. Receiver chooses the feature based on the audio application that runs on the user

device, and whether the correct codecs and plug-ins are available on it. Standard audio is used as a fallback if these are missing.



In this graphic, note the following:

- **WMF** - This stands for Windows Media Foundation.
- **Other app** - Other applications include the VLC Media Player and Audacity.
- **Does device have GStreamer?** - The presence of GStreamer is checked during installation. This determines if HDX MediaStream Windows Media can be used.

Consider GStreamer audio

GStreamer audio is an experimental feature in this release. Consider using it in your deployment but be aware of the limitations in doing so. For information on this feature, see GStreamer audio on page 52.

Enable audio input

You can enable standard audio input in two ways:

- With the HDX Realtime feature. Set `AllowAudioInput=True` in the [WFClient] section of /opt/Citrix/ICAClient/config/module.ini.
- With HDX MediaStream for Flash.

Test audio

To test whether audio is being rendered on the server, run an audio file in any player other than Windows Media Player.

Configure audio latency correction

If you have an Advanced Linux Sound Architecture (ALSA) implementation of VDCAM, you can control how audio latency in Receiver connections is processed. The audio redirection feature can detect periods of client overload and any delays in audio output. When client overload is detected, audio temporarily runs at a higher latency to increase the smoothness of the audio output. In periods of client stability, any excess latency is discarded to improve synchronization.

1. In the [ClientAudio] section of module.ini, enable the feature by setting AudioLatencyControlEnabled to True.
2. With AudioMaxLatency, set the maximum latency in milliseconds that Receiver waits before trying to discard audio data.
3. With AudioLatencyCorrectionInterval, define how often in milliseconds Receiver tries to correct the latency.
4. With PlaybackDelayThresh, specify the initial level of output buffering in milliseconds. Receiver tries to maintain this level of buffering throughout a session's duration.
5. With AudioTempLatencyBoost, specify the level for the higher latency band in milliseconds. This must be above the lower band set by PlaybackDelayThresh.

Performance

Memory

In environments where limited memory is a problem, you can minimize the amount of memory used by Receiver with the following parameters.

Item	Description
ClientToServerNormalPowerOf2=*integer*	Compression buffer size for reducer versions 2 and 3. Default=0 if data compression off, default=16 if on.
ServerToClientNormalPowerOf2=*integer*	Expansion buffer size for reducer versions 2 and 3. Default=0 if data compression off, default=18 if on.
Tw2CachePower=*integer*	Sets the size of the Thinwire 2 bitmap cache. Setting this lower than 19 (512KB cache) or higher than 25 is ineffective. The default value is calculated dynamically to be 1.25 times the screen image size at the preferred color depth.

You can also control the allocation of memory used to draw the session image by creating a plug-in with the Platform Optimization SDK. For information on this, see Customize connections using the Platform Optimization SDK on page 24.

Kiosk mode

You can configure a user device to start up in *kiosk mode*, in which Receiver starts automatically in full-screen mode when a user logs on to the device. This can be useful if users do not need to interact with the local operating system (OS) or any local applications. In this access scenario, Receiver effectively replaces the local OS, allowing the user to interact with virtual desktops and applications as if they were local. Together with the clearing of caches and credentials that Receiver performs, this lets you to use workstations as thin clients.

The user scenario in kiosk mode is:

1. The user starts their terminal.
2. A startup UI is displayed with just one object, a **Go** button.
3. The user clicks the button and is prompted for credentials.
4. Receiver starts the self-service UI in full-screen mode.
5. The user starts one or more applications.

6. When they have finished working at the terminal, the user clicks **Preferences** > **Log Off**.
7. Receiver clears its application caches and Authentication Manager clears the user's credentials.
8. Receiver closes the self-service UI and redisplays the startup UI, ready for the next user.

Set up kiosk mode

Setting up kiosk mode involves configuring the self-service UI as follows.

Important: If a terminal user needs to interact with the local OS or any local applications, do not make the window full-screen (FullscreenMode=1). In this scenario, or if you want the self-service UI windows to be displayed maximized and undecorated (FullscreenMode=2), Receiver does not cover the entire screen, so the user can interact with the environment in possibly unwanted ways. You should therefore take further steps to prevent this.

1. Set the desired values for the following settings. These are located in .ICAClient/cache/Stores/StoreCache.ctx:

Setting	Value	Notes
SharedUserMode	Boolean (True or False)	Indicates whether Shared User Mode is enabled. This allows the self-service UI to use one system user account for multiple users by removing user data from the device when users log off or close the UI.
FullscreenMode	Integer	Indicates whether and how the self-service UI window should appear full-screen: 0=the window is not displayed full-screen; 1=the window is displayed full-screen; 2=the window is displayed maximized and undecorated, which does not mask the desktop environment's taskbar. This can be useful users can launch seamless applications.

Setting	Value	Notes
		Default=0 (not full-screen).
SelfSelection	Boolean (True or False)	Used to disable the search box and the self-selection pane that appears on the left of the self-service UI when you click + (the plus sign). Disabling these UI elements prevents users from subscribing to extra applications.

These settings can alternatively be set using the `storebrowse -c` option or by editing the template file as described elsewhere in this topic.

2. Modify the device's boot sequence to replace the OS's user interface with the Receiver session.
3. Create the startup UI that prompts the user to log on to the session.
4. Use the selfservice command to launch the session.

Alternatives ways to configure the self-service UI for kiosk mode

Instead of editing the self-service UI settings for kiosk mode in StoreCache.ctx, you can alternatively use the storebrowse option `--configselfservice` (or, in short form, `-c`). This may be more convenient than editing the .ctx file directly.

To display the SharedUserMode setting, run:

```
./util/storebrowse --configselfservice SharedUserMode
```

To edit the SharedUserMode setting, run:

```
./util/storebrowse --configselfservice SharedUserMode=True
```

Note: The `-c` command is also used by configmgr (the Preferences UI)

Before any user has launched the self-service UI, you can also configure the settings by editing the default StoreCache.ctx-template file in $ICAROOT/config/. This file is renamed StoreCache.ctx and copied to users' .ICAClient/cache/Stores/ directory when selfservice or storebrowse are first launched. Editing the template lets you enable settings such as SharedUserMode and FullscreenMode for anyone who uses the system after your edits are saved.

Multi-threading

It can be useful to multi-thread connections to the Receiver in environments that contain multiple processors. Multi-threading support is on by default but you can turn it off.

By default, the Thinwire (Graphics) and Audio virtual channels run in their own thread. You can configure this using the following settings in module.ini:

```
 [Thinwire3.0]
 UseThread={TRUE, FALSE} // set to "TRUE" by default
ThreadQueueSize=65536    // Thread data queue size in bytes

[ClientAudio]
UseThread={TRUE, FALSE}     // set to "TRUE" by default
ThreadQueueSize=32768    // Thread data queue size in bytes
```

A larger queue means more buffering takes place and results in increased latency.

Monitor real-time performance

The following procedure applies to Receiver deployments involving XenApp or XenDesktop 7. It uses Citrix End User Experience Monitoring to monitor the following aspects of Receiver performance, in real time, in a desktop group:

- 2D graphics
- Playback of HDX MediaStream Windows Media
- Network data received within the session (except for UDP audio data)
- CPU used by the wfica program instance

Important: This monitoring feature is designed for OEM's in-house testing not for administrator's use in customer deployments.

1. On the user device, browse to the Receiver installation location by typing `cd /opt/Citrix/ICAClient` where `/opt/Citrix/ICAClient` is the installation location.
2. Run the following command:

   ```
   wfica –rm <options> <ICA file>.ica
   ```

   where `<options>` are any of the following options, and `<ICA file>.ica` is the connection you want to monitor. Options are case sensitive but can be provided in any order and in any combination, except for `D`, which is required to display results.

Option	Description
D	Displays real-time data on screen.
l	Logs data to file. This creates ica_instr.csv under $HOME directory. The same file is always used for logging, and is overwritten automatically if a new session is launched.
f	Frame rate (frames per second)
s	Screen response time
c	CPU usage
o	Data sent (kb per second)
d	Data received (kb per second)
j	JPEG decoding rate
r	RLE decoding rate
e	Direct Decode rate for JPEG
t	Text tracking rates for additions, deletions, H.264 objects, and lossless text
g	GStreamer frame rate overlay (frames per second) if HDX Windows Media Redirection is used

For example, the command `wfica -rm Dcf launch.ica` displays real-time CPU usage and frame rates for the connection created by launch.ica.

The performance data is displayed in an **ICA Metrics** dialog box.

Important: Click **Reset** in the dialog box to clear all parameters before starting each new test.

Monitor audio input and output using HDX Monitor or Perfmon

You can use HDX Monitor or Perfmon to monitor audio input to and output from the Virtual Delivery Agent (part of any XenDesktop deployment). Note that configuration details for these two monitoring components vary depending on the audio feature that is being used.

