HELIX SERVER ADMINISTRATION GUIDE

Helix™ Server Version 11.1

Revision Date: 30 November 2006

RealNetworks, Inc.
PO Box 91123
Seattle, WA 98111-9223
U.S.A.

http://www.real.com
http://www.realnetworks.com

Printed in the United States of America.

SUMMARY OF CONTENTS

CONTENTS

INTRODUCTION

Welcome to Helix™ Server Version 11.1, the most powerful server software available for streaming media files across an intranet or the Internet. This manual will help you use and optimize Helix Server for real-time delivery of media files.

What is Helix?

Helix™ from RealNetworks is a universal digital media delivery platform. With industry-leading performance, integrated content distribution, advertising, user authentication, Web services support, and native delivery of RealMedia, Windows Media, QuickTime, and MPEG-4, Helix from RealNetworks is a robust digital media foundation that meets the needs of enterprises and networking service providers.

Audience for this Guide

This guide is intended for technical system administrators who will manage Helix Server and its activities, but not necessarily create the content that's streamed. Content creation information is available in a companion book, the *RealNetworks Production Guide*. Information services professionals, server administrators, Web masters, and others who provide Web pages for the Internet and for intranets may also find this book useful.

How this Guide Is Organized

This administration guide contains the following chapters and appendixes.

Chapter 1: New Features

If you're familiar with previous versions of Helix Server, this chapter will give you a quick update on the new features found in Helix Server.

Chapter 2: Overview

This chapter presents the "big picture" of how Helix Server works with a Web server to stream media to client software such as RealPlayer.

Chapter 3: Installation and Quick Start

Find out how to install and start Helix Server, and how to use the Web-based administration tool, Helix Administrator.

Chapter 4: Server Setup

This chapter covers the basic Helix Server configuration options involving addresses and ports. Most options are configured at installation and may need no changing.

Chapter 5: Clip Delivery

This chapter describes on-demand streaming features, and explains how to construct links to your streaming media clips.

Chapter 6: Multiple Servers

This chapter covers several features that you can use on a large network, including redundant servers, content caching, and proxy servers.

Chapter 7: Unicasts

Read this chapter to learn how to broadcast live events in RealMedia, Windows Media, QuickTime, and other formats.

Chapter 8: Multicasts

This chapter discusses multicasting, which sends a single, live stream to multiple clients, rather than a separate stream to each client. Clients connect to this stream rather than to the Helix Server computer.

Chapter 9: Transmitters and Receivers

Splitting is a method of server-to-server communication. This communication can be between two or more servers, or between Helix Server and RealProducer. Splitting enables you to distribute broadcasts broadly.

Chapter 10: Simulated Live Broadcasts

This chapter explains how to deliver archived or other on-demand content as if it were a live broadcast.

Chapter 11: Firewalls

If you're streaming media to users on the Internet, you'll need to know how Helix Server and other RealNetworks products interact with firewalls. This

chapter provides detailed information on the ports Helix Server requires for various server feature configurations.

Chapter 12: Access Control

This chapter shows you how to limit access to Helix Server through the IP addresses of clients attempting to connect.

Chapter 13: Authentication

You can control and limit who can view your content. This chapter describes the different Helix Server authentication methods.

Chapter 14: ISP Hosting

As described in this chapter, Internet Service Providers (ISPs) can host streaming media on behalf of their customers.

Chapter 15: Basic Logging

The basic logging feature records access requests made to Helix Server, as well as any errors that have occurred.

Chapter 16: Advanced Logging

Helix Server includes advanced reporting functionality that can report on many aspects of Helix Server operation, from server health to client connections.

Chapter 17: Activity Monitors

To provide the highest possible quality of service, you'll want to keep track of how many people request media from your Helix Server. This chapter describes the features available in the monitoring utility.

Chapter 18: SNMP

This chapter explains how to configure the Simple Network Monitoring Protocol (SNMP) plug-in and master agent to monitor Helix Server activity using third-party monitoring software.

Appendix A: Configuration File

This appendix presents a discussion on basics, editing guidelines, and XML syntax used in the Helix Server configuration file.

Appendix B: Address Space Bit Masks

This appendix explains how to identify a range of IP addresses by assigning a bit mask to a 32-Bit IP address. A number of Helix Server features can use these bit masks.

Appendix C: Authentication Data Storage
Helix Server comes with different methods for tracking authentication information, as described in this appendix. You can use this data for billing purposes, or to track who's watching what.

Conventions Used in this Manual

This section explains some conventional terms and formats used throughout the book.

Terminology

- Because this guide is designed for the Helix Server administrators, the term *you* refers to the administrator. Persons who play clips served by Helix Server are referred to as *visitors*, *viewers*, or *users*.
- Media players such as RealPlayer or Windows Media Player are referred to as *media players* or, more generically, as *clients*. Where information applies specifically to the RealNetworks® RealPlayer, this is clearly stated.
- The terms *clips*, *content*, *media clips*, and *media files* are used interchangeably to indicate the material that Helix Server streams.
- Production tools used to create the media clips that Helix Server streams are referred to collectively as *encoders*.