CPU frequency governor

The CPU frequency governor is an operating system component that allows the clock speed of processors to be adjusted on the fly.

On some systems (for example, ARM devices), the CPU frequency governor can influence the performance of Receiver. Specifically, you might notice that the frame rate alternates between low and high repeatedly when a 720p host-rendered video is played, or when some other, equally intensive activity is performed.

Furthermore, in *ondemand* mode the CPU frequency can change dynamically, based on the system load. In some cases, the frequency appears to be miscalculated in a given time period, resulting in a lower frequency being momentarily set. A low frame rate therefore results during that period.

Check that your CPU frequency governor functions correctly, or enforce a *performance* setting if consistent performance is required.

Flow control

With XenDesktop 7, Receiver can throttle session performance based on two factors, the available server-to-client bandwidth and the client processing load. With XenDesktop 7.1, this feature lets you control performance using a third factor, the client-to-server bandwidth. Client processing load is especially important for maintaining an optimal user experience on low-performance user devices by allowing a server of higher performance to match the devices' capabilities by dynamically adjusting its Thinwire frame rate. This avoids overloading devices, which in turn reduces session latency.

In XenDesktop 7 this feature is disabled by default on the server, but in Version 7.1 it is enabled by default so, depending on the performance capabilities of your user devices, you may want to disable it in Receiver.

Note: This feature is separate to the flow control feature for HDX MediaStream Windows Media Redirection, which is described elsewhere in this document.

To disable flow control in Receiver

In wfclient.ini, set **FlowControlEnabled** to **False**.

Chapter 3

Experimental features

Topics:

The following experimental features are available in this release:

- GStreamer audio

Configuration information and any limitations on the features' use is provided in this section of the guide.

GStreamer audio

Switch to GStreamer audio

You can switch between your existing implementation and an implementation based on GStreamer, as follows.

Important: Be aware that GStreamer audio has limitations that are described elsewhere in this section.

If wfica encodes or decodes audio using CPU, VDCAM.DLL is loaded. In a GStreamer implementation, ensure that VDGSTCAM.DLL is loaded instead by editing the [ClientAudio] section of module.ini to include the appropriate driver name:

- **VDCAM.DLL** - DriverName=VDCAM.DLL
- **VDGSTCAM.DLL** - DriverName=VDGSTCAM.DLL

If VDGSTCAM is used for audio output, gst_aud_play passes the encoded audio stream to GStreamer with the correct codec information to decode the stream. For audio input, gst_aud_read reads the encoded audio stream from GStreamer.

The input and output pipelines are as follows:

- **Speex audio input** - autoaudiosrc > audioconvert > speexenc > appsink
- **Vorbis audio input** - autoaudiosrc > audioconvert > vorbisenc > oggmux > appsink
- **Speex audio output** - appsrc > speexdec > audioconvert > autoaudiosink
- **Vorbix audio output** - appsrc > oggdemux > vorbisdec > audioconvert > autoaudiosink

Optimize GStreamer audio

If you have set up Receiver to use the GStreamer framework, you can modify how audio is processed by it using the following settings in the [ClientAudio] section of the appropriate configuration file.

Before using this procedure, you should be familiar the GStreamer SDK, including the concepts of audio sinks and audio sources.

1. Reduce GStreamer startup time by setting:
    - **GSTAudioSinkName** - This is the GStreamer element that you want to use as the sink for audio. By default, this is **autoaudiosink**, the automatically detected audio sink.
    - **GSTAudioSrcName** - This is the GStreamer element that you want to use as the source for audio. By default, this is **autoaudiosrc**, the automatically detected audio source.

2. In **GSTSpeexBufferingLatency**, specify the amount of additional output buffering when rendering audio in Speex format. The default is 50 ms.
3. In **GSTVorbisBufferingLatency**, specify the amount of additional output buffering when rendering audio in Vorbis format. The default is 150 ms.

Configure GStreamer in non-default locations

If GStreamer is installed in a non-default location (for example, /gstreamer), you must make the following changes in addition to adjusting the configuration file.

1. Allow $ICAROOT/util/gst_play to link with GStreamer:

```
ldconfig /gstreamer/lib
```

2. In $ICAROOT, rename selfservice to selfservice.real.
3. Create a shell script wrapper called selfservice and set an environment variable that GStreamer uses to locate its plug-ins:

```
#!/bin/sh
export GST_PLUGIN_SYSTEM_PATH=/gstreamer/lib
exec /opt/Citrix /ICAClient/selfservice.real $@
```

Note: You can also create a similar wrapper for storebrowse.

GStreamer audio limitations

Bear the following limitations in mind when implementing GStreamer audio.

armhf platform

No GStreamer processes run on the user device on the ARM hard float (armhf) platform. As a result, the server feature HDX Windows Multimedia Redirection, which relies on GStreamer, is not supported on this platform unless you are running Ubuntu 12.10 or later.

Recording

Audio input is not always functional on some user devices. Symptoms include an inability to record more than once or twice, an inability to record any input, and distorted input. There is no known workaround for this issue.

Pausing and resuming

Some audio software does not stop and restart the audio device when users pause, and then try to restart, playback. The software might also not issue any data to the device during silence. Both of these symptoms prevent new audio data from being played. Use the following setting to mitigate this problem.

In the [ClientAudio] section of module.ini, set **GSTUseNoClock** to **True**. This disables the built-in clock in GStreamer.

A disadvantage of using this setting is that any gaps in the audio that are caused by network outages result in permanently increased latency, since they are no longer detected.

Alternatives to GStreamer audio

The advantages of using GStreamer audio can in some cases be achieved in other ways:

- **Playback in a separate thread** - GStreamer lets you decode and play audio in a new process on a separate CPU. You can also achieve this without using GStreamer by ensuring that VDCAM.DLL, rather than VDGSTCAM.DLL, is loaded.
- **Hardware acceleration** - GStreamer lets you encode and decode audio using hardware. You can also achieve this without GStreamer by replacing the supplied libvorbis.so or libspeex.so library files with your own. Any replacements must be API-compatible.

Chapter 4

Reference information

Topics:

This section contains reference information on the following items that ship with this release:

- Command-line utilities
- Configuration files
- Library files
- Scripts

Command-line utilities

The tables below list Receiver for Linux command-line parameters. A list of the parameters can be obtained typing `wfica` or `storebrowse` with the `-?`, `-help`, or `-h` options.

wfica

You can use a connection file simply by typing its name after wfica without any of the options below.

To	Type
Specify the custom connection to use from the Connection file. **Note:** With the new self-service UI, you cannot set up a custom connection in this way.	`-desc` *`description`*
	`-description` *`description`*
Specify a connection file.	`-file connection` *`filename`*
Set alternative protocol file. This enables the use of an alternative module.ini.	`-protocolfile` *`filename`*
Set alternative client configuration file. This enables the use of an alternative wfclient.ini.	`-clientfile` *`filename`*
Display a different name for Receiver, specified by name, wherever that name appears. The default name is the device name. However if you use a Sunray device, the default name is derived from the device's MAC address. This is overridden by the ClientName entry in .ICAClient/wfclient.ini, which is itself overridden by issuing the -clientname name command.	`-clientname` *`name`*
Show this list of parameters.	`-help`
Display version information.	`-version`

To	Type
Show error numbers and string.	`-errno`
Set the location of Receiver installation files. This is equivalent to setting the ICAROOT environment variable.	`-icaroot` *`directory`*
Suppress connection dialogs.	`-quiet`
Log connection process.	`-log`
Enable key logging.	`-keylog`
Set session geometry.	`-geometry WxH+X+Y`
Set color depth.	`-depth <4 | 8 | 16 | 24 | auto>`
Set monitor spanning.	`-span [h][o][a| mon1[,mon2[,mon3,mon4]]]`
Use private colormap.	`-private`
Use shared colormap.	`-shared`
Specify a string to be added to a published application.	`-param` *`string`*
Specify the UNIX path to be accessed through client drive mapping by a published application.	`-fileparam` *`unixpath`*
Specify a user name.	`-username` *`username`*
Specify a disguised password.	`-password` *`password`*
Specify a clear text password.	`-clearpassword "`*`clear password`*`"`
Specify a domain.	`-domain` *`domain`*
Specify an initial program.	`-program` *`program`*
Specify a directory for the initial program to use.	`-directory` *`directory`*

To	Type
Turn on sound.	`-sound`
Turn off sound.	`-nosound`
Set drive mapping overrides. These are of the form A$=path, where path can contain an environment variable (for example A$=$HOME/tmp). This option must be repeated for each drive to be overridden. For the override to work, there must be an existing mapping, though it need not be enabled.	`-drivemap` *`string`*
Associate document with published application.	`-associate`
Only launch the associated published application. Do not open the document.	`-launchapponly`

Tip: All wfica command line options can also be specified in the environment variable WFICA_OPTS, allowing them to be used with Receiver's native user interface or with Citrix StoreFront.

storebrowse

The following table documents the options that you can use with the storebrowse utility.

Option	Description	Notes
`-L, --launch`	Specifies the name of the published resource to which you want to connect. This launches a connection to a published resource. The utility then terminates, leaving a successfully connected session.	
`-E, --enumerate`	Enumerates the available resources.	By default, the resource name, display name, and folder of the resource are displayed. Additional

Option	Description	Notes
		information can be displayed, by using the `--details` option.
`-S, --subscribed`	Lists the subscribed resources.	By default, the resource name, display name, and folder of the resource are displayed. Additional information can be displayed using the `--details` option.
`-M, --details` Use in conjunction with the `-E` or `-S` option.	Selects which attributes of published applications are returned. This option takes an argument that is the sum of the numbers corresponding to the required details: Publisher(0x1), VideoType(0x2), SoundType(0x4), AppInStartMenu(0x8), AppOnDesktop(0x10), AppIsDesktop(0x20), AppIsDisabled(0x40), WindowType(0x80), WindowScale(0x100), and DisplayName(0x200). CreateShortcuts(0x100000) can be used in conjunction with -S, -s, and -u to create menu entries for subscribed applications. RemoveShortcuts(0x200000) can be used with -S to delete all menu entries.	Some of these details are not available through storebrowse. If this is the case, the output is 0. Values can also be expressed in decimal as well as hexadecimal(for example, 512 for 0x200).
`-v, --version`	Writes the version number of storebrowse to the standard output.	
`-?, -h, --help`	Lists the usage for storebrowse.	An abbreviated version of this table is displayed.
`-U, --username`	Passes the user name to the server.	These options are deprecated and may be removed in future

Option	Description	Notes
`-P, --password`	Passes the password to the server.	releases. They work with Program Neighborhood Agent sites but are ignored by StoreFront sites. Citrix recommends that you do not use these options and instead let the system prompt users for their credentials.
`-D, --domain`	Passes the domain to the server.	
`-r, --icaroot`	Specifies the root directory of the Receiver for Linux installation.	If not specified, the value is determined at run time.
`-i,--icons` Use in conjunction with the `-E` or `-S` option.	Fetches desktop or application icons, in PNG format, of the size and depth given by the `best` or `size` argument. If the `best` argument is used, the best sized icon available on the server is fetched. You can convert this to any size required. The `best` argument is the most efficient for storage and bandwidth, and can simplify scripting. If the `size` argument is used, an icon is fetched of the specified size and depth. In both cases, icons are saved in a file for each of the resources that the -E or -S option returns.	The `best` argument creates an icon of the form <resource name>.png. The `size` argument is of the form WxB, where W is the width of the icon (all icons are square, so only one value is needed to specify the size), and B is the depth (that is, the number of bits per pixel). W is required but B is optional. If it is not specified, icons of all available image depths are fetched for that size. The files that are created are named <resource name>_WxWxB.png.
`-u, --unsubscribe`	Unsubscribes the specified resource from the given store.	
`-s, --subscribe`	Subscribes the specified resource from the given store.	If you use a different Receiver, subscriptions on Program Neighborhood Agent servers are lost.

Option	Description	Notes
`-W [r|R], --reconnect [r|R]`	Reconnects disconnected and active sessions.	`r` reconnects all disconnected sessions for the user. `R` reconnects all active and disconnected sessions.
`-WD, --disconnect`	Disconnects all sessions.	Only affects sessions to the store specified on the command line.
`-WT, --logoff`	Logs off all sessions.	Only affects sessions to the store specified on the command line.
`-l, --liststores`	Lists the known StoreFront stores, that is those that storebrowse can contact. These are the stores registered with the ServiceRecord proxy. Also lists Program Neighborhood sites.	You can connect to any store, but if you have previously added a store using the `--addstore` command, Receiver can use the store location and gateway details to form connections.
`-a, --addstore`	Registers a new store, including its gateway and beacon details, with the Service Record daemon.	Returns the full URL of the store. If this fails, an error is reported.
`-g, --storegateway`	Sets the default gateway for a store that is already registered with the Service Record daemon.	This command takes the following form: `./util/storebrowse --storegateway "<unique gateway name>" '<store URL>'` **Important:** The unique gateway name must be in the list of gateways for the specified store.
`-d, --deletestore`	Deregisters a store with the Service Record daemon.	
`-c, --configselfservice`	Gets and sets the self-service UI settings that are stored in StoreCache.ctx. Takes an argument of the	Example: `storebrowse --configselfservice SharedUserMode=True`

Option	Description	Notes
	form `<entry[=value]>`. If only entry is present, the setting's current value is printed. If a value is present, it is used to configure the setting.	**Important:** Both entry and value are case sensitive. Commands that use this option will fail if the case is different to the documented case of the setting itself (in StoreCache.ctx).
`-C, --addCR`	Reads the provided Citrix Receiver (CR) file, and prompts the user to add each store.	The output is the same as -a but might contain more than one store, separated by newlines.
`-K, --killdaemon`	Terminates the storebrowse daemon process.	Deletes any stored Program Neighborhood Agent site credentials.

pnabrowse

Important: The pnabrowse utility is deprecated but can still query Program Neighborhood Agent sites for lists of servers and published resources, and lets you connect to a published resource. Citrix discourages the use of pnabrowse because it prevents users from accessing StoreFront stores; use storebrowse instead. storebrowse can prompt for credentials from both sites and stores. The -U, -P and -D options only work with Program Neighborhood Agent sites.

An optional argument of pnabrowse specifies the server to connect to. This may be either:

- The name of the XenApp server, for options -S and -A.
- The URL of the server running a Program Neighborhood Agent site, for options -E and -L.

The pnabrowse utility returns an exit value indicating success or failure, and uses the following options:

Option	Description
-S	List servers, one per line.
-A	List published applications, one per line.

Option	Description
-m	Used in conjunction with -A, this expands the information returned about published applications to include Publisher,Video Type, Sound Type, AppInStartMenu, AppOnDesktop, AppIsDesktop, AppIsDisabled, Window Type, WindowScale, and Display Name.
-M	Used in conjunction with -A, this selects individual columns of information returned about published applications. It takes a argument (1-1023) which is the sum of the numbers corresponding to the required details: Publisher(1), Video Type(2), Sound Type(4), AppInStartMenu(8), AppOnDesktop(16), AppIsDesktop(32), AppIsDisabled(64), Window Type(128), Window Scale(256), and DisplayName(512).
-c	When appended to option -A, create files specifying the minimum information the client engine needs to connect to published applications; for example, application name, browse server, window resolution, color depth, audio, and encryption settings. File names are formatted as follows: /tmp/xxx_1.ica, /tmp/xxx_2.ica where xxx is replaced by the decimal process identifier for the pnabrowse process.
-i	Include paths to files containing icon images for published applications in the output from option -A. Either .xpm or .png files are returned depending on the use of the size (WxB) option: ◆ -i returns 16x16 icons in XPM format at 4 bits per pixel ◆ -iWxB returns WxW icons in PNG format at B bits per pixel
-f	Include Citrix XenApp folder names for published applications in the output from option -A.

Option	Description
-u	Specify a user name for authenticating the user to a proxy server.
-p	Specify a password for authenticating the user to a proxy server.

The following options provide both XenApp and XenDesktop functionality:

Option	Description
-D	Specify a domain for authenticating the user to the server running the Web Interface or the server running the Citrix XenApp (Program Neighborhood Agent) Service.
-E	Invoke Citrix XenApp and enumerate all published resources. If you specify both -E and -L, the last option on the command line takes effect. The utility then terminates, possibly leaving a connection open. For each resource the following details are written to standard output, enclosed in single quotation marks and separated by tab characters: Name: The display name from the Access Management Console Application Properties dialog box. Folder: The Program Neighborhood folder from the Access Management Console Application Properties dialog box. Type: Either Application or Content. Icon: The full path name of an .xpm format icon file.
-L	Specify the name of the published resource to which you want to connect. This invokes Citrix XenApp and launches a connection to a published resource. If you specify both -E and-L, the last option on the command line takes effect. The

Option	Description
	utility then terminates, possibly leaving a connection open.
-N	Specify a new password. This option must be used with existing credentials and is valid only when the existing password has expired, as indicated by the exit code 238: E_PASSWORD_EXPIRED. This option can be given with other options, such as -E or -L.
-P	Specify a password for authenticating the user to the server running the Web Interface or the server running the Citrix XenApp (Program Neighborhood Agent) Service.
-U	Specify a user name for authenticating the user to the server running the Web Interface or the server running the Citrix XenApp (Program Neighborhood Agent) Service.
-WD	Disconnects all active sessions for the user.
-WT	Terminates all sessions for the user.
-Wr	Reconnects to all disconnected sessions for the user.
-WR	Reconnects to all sessions (active or disconnected) for the user.
-k	Use an existing Kerberos ticket to authenticate, rather than user name, password, and domain. This requires configuration of the client and server. For more information, see the *Using Kerberos with Citrix Receiver for Linux* documentation.