Typographical Conventions

The following table explains the typographic conventions used in this manual.

Notational Conventions

Convention	Meaning
`syntax`	This font is used for syntax of configuration files, URLs, or command-line instructions.
variables	Italic text represents variables. Substitute values appropriate for your system.
emphasis	Bold text is used for emphasis.

(Table Page 1 of 2)

Notational Conventions (continued)

Convention	Meaning
. . .	Ellipses indicate nonessential information omitted from examples.
[ ]	Square brackets indicate optional material. If you choose to use the material within the brackets, don't type the brackets themselves. An exception to this is in the basic access log, where statistics generated by the `StatsMask` variable are enclosed in regular brackets.

(Table Page 2 of 2)

Sample Links

Links that point to Helix Server take the following form:

`helixserver.example.com`

where:

- `helixserver` is a placeholder for the name of the computer on which you are running your Helix Server. Substitute the name of your organization's Helix Server computer wherever you see this syntax.
- `example.com` is a placeholder for a domain name. Substitute the domain name of your organization's computers wherever you see this syntax.

Default Locations and Values

In all of the examples given in this book, it's assumed that you've installed Helix Server in the default location for your operating system and that you're using default values for all settings. Of course, you can customize Helix Server however you want to meet your specific needs. Default values are used here for clarity of illustration. On Windows-based platforms, the default installation directory is:

```
C:\Program Files\Real\Helix Server
```

Additional Resources

In addition to this manual, you may want the following resources, which are available at **http://service.real.com/help/library/index.html**.

- Helix Server Readme File

  This file contains supplemental information not covered in this guide. To view this guide, click **Readme** in Helix Administrator.

- *Helix Server Configuration and Registry Reference*

  This guide explains the configuration variables in the Helix Server configuration file. It also provides a reference for the registry variables that you can add to customized reports.

- *RealProducer User's Guide*

  When you encode RealMedia content for desktop audiences, refer to this manual for instructions on running RealProducer and selecting streaming options.

- *Helix Proxy Administration Guide*

  This manual describes the Helix Proxy configuration and operation.

- *Helix Proxy Configuration and Registry Reference*

  This guide explains the configuration variables in the Helix Proxy configuration file. It also provides a reference for the registry variables that you can add to customized reports.

- *Helix Server and Helix Proxy Troubleshooting Guide*

  Refer to this document if you encounter problems while running Helix Server.

- *RealNetworks Production Guide*

  This manual explains the basics of creating streaming clips. You'll learn how to calculate bandwidth needs, and how to put a multimedia presentation together.

PART

I

HELIX SERVER BASICS

Use this section to find out about new features in Helix Server. If you're new to this technology, this section gives you an overview of how Helix Server works.

CHAPTER

1

NEW FEATURES

This chapter describes new features in Helix Server, and covers upgrade issues from previous versions of Helix Server or RealSystem Server.

New Features in Helix Server Version 11.1

The following sections describe features added to Helix Server Version 11.1.

Windows Media Player 11 Support

As explained in the section "Windows Media Player 11 and Later" on page 23, Windows Media Player 11 no longer supports the MMS protocol. The ASXgen utility can automatically provide an alternate HTTP URL that this player can use to request content. ASX files must be manually updated with an HTTP URL, as shown in the section "Windows Media ASX File" on page 89.

StreamerCount Variable

The new `StreamerCount` variable determines how many streaming processes to create. It replaces the `ProcessorCount` variable, and typically does not need to be configured manually.

> **For More Information:** Refer to the chapter on basic streaming features in *Helix Server Configuration and Registry Reference*.

SNMP Support

Using Simple Network Monitoring Protocol (SNMP) version 1, 2c, or 3, you can monitor Helix Server Version 11.1 from any SNMP-compliant management system. The SNMP feature allows you to monitor Helix Server performance and update server configuration remotely. It includes a master agent that acts as an intermediary between Helix Server and the management

system. The agent can run as an independent application or a Windows Service.

> **For More Information:** Chapter 18 explains how to configure the SNMP feature. The section "Installing Helix Server" on page 44 explains how to install the master agent as a service on Windows.

Latency Reduction in Live Broadcasts

By default, Helix Server Version 11.1 supports end-to-end latency reduction in live broadcasts. This requires that RealProducer 11 generates the broadcast stream. For more information, see "End-to-End Latency Reduction" on page 192.

Reduced Media Start-Up Delay

Helix Server significantly reduces start-up delay for on-demand and live RealMedia streams delivered to RealPlayer 11. Start-up delay is the time that elapses between when the user clicks the link to a stream and when the media begins to play. This feature requires no user configuration.

> **Tip:** To further reduce start-up delay, you can lower the preroll encoded into on-demand clips. For more information, refer to *RealProducer User's Guide