The following common options are used:

Option	Description
-q	Quiet mode; do not print error messages.
-r	Include raw icon data for published applications in the output from options -E or -A.
-h	Print a usage message listing the options.
-?	Print a usage message listing the options.

Exit Status values

The command-line utilities storebrowse and pnabrowse report exit status values to indicate success or failure. If problems arise, these values give guidance on possible error causes and their meanings, and are listed in the following table.

Note that some error conditions may result in different exit values depending on which part of the code detects them.

Value	Description
0	Success
1-238	These error codes are associated with common error messages. Run wfica -errno for a list of these messages.
246	Citrix XenApp has reported an error. See the text written to standard output for more information on this error.
247	A published resource has not been recognized.
248	Invalid credentials.
249	Failed to enumerate servers.
250	Failed to make a directory.
251	Failed to load an .ini file.

Value	Description
252	No Web Interface server was specified.
253	No Program Neighborhood Agent server was specified.
254	A parameter is missing
255	Execution failed

When storebrowse or pnabrowse fails to change a password, the exit code can be useful in diagnosing the problem. For example:

```
0 SUCCESS

4 E_MISSING_ARG

63 E_NOT_ALLOWED WI configuration prohibits change

65 E_NOT_SUPPORTED Could be seen if :-
-    WI config requires "direct connection" (=Kerberos), but
couldn't load Kerberos library
-    Support is not compiled into client - might see it with
pre 11.114 version
-    Trying to change a Novell password

74 E_NEW_PASSWORD_INVALID

248 EX_INVALID_CREDENTIALS

255 EX_EXEC_FAILED Some problem with the server changing the
password, such as it hasn't expired.
```

Configuration files

For any given connection, the configuration files are checked in a specific order. For information on this, see Configuration files on page 17.

wfclient.ini

This .ini file contains a section for parameters specific to the Receiver user interface such as version number and desired resolution.

In Version 10.x and later of Receiver for Linux, for each entry in wfclient.ini there must be a corresponding entry in All_Regions.ini for the setting to take effect. In addition, for each entry in the [Thinwire3.0], [ClientDrive], and [TCP/IP] sections of wfclient.ini, there must be a corresponding entry in canonicalization.ini for the setting to take

effect. See the All_Regions.ini and canonicalization.ini files in the $ICAROOT/config/ directory for more information.

Parameter syntax

Boolean parameters use Yes, True, 1, or On to indicate TRUE. Any other values, including No, False, 0, or Off, are interpreted as FALSE.

For all parameters, spaces are significant and values are case-sensitive.

Parameters that can be modified from the user interface are in **this typeface**. Parameters that cannot be modified from the user interface, but are read by the client, are in normal typeface. Those marked as ignored are not currently used by the client, but can be reserved for future use, redundant, or used by other clients; for example, Win32 or Macintosh. In the last case, the parameter is read by the client but the result discarded.

Default values are embedded into the client program itself. Fixed values are set by the unmodified .ini configuration files.

In the following table, the parameters are listed alphabetically within each section of the file.

Item	Description
[WFClient]	This section is used by the engine. It contains default session-oriented parameters.
AllowAudioInput=boolean	To enable Webcam and audio input for connections you must ensure this parameter is set to True, otherwise it overrides the setting for the EnableAudioInput parameter in appsrv.ini. Default=False.
ApplySucConnTimeoutToDesktops=boolean	Works with the SucConnTimeout setting. Ensures that the setting SucConnTimeout is honored by virtual desktops as well as virtual applications. When ApplySucConnTimeoutToDesktops is applied to desktops, repeated clicks launch multiple sessions, but you can set SucConnTimeout to a suitable timeout and run a custom script in between the desktop launches. Default=False
AutoResponse=integer	Specifies a bitmapped value that enables automatic response to user prompts

Item	Description
	(such as dialog boxes). The values are as follows: 1: log message to standard error output 2: exit program whenever that choice is offered. Default=0 (wait for user response).
BufferLength=integer	Input buffer length. Default=2048.
BypassSetLED=boolean	Prevents virtual applications running macros multiple times. When a virtual application runs a macro on one of the LED key presses (that is on the Caps Lock, Number Lock, or Scroll Lock key), the application expects the key state to be sent once. However, the macro runs multiple times and sends the state each time. Default=False
ClientComm=[On|Off]	Determines if Client COM Port Mapping is on. Default=On.
ClientName=string	Allows client name to be overridden; normally this is obtained from the system. Default=none (no override).
ClientPrinterList=string	Allow specified named printers to be used; for example, lp1:laser1:lp2. No default (obtain printer list from client OS).
ClientUnicodeEnabled=boolean	Client can use UNICODE. Default=True.
ComPort1..99=string	COM port device name. No default.
ContinueWithoutPDALockFile=boolean	Allows a connection to a Pocket PC, even if the user has insufficient permission to create a lockfile.

Item	Description
CRBrowserCommand=string	Indicates the command used to request the display in an existing browser or start a new browser from those listed. This command is executed after appending the URL. Default= nslaunch firefox, mozilla, iceweasel.
CRBrowserPath=string	Server to client content redirection browser path. No default. Use $PATH. Obsolete.
CREnabled=boolean	Server to client content redirection enabled. Default=True.
CRPlayerCommand=string	Server to client content redirection media player command. Default=realplay %s. Obsolete.
CRPlayerPath=string	Server to client content redirection media player path. No default. Use $PATH. Obsolete.
CursorStipple=hex_integer,hex_integer	Defines a stipple pattern in cursor masks to replace inversion regions in Windows cursors. Default=aaaa,5555.
DefaultPrinter=string	Print queue to be used as the default printer in the Citrix XenApp session. For more information, see the Receiver for Linux topics in eDocs.
DefaultPrinterDriver=string	Printer driver to be used for the default printer in Citrix XenApp sessions on Windows. For more information, see the Receiver for Linux topics in eDocs.
DeferredUpdateMode=boolean	Enables batched updates from the Local Video Buffer (LVB) to the screen. The LVB is used when seamless windows or Speedscreen Latency Reduction are in use and for 256-color connections when specified by the UseSDLVB parameter.

Item	Description
	Default=False.
DisableClientAutoQuit=boolean	Quit client on disconnect. Ignored.
DisableCtrlAltDel=boolean	Disable requirement for Ctrl+Alt+Delete event to start logon to a Windows server. Default= On. Must be Off for smart card logons.
DisableSound=boolean	Disables Windows alert sounds. Default=False.
DriveEnabledA..Z=boolean	True if drive is mapped. Default=False.
DrivePathA..Z=string	UNIX file path for client drive mapping. No default.
DriveReadAccessA..Z=[0|1|2]	0=full access, 1=no access, 2=ask user. Default=0.
DriveWriteAccessA..Z=[0|1|2]	0=full access, 1=no access, 2=ask user. Default=0.
DynamicCDM=[On|Off]	Enabled Dynamic Client Drive Mapping. Default=On.
DynamicCDMDirs=string	Comma-separated list of directories to monitor for newly-mounted file systems. No default.
EnableAudioLRVolume=boolean	Specifies that the client audio accepts the left and right volume control set by server. Default=True.
EnableAudioPlaybackRate=boolean	Specifies that the client audio accepts the playback rate control set by server. Default=True.
EnableAudioVolume=boolean	Specifies that the client audio accepts the volume control set by the server.

Item	Description
	Default=True.
EnableICC=boolean	Enables inter-client communication features used by seamless session sharing and Connection Center. Default=True.
EnableSessionSharingClient=boolean	Sends session sharing requests to other ICA sessions on the same X display. Default=False.
EnableSessionSharingHost=boolean	Accepts session sharing requests from other ICA sessions on the same X display. Default=False.
EnableSSOnThruICAFile=boolean	Allow ICA file to turn on single sign-on. Default=False.
ForceLVBMode=boolean	Ensures that the Local Video Buffer (LVB) is used. Default=False.
HDXWebCamDebug=boolean	Enables the gst_read debug option. Default=False.
HDXWebCamDelayTime=integer	The period of time, in milliseconds, to wait before opening a webcam during a session. Default=2000ms.
HDXWebCamDelayType=integer	Determines whether or not to delay the opening of a webcam during a session. 0=do not delay opening, 1=if last close was less than delay time, delay by time remaining, 2=always delay. Default=1.
HDXWebCamDevice=string	Location of the webcam device. Default= /dev/video0
HDXWebCamEnabled=boolean	Enables webcam support if AllowAudioInput is also true.

Item	Description
	Default=True.
HDXWebCamFramesPerSec=integer	Frame rate requested from a webcam. Default=15.
HDXWebCamGStDebug=string	Comma-separated list of GStreamer debug options. No default.
HDXWebCamHeight=integer	Height of image requested from a webcam. Default=288.
HDXWebCamQuality=integer	Theora quality requested from a webcam, within a range of 1-63. Default=16.
HDXWebCamWidth=integer	Width of image requested from a webcam. Default=352.
HoldComPortsOpen=boolean	Determines whether the client holds system serial ports open for session duration. Default=False.
Hotkey1..12Char=[F1...F12]	Function key to use for mapping keyboard shortcut sequence ALT+Fn.
HotKey1..12Shift=string	Shift state to get keyboard shortcut mapping for Alt+Fn; for example, ALT +CTRL.
HowManySkipRedrawPerChange =integer	The maximum number of successive palette changes that can follow one another closely without a redraw. Default=9.
HttpBrowserAddress=string	Server name or IP address used for HTTP browsing. Default=ica.

Item	Description
HttpBrowserAddress2..14=string	Server names or IP addresses for business failover. No default.
ICAKeepAliveEnabled=boolean	Monitors reception of data from the ICA host and assumes the connection has failed if a request packet fails to produce a response. Default=TransportReconnectEnabled setting.
ICAKeepAliveInterval=integer	The interval, in milliseconds, for checking on data received when ICAKeepAliveEnabled is set. Disconnection occurs if the connection is idle for the specified period and if no response is received during this time after a request. Default=10000.
IgnoreErrors=integer list	A comma-separated list of the error numbers to be ignored by the client. No default.
IgnoreFileChangeSize=boolean	Stops time-out copying large files to floppies. Default=False.
IgnoreShutdownErrors=boolean	Error messages are not shown during session shutdown when this is enabled. Default=True.
KeyboardDescription=string	Description of keyboard mapping. Default=Automatic (User Profile).
KeyboardLayout=string	Keyboard layout from module.ini. Default=none.
KeyboardMappingFile=string	Name of file in $ICAROOT/keyboard. Default=automatic.kbd.
KeyboardTimer=integer	Keyboard event flush interval. Default=0ms.

Item	Description
KeyboardType=string	Selects a keyboard type code to be sent to the server. The value should be one of the strings in the [KeyboardType] section of module.ini.
LastComPortNum=integer	Last COM port device number used. Default=0.
MapMouseButton2=boolean	Treats the middle mouse button the same as the right button. Default=False.
MouseDoubleClickHeight=integer	Mouse double-click height in pixels Default=4.
MouseDoubleClickTimer=integer	Mouse double-click time. Default=500ms.
MouseDoubleClickWidth=integer	Mouse double-click width in pixels. Default=4.
MouseScrollAmount=integer	Sets the amount moved for each scroll wheel click. Default=120.
MouseTimer=integer	Mouse event flush interval in milliseconds or zero. Default=0.
MouseWheelMapping=integer,integer	Mouse buttons whose down events are treated as a mouse wheel motion in the ICA protocol. Default=4,5.
MouseXButtonMapping =integer,integer	Specifies mouse buttons that should be mapped as additional buttons X1 and X2. Default=8,9.
MSLocaleNumber=hex number	The Microsoft locale identifier to send to the server. These numbers are identical to the low 16-bits of the corresponding keyboard layout numbers.

Item	Description
NDSTree=string	Space-separated list of NDS® trees specified by name or by IP address. Each entry may have :ppp added to indicate a port number. The tree specified by the server running the Web Interface will always be tried before the configuration file is checked. Default="".
PointerClickTime=integer	Specifies the length of time after a mouse click that the client allows attempts by the server to move the pointer, overriding the effect of PointerGrabTime. Default=1000 (milliseconds).
PointerGrabTime=integer	Specifies the length of time after mouse movement that the client ignores attempts by the server to move the pointer. This is for echo suppression. Default=750 (milliseconds).
ReaderStatusPollPeriod=integer	Smart card status polling period. Default=5000ms.
Realm_abc=ANY.COM	Causes Windows domain abc to be mapped to Kerberos realm ANY.COM when changing an expired password. The default action is to map to uppercase (ABC)
SetTWIFocus=boolean	Propagates local focus changes for seamless windows to the server. Default=False. Note that the default setting for versions earlier than 8.2 is True.
ShadowPointer=boolean	Passes mouse pointer positioning commands to the X server. Default=True.
SkipRedrawPerPaletteChange=boolean	Enables batching of redraw requests following palette changes. This reduces flickering when an application changes the palette rapidly. It is only relevant in

Item	Description
	256 color mode when shared colors or a TrueColor visual are used. It is ignored if Session-Depth Local Video Buffer (SDLVB), the default, is used. Default=Off.
SpeedScreenMMAAudioEnabled=boolean	Enables SpeedScreen Multimedia Acceleration support for compressed audio data. Default=True.
SpeedScreenMMAFlowControlV3=boolean	Enables Version 3 flow control for SpeedScreen Multimedia Acceleration when used with suitable servers. Default =True
SpeedscreenMMAForceAspectRatio= boolean	Sets the force_aspect_ratio property for the GStreamer image sink element. Default=False.
SpeedscreenMMAGSTCheck=boolean	When enabled, checks for GStreamer support. Default=False.
SpeedScreenMMASecondsToBuffer= integer	Number of seconds of multimedia data that the server expects to be buffered in the client. Default=10.
SpeedScreenMMAStopOverlayHandling Events=boolean	Stops GStreamer overlay from handling X events. This avoids a problem with mouse movements not ending Windows Media Player's full screen mode properly. Note, however, that this may cause problems with the size of the video window. Default=True
SpeedScreenMMAVerbose=boolean	Enables logging of format information for audio and video streams in the Citrix SpeedScreen Multimedia Acceleration channel. Default=False.

Item	Description
SpeedScreenMMAVideoEnabled=boolean	Enables SpeedScreen Multimedia Acceleration support for compressed video data. Default=True.
SSLEnable=boolean	Controls the use of SSL for TCP connections that do not specify their own value.
SSLInTitle=boolean	Controls whether or not the SSL strength indicator is shown in a session window's title bar. Default=On.
SSOnUserSetting=boolean	Allow appsrv.ini to turn on single sign-on. Default=False
StopOnUnmap=boolean	Commands the server to stop sending screen updates when the session window is iconified. Default=True.
SucConnTimeout=integer	Works with the ApplySucConnTimeoutToDesktops setting. Specifies the number of seconds to wait for a recently started session to become available for session sharing. When ApplySucConnTimeoutToDesktops is applied to desktops, repeated clicks launch multiple sessions, but you can set SucConnTimeout to a suitable timeout and run a custom script in between the desktop launches. Default=20. **Note:** To revert to the behavior in versions before Receiver for Linux 13.0, and allow a separate session launch for each click, set SucConnTimeout to 0.
SunRayClientName=string	Specifies the prefix part of a SunRay client name with URL escape characters. This allows trailing spaces, represented

Item	Description
	by %20. The remaining part of the client name is based on the ethernet address of the SunRay terminal. Default=SunRay-.
TcpBrowserAddress=string	Server name or IP address to use for browsing. No default. Use broadcast.
TcpBrowserAddress2..15=string	Controls the protocol used to locate the ICA host for the connection. This is a default value for connections that do not specify it individually.
TransportReconnectDelay=integer	Time in seconds to wait for the network to recover before automatic reconnection starts. Default=30.
TransportReconnectEnabled=boolean	Enables automatic reconnection of sessions when the network connection to the ICA host is lost. Default=True.
TransportReconnectOptions=integer	Specifies options for automatic reconnection. Add 1 to show a dialog during reconnection, and 2 to remove session windows when reconnection starts. Default=3.
TransportReconnectRetries=integer	Specifies how often to retry automatic reconnection. Default=3.
UnixPrintCommand=string	Command format used to print files. Default="lpr -P\"%s\"".
UpdateTime=integer	Time in milliseconds between batched Local Video Buffer (LVB) updates. Default=100. Note that this value is used only if your server does not control updates,

Item	Description
UseAlternateAddress=boolean	Uses alternate address for firewall connections. Default=False.
UseDynamicFileTypeAssociations=boolean	Uses dynamic file type association data from Citrix XenApp for file drag and drop operations. When this value is false, user interface options are enabled to allow users to specify static file type associations. Default=True.
UseIconWindow=boolean	Uses a window rather than a pixmap for the icons of session windows. This is required for strict CM compliance, but note that many window managers do not show icons correctly if this is set to True. Default=False
UseLocalIME=boolean	Uses the local X input method to interpret keyboard input. This is supported only for European languages. Default=True.
UsePrintcap=boolean	Allows Receiver for Linux to look for printers in /etc/printcap. Default=False.
UserVisualClass=string	Allow user-specified X visual class. Value is PseudoColor, TrueColor, or Grayscale. No default.
UserVisualID=hexadecimal integer	Uses this X visual, if possible, for session windows. No default.
Version=integer	Fixed value=2, overrides value in appsrv.ini, ignored.
WindowManagerHeightAllowance=integer	Estimated height in pixels of window manager top and bottom frames. Default= 60.

Item	Description
WindowManagerWidthAllowance=integer	Number of pixels to allow for Window Manager decoration. Default=20.
WpadHost=string	Specifies the URL to query for the automatic proxy detection configuration file. Default= http://wpad/wpad.dat.
XmlAddressResolutionType =[DNS-Port | IPv4-Port]	Controls the form used for the ICA host location. Using DNS-Port (the default) may help a connection to pass through an address-translating firewall.
XmsReserve=integer	Default=0. Ignored.
[Thinwire3.0]	Thinwire Virtual Driver configuration.
ApproximateColors=boolean	Default color approximation setting. Default=False.
BypassWindowManager=boolean	Creates all seamless windows with the override-redirect attribute, so that they are ignored by the local window manager. Default=False.
DesiredColor=[1 | 2 | 4 | 8 | 15]	Default number of colors to use, 1=16 colors, 2=256 colors, 4=32K colors, 8=16 million colors, 15=automatically select highest available color depth. Default=15.
DesiredHRES=integer	Default horizontal window dimension. Default=640.
DesiredVRES=integer	Default vertical window dimension. Default=480.
DisableXRender=boolean	Disables the use of the X11 Render extension required for color cursors. Default=False.

Item	Description
ForceEmbeddedColormapSwitch=boolean	Forces sessions that are embedded in a web page to use a private colormap. Default=False.
IgnoreXErrors=string	Comma separated list of entries such as m.n/ p meaning ignore error code p on X protocol request with major type m and minor type n. No default.
InstallColormap=boolean	Installs the colormap when an override-redirect seamless window gains focus. Default=True.
LargeCacheSizeInK=integer	Large cache size in KB. Default=2048.
LocalWMDecorations=boolean	Allows the X window manager to decorate seamless windows. Default=False.
MMExtension=string	X extension to use for checking multimod settings. No default.
PersistentCacheMinBitmap=integer	Minimum size of bitmap for caching. Default=8192 bytes.
PersistentCachePath=string	Location of persistent cache.
PersistentCachePercent=integer	Persistent cache size as percentage of disk size. Default=3.
PersistentCacheSize=integer	Persistent cache size in KB. Default=0.
RedrawTimer=integer	Time delay (milliseconds) before a new screen update is requested after copying multiple obscured screen regions. Default=1000.

Item	Description
ScreenPercent=integer	Percentage of screen to use. Default=-1. Only values 1-100 are used.
Tw2CachePower=integer	Sets the size of the Thinwire 2 bitmap cache. Attempting to set this lower than 19 (512KB cache) or higher than 25 is ineffective. The default value is calculated dynamically to be 1.25 times the screen image size at the preferred color depth.
TWIMoveResizeHideWindowType=integer	Controls the method used for hiding server-side windows when moving or resizing client-side seamless windows that are controlled by a window manager. 1 hides server-side windows by minimizing them. 2 hides server-side windows by moving them to the bottom right corner, outside the screen. Default=1. Other values are invalid.
TWISetFocusBeforeRestore=boolean	Sets the focus on server-side windows before restoring them. This is a workaround for an issue with virtual Java applications. Default=False.
TWIWSHideWindowType=integer	Controls the method used for hiding server-side windows when switching between client-side workspaces. 1 hides server-side windows by minimizing them. 2 hides server-side windows by moving them to the bottom right corner, outside the screen. Default=1. Other values are invalid.
TwTotalOssSizePowerOf2=integer	Sets the maximum size of off-screen drawing surfaces used by the X server. (See EnableOSS). Default=24, meaning 16Mb.
UserVisualClass=boolean	Allows default X visual to be used. Default=No. Client selects visual.

Item	Description
UserVisualID=integer	User-specified visual ID. Default=0.
XFree86ShapeFixLevel=hexadecimal integer	Highest version number of XFree86 X servers that require a workaround when using the SHAPE extension. Default=40200001 (Version 4.2.1).
[Xdpy - X server vendor identification string]	X server vendor multimod requirements. These parameters identify vendors' X server versions that produce both key up and key down events in response to each key up or key down for the Caps, Scroll, or Num Lock keys.
All=string	One or more of Caps, Scroll, or Num Lock; multimod refers to all versions of the server.
n-=string	One or more of Caps, Scroll, or Num Lock; multimod refers to version n and upwards of server.
-n=string	One or more of Caps, Scroll, or Num Lock; multimod refers to version up to n inclusive of server.
m-n=string	One or more of Caps, Scroll, or Num Lock; multimod refers to server versions between m and n inclusive.

module.ini

This file contains a comprehensive listing of parameters used to select and configure the communications stack modules. The section headings identify the target module by name. The stack element types are:

- Transport Drivers (TD) - manage the communications connection
- Protocol Drivers (PD) - manage intermediate data stream filters
- WinStation Drivers (WD) - manage the presentation data stream
- Virtual Drivers (VD) - manage ICA protocol extensions

These elements are all loaded depending on the user configuration and the required stack relationships. The transport driver is loaded first, then protocol drivers, the WinStation driver, and virtual drivers. Each of the supported types has a section that

describes the module name and default parameters. Most parameters in this file are defaults. They can be overridden by equivalent entries in appsrv.ini.

Parameter syntax

Boolean parameters use Yes, True, 1, or On to indicate TRUE. Any other values, including No, False, 0, or Off, are interpreted as FALSE.

For all parameters, spaces are significant and values are case-sensitive.

Those marked as ignored are not currently used by the client, but can be reserved for future use, are redundant, or are used by other clients; for example, Win32 or Macintosh. In the last case, the parameter is read by the client but the result discarded.

Default values are embedded into the client program itself. Fixed values are set by the unmodified .ini configuration files.

Note: The values in module.ini are not affected by those in All_Regions.ini in the same way that values from other configuration files are. This is because the same permissions are required to change both files.

In the following table, the parameters are listed alphabetically within each section of the file.

Item	Description
[WFClient]	This section is used by the engine. It contains default session-oriented parameters.
AllowWriteOpenToROF=boolean	Emulates Microsoft Windows behavior by allowing files on a read-only disk to be opened for writing. Default=True.
AttemptCrossPlatformSessionReuse =boolean	Allows a seamless published application launched from one system to run in an ICA session originally started from a different system. The two systems must use the same X display. Default=False.
ContentRedirectionScheme=scheme1, scheme2	Defines a list of schemes for server to client content redirection. Server-client content redirection allows administrators to specify that URLs in a published application are opened using a local application. Each scheme is defined by its own section.

Item	Description
DeferredUpdateMode=boolean	Enables an efficient algorithm for updating seamless windows. Default=False.
EnableSessionSharingClient=boolean	When launching a seamless published application, search for an existing ICA session that can run it. Default=False.
EnableSessionSharingHost=boolean	Allow independently launched seamless published applications to run in the same ICA session. Default=False.
HostLookupTimeout=integer	Time-out (in seconds) for calls to gethostbyname(). Used only on Solaris. Default=5.
IsDesktopAppliance=boolean	Enables special behavior for terminals configured for XenDesktop exclusively. Default=Off.
KeyPassthroughEscapeChar=string	Key for the keyboard command to disable the transparent keyboard mode. Default=F2.
KeyPassthroughEscapeShift=string	Keyboard shift for the keyboard command to disable the transparent keyboard mode. Default=Ctrl.
PrinterQueryRefreshTime=integer	Maximum time in seconds to cache list of available printer queues. Default=60.
ReplaceOverlineWithTilde=boolean	Treat overline key as tilde. Used only when Japanese keyboard layout is selected. Default=False.
ServerToClientPowerOf2=integer	Controls the buffer size for the compression method used in MetaFrame 1.0.

Item	Description
	Default=15.
TransparentKeyPassthrough=string	Enables keyboard shortcut sequences defined by the local Windows manager in the session. Keywords are: Local, Remote, FullScreenOnly. Default=FullScreenOnly.
UseSystemCharacterConversion=boolean	Use CHARICONV.DLL in preference to CHARCONV.DLL for character encoding conversions. Default=True.
Version=2	Fixed value; overrides value in appsrv.ini. Ignored.
[ICA 3.0]	Client module configuration.
AllowShared16Colors=boolean	Enable colormap entry sharing for 16-color.
BufferLength2=integer	High performance buffer length. Default=5000.
ClientAudio=boolean	Enable client audio mapping. Default=On
ClientComm=boolean	Enable serial port mapping. Default=On.
ClientDrive=boolean	Enable client drive mapping. Default=On.
ClientPrinterQueue=boolean	Enable printer queue mapping. Default=On.
Clipboard=boolean	Enable the clipboard. Default=On.
ICACTL=boolean	Enable the ICA control channel. Default=On.
MaxRequestSize2=integer	High performance buffer request size.

Item	Description
	Default=4116.
MaxWindowSize2=integer	High performance buffer window. Default=62500.
MultiMedia=boolean	Enable HDX Mediastream Multimedia Acceleration. Default=On
SmartCard=boolean	Enable smart card support. Default=On
ThinWire3.0=boolean	Enable Thinwire. Default=On.
TWI=boolean	Enable seamless VD. Default=On
UserExperience=boolean	Enable performance information. Default=On.
VirtualDriver=string list	Comma-separated list of VDs to load.
WindowSize2=integer	High performance window size. Default=4102 bytes.
ZL_FONT=boolean	Enable latency reduction font VD. Default=On
ZLC=boolean	Enable latency reduction VD. Default=On
[TransportDriver]	This section lists all of the sections in module.ini that define transport settings.
TCP/IP=	Fixed null value.
[TCP/IP]	Transport driver configuration.
BrowserRetry=integer	Number of attempts to locate data collector, which acts as master browser. Default=3.

Item	Description
BrowserTimeout=integer	Number of milliseconds to wait before retry. Default=1000.
Encrypt=boolean	Turn on basic encryption. Default=Off. Fixed value=On.
ICAPortNumber =integer	Server port to use for ICA connection. Default=1494 (from the Internet Assigned Numbers Authority (IANA)).
ProtocolSupport=string list	Protocol drivers to load, fixed value=Rframe, Encrypt.
OutBufCountClient=integer	Number of client output buffers to allocate. Default=6.
OutBufCountClient2=integer	High performance client buffer count. Default=42.
OutBufCountHost=integer	Number of server output buffers to allocate. Default=8.
OutBufCountHost2=integer	High performance server buffer count. Default=42.
OutBufLength=integer	Size of output buffer. Default=512. Fixed value=530 bytes.
OutBufLength2=integer	High performance buffer length. Default=530 bytes.
RFrame=boolean	Turn on reliable framing. Default=Off. Fixed value=On.
[XenDesktop]	Parameters specifically for connection to XenDesktop, particularly for full-screen sessions.

Item	Description
ResetProgram=string	Path to a program to reset the session and the remote host. The user can invoke this by typing Ctrl+Alt+Del. Not set by default.
[RFrame]	Reliable framing protocol driver configuration, no parameters.
[EncryptionLevelSession]	This section specifies the encryption protocol for each level of encryption. EncryptionLevelSession in appsrv.ini defines the level used by each connection. Each encryption protocol is defined by corresponding drivers specified in module.ini.
Basic=Encrypt	Fixed value.
RC5 (128 bit - Login Only)=EncRC5-0	Fixed value.
RC5 (40 bit)=EncRC5-40	Fixed value.
RC5 (56 bit)=EncRC5-56	Fixed value.
RC5 (128 bit)=EncRC5-128	Fixed value.
[Encrypt]	Encryption protocol driver configuration.
DriverName=PDCRYPT1.DLL	Fixed value.
[EncRC5-0]	Encryption protocol configuration.
DriverName=PDCRYPT2.DLL	Fixed value.
[EncRC5-40]	Encryption protocol configuration.
DriverName=PDCRYPT2.DLL	Fixed value.
[EncRC5-56]	Encryption protocol configuration.
DriverName=PDCRYPT2.DLL	Fixed value.
[EncRC5-128]	Encryption protocol configuration.
DriverName=PDCRYPT2.DLL	Fixed value.
[Reliable]	Reliable transport protocol driver. Ignored.

Item	Description
[Compress]	Compression protocol driver configuration. Ignored.
[Framing]	Framing protocol driver configuration. Ignored.
[Modem]	Async protocol driver configuration. Ignored.
[Thinwire3.0]	Thinwire virtual driver configuration, overridden by parameters in wfclient.ini.
[Clipboard]	Clipboard virtual driver configuration.
ClipboardAllowed=boolean	Enables the clipboard channel. Default=True.
[ClientDrive]	Client drive mapping virtual driver configuration.
AllowSymlinkTraversalOutsideMap =boolean	Allows the following of symlinks outside the mapped root of the client drive mapping host. Default=False.
CacheDisable=boolean	Disable cache. Default=False.
CacheTimeout=integer	Cache time-out (seconds). Default=600.
CacheTimeoutHigh=integer	Cache time-out for times greater than 18 hours. Default=0.
CacheTransferSize=integer	Amount of data to transfer per operation. Default=0 (ICA buffer size).
CacheWriteAllocateDisable=boolean	Disable cache for write operations. Default=False.
CDMReadOnly=boolean	Allow only read-only access to client filesystems.

Item	Description
	Default=False.
DesktopFolder=path	Sets the desktop directory for the Special Folder Redirection feature. No default.
DocumentsFolder=path	Sets the documents directory for the Special Folder Redirection feature. No default.
MaxRequestSize=integer	For flow management. Fixed value=1046 bytes.
MaxWindowSize=integer	Window size for flow management. Fixed value=6276 bytes.
SFRAllowed=boolean	Enables the Special Folder Redirection option. Default=False.
TranslateCDMFileNames=boolean	The file names passed through the CDM channel are translated into the character encoding of the receiving system, in both directions. Default=True.
[ClientPrinterQueue]	Client printer mapping virtual driver configuration.
MaxWindowSize=integer	Maximum window size for flow management. Fixed value=1024 bytes.
MFPrintCommand=string	Command to use for Universal Printer Driver (UPD) printing. Default=lpr -P for BSD systems and lp -d for SYSV systems.
UnicodeEnabled=boolean	Enable UNICODE printer names. Default=True.
UnixPrintCommand=string	Command to use for non-UPD printing.

Item	Description
	Default=lpr -l -P for BSD systems and lp -d for SYSV systems.
WindowSize=integer	Write window size for flow management. Fixed value=512 bytes.
WindowsPrinter=string	Default Windows queue name to use. No default. Ignored.
[ClientAudio]	Client audio mapping virtual driver configuration.
AckDelayThresh=integer	Max time (in milliseconds) between sending "resource free" message if any resources free. Default=350.
AudioBufferSizeMilliseconds=integer	Audio buffer size, in milliseconds. Default=200ms.
AudioDevice=string	Audio device name. Linux default=default, SPARC default=/dev/audio. No default for other platforms.
AudioLatencyControlEnabled=boolean	Enables latency control. Default=False.
AudioMaxLatency=integer	Sets the maximum latency (in ms) before trying to discard audio data. Default=300ms.
AudioLatencyCorrectionInterval=integer	Defines how often to correct the latency (in ms). Default=300ms.
AudioTempLatencyBoost=integer	Sets the higher latency band (in ms) above the lower PlaybackDelayThresh band. Default=300ms
AudioWakeOnInput=boolean	Uses the client's event loop to wake up immediately when audio data is available

Item	Description
	to be read, for example, when recording audio. Default=True.
AudioWakeOnOutput=boolean	Uses the client's event loop to wake up immediately when audio data is available to be written, for example, when playing audio. Default=True.
CommandAckThresh=integer	Number of free client command buffers causing a "resource free" message to be sent to the server. Default=10.
DataAckThresh=integer	Number of free client data buffers causing a "resource free" message to be sent to the server. Default=10.
DriverName=VDCAM.DLL	Fixed value.
MaxDataBufferSize=integer	Maximum size of each data buffer. Default=2048 bytes.
NumCommandBuffers=integer	Number of client buffers to use for audio commands. Default=64.
PlaybackDelayThresh=integer	Delay (in milliseconds) between being asked to start audio playback and actually starting audio playback in order to build up a backlog of sound. Default=150.
[AudioConverter]	Audio format converter configuration.
DriverName=ClientAudCvt	Fixed value.
[AudioConverterList]	Audio format converter configuration.
Converter0=ADPCMConverter	Fixed value.
NumConverters=1	Fixed value.

Item	Description
[ADPCMConverter]	Audio format converter configuration.
DriverName=ADPCM_Module	Fixed value.
NumDataBuffers=integer	Number of client audio data buffers. Default=32.
[ClientComm]	Client COM port mapping virtual driver configuration.
CommPollSize=string	Use asynchronous polling. Default=Off.
CommPollWaitInc=integer	See CommPollWaitIncTime. Default=1.
CommPollWaitIncTime=integer	Time (in milliseconds) polling will poll before slowing by the number of milliseconds defined in CommPollWaitInc. Default=20.
CommPollWaitMax=integer	Slowest COM port polling rate. Default=500ms.
CommPollWaitMin=integer	Time (in milliseconds) to delay after receiving data. Default=1.
CommWakeOnInput=boolean	Uses the client's event loop to wake up immediately when serial port data are available to be read. Used only when CommPollSize=True. Default=True.
WindowSize=integer	Window for flow management. Default=1024 bytes.
[TWI]	Seamless parameters.
DriverName=VDTWIN.DLL	Fixed value.
[ZLC]	Zero latency parameters.
DriverName=VDZLC.DLL	Fixed value.

Item	Description
[ZL_FONT]	Zero latency font parameter.
DriverName=VDFON30W.DLL	Fixed value.
[ICACTL]	ICA control channel parameters.
[KeyboardLayout]	List of possible keyboards supported.
keyboardname=locale	Keyboard name; for example, British, German, US, and NT locale identifier. Fixed value.
[KeyboardType]	List of supported keyboard types.
keyboardtype=identifier	One keyboard type entry for each supported keyboard type.
[SmartCard]	Smart card virtual driver configuration.
DriverName=VSCARD.DLL	Fixed value.
PCSCCodePage=integer	Code page that should be used for communication with smart cards and readers. Default=0. A value of zero means use the default code page for the language used by the client.
PCSCLibraryName=string	File name of PC/SC shared library for smart card access. Default=libpscsclite.so.
SmartCardAllowed=boolean	Allows access to smart card devices on the client machine. Default=True.
[Hotkey Shift States]	Fixed values for keyboard shortcut masking.
(none)=0	Fixed value.
Alt=2560	Fixed value.
Ctrl=1280	Fixed value.
Shift=3	Fixed value.
Alt+Ctrl=3840	Fixed value.

Item	Description
Alt+Shift=2563	Fixed value.
Ctrl+Shift=1283	Fixed value.
Alt+Ctrl+Shift=3843	Fixed value.
[Hotkey Keys]	Fixed scancode values for keyboard shortcut keys.
(none)=0	Fixed value.
F1...F12=112...123	Fixed values.
Minus=12	Fixed value.
Plus=13	Fixed value.
Tab=16	Fixed value.
[File Type Associations]	This section lists the names of applications together with the file name extensions of their data files. It is used to construct the File Associations properties menu on clients with CDE support, and when the client is configured to use static file type associations.
[Scheme]	Defines the type of scheme for this section, for example [Browser] or [Player]. For more information, see the ContentRedirectionScheme parameter in module.ini.
AcceptURLType=type1, type2	The types of URL accepted by a given scheme, for example http, https.
Command=string	The command that runs the executable used for server to client redirection. No default.
Path=string	Search path for the executable used for server to client redirection. No default.
PercentS=integer	Number of "%s" occurrences in the command used for server to client redirection.

Item	Description
RejectURLType=type1, type2	The types of URL rejected by a given scheme.
[SSPI]	Kerberos configuration.
KerberosSelection=string	Specifies the order of preference for Kerberos implementations. Default=Heimdal/MIT.
[HeimdalKerberos]	This section contains information about the Heimdal implementation of Kerberos.
LIBKCP=string	The library to use for changing expired passwords using Heimdal Kerberos. Default=libkcph.so.
[MITKerberos]	This section contains information about the MIT implementation of Kerberos.
LIBKCP=string	The library to use for changing expired passwords using MIT Kerberos. Default=libkcpm.so.

reg.ini

reg.ini contains Citrix XenApp configuration settings. It is written by pnabrowse so it does not exist immediately after a typical installation. reg.ini provides initial values to pnabrowse that you can override through command-line arguments.

Important: reg.ini works with pnabrowse only. It has no effect on the deployments involving storebrowse or selfservice.

You may prefer to have a password in reg.ini, because this file has restricted read permission, rather than have it appearing on the command line. To do this, change `lastSavePassword=REG_DWORD:0` to `lastSavePassword=REG_DWORD:1`, and append the password using basic encryption to `lastPassword=REG_SZ:` .

This allows pnabrowse to run without the -P option. To do this, you must also omit the -U and -D options. pnabrowse continues to be governed by config.xml and therefore may reset these entries if the Web Interface does not have the authentication method properties set to allow the user to save the password.

Other configuration files

The $ICAROOT/config/ directory also contains several other .ini files, including All_Regions.ini, canonicalization.ini, regions.ini, Trusted_Region.ini, Unknown_Region.ini, and Untrusted_Region.ini. These files offer administrators an alternative way to configure the client settings described in previous sections. The files also allow administrators to configure client selective trust, a security feature that restricts the characteristics of an ICA session depending on the server to which the client connects.

For more information, see the configuration files in the $ICAROOT/config/ directory.

Library files

You can disable specific functionality from Receiver for Linux by removing the appropriate shared library file (.dll or .so file) from a client installation. The following table describes these libraries.

File	Location	Description
ADPCM.DLL	/opt/Citrix/ICAClient	Provides support for low quality audio if Speex is not available.
CHARICONV.DLL	/opt/Citrix/ICAClient	Provides character conversion functionality using the facilities of the standard system libraries. An alternative version, CHARCONV.DLL, is available for embedded system environments that lack the necessary library support. Note that Citrix recommends you do not remove this library without replacing it with the alternative.
ctxusb	/opt/Citrix/ICAClient	Helper utility for Generic USB redirection.
ctxh264.so	/opt/Citrix/ICAClient	Decoder for H.264 images.
ctxh264_fb.so	/opt/Citrix/ICAClient	Fallback decoder for H.264 images.
ctxjpeg.so	/opt/Citrix/ICAClient	Decoder for JPEG images.

File	Location	Description
ctxjpeg_fb.so	/opt/Citrix/ICAClient	Fallback decoder for JPEG images when Version 6 of libjpeg is present. This decoder also supports libjpeg-turbo.
ctxjpeg_fb_8.so	/opt/Citrix/ICAClient	Fallback decoder for JPEG images when Version 8 of libjpeg is present.
ctxusbd	/opt/Citrix/ICAClient	Daemon process for Generic USB redirection.
ctx_usb_isactive	/opt/Citrix/ICAClient	Helper utility for Generic USB redirection.
FlashContainer.bin	/opt/Citrix/ICAClient	Provides support for Flash redirection.
gst_play	/opt/Citrix/ICAClient/util	A GStreamer utility required for HDX Windows Multimedia Redirection.
gst_read	/opt/Citrix/ICAClient/util	A GStreamer utility required for HDX Realtime Webcam Video Compression.
libAMSDK	/opt/Citrix/ICAClient/lib	SDK used for communications between Receiver and Authentication Manager. This is required for connections using storebrowse or selfservice, but not pnabrowse.
libavcodec.so	/opt/Citrix/ICAClient/lib	This library is part of the FFmpeg suite required for HDX 3D Pro support.
libavformat.so	/opt/Citrix/ICAClient/lib	This library is part of the FFmpeg suite required for HDX 3D Pro support.
libavutil.so	/opt/Citrix/ICAClient/lib	This library is part of the FFmpeg suite required for HDX 3D Pro support.

File	Location	Description
libcrypto.so	Usually located in /opt/Citrix/ICAClient/lib	Cryptographic functions used to authenticate to NTLM proxies. Can be downloaded from http://www.openssl.org/.
libctxssl.so	/opt/Citrix/ICAClient	Contains functionality for Citrix SSL Relay, which provides end-to-end Secure Sockets Layer/Transport Layer Security (SSL/TLS) encryption between specific servers and clients.
libgstflatstm.so	/opt/Citrix/ICAClient/util	The GStreamer plug-in required for SpeedScreen Multimedia Acceleration.
libkcph.so	/opt/Citrix/ICAClient/lib	Provides password change support using Heimdal Kerberos.
libkcpm.so	/opt/Citrix/ICAClient/lib	Provides password change support using MIT Kerberos.
libldapsdk.so	Usually located in /opt/Citrix/ICAClient/lib	This library is used only by NDS.dll. It is provided by the NLDAPsdk package, part of the eDirectory product from Novell®, and is available from http://www.novell.com/.
libswscale.so	/opt/Citrix/ICAClient/lib	This library is part of the FFmpeg suite required for HDX 3D Pro support.
new_store.sh	/opt/Citrix/ICAClient/util	A helper script used by npica.so to handle CR files.
npica.so	/opt/Citrix/ICAClient	Plug-in for web browsers that are compatible with Netscape® software.
PDCRYPT1.DLL	/opt/Citrix/ICAClient	Contains basic Citrix encryption functionality.

File	Location	Description
		Citrix recommends that you do not remove this library.
PDCRYPT2.DLL	/opt/Citrix/ICAClient	Contains functionality for Citrix Secure ICA, which encrypts information sent between servers and clients.
SPEEX.DLL	/opt/Citrix/ICAClient	Provides preferred support for low and medium quality audio.
UIDialogLib.so	/opt/Citrix/ICAClient/lib	Allows creation of customized dialogs for non-X Windows systems. See Dialog library on page 28.
VDFLASH2.DLL	/opt/Citrix/ICAClient	Provides support for Flash redirection.
VDGUSB.DLL	/opt/Citrix/ICAClient	Provides support for Generic USB redirection.
VDMM.DLL	/opt/Citrix/ICAClient	Contains functionality for SpeedScreen Multimedia Acceleration.
VDSCARD.DLL	/opt/Citrix/ICAClient	Contains functionality for smart card support. The support is based around the PC/SC standard, to which any deployment of Receiver for Linux involving smart cards must adhere.
VDMSSPI.DLL	/opt/Citrix/ICAClient	Contains functionality for authentication using Kerberos tickets. Used when running the MIT version of the Kerberos software.
VORBIS.DLL	/opt/Citrix/ICAClient	Provides preferred support for high quality audio.

Note: gst_play, gst_read, and libgstflatstm.so are provided as both 32-bit and 64-bit versions, and a symbolic link to the appropriate version is created at installation time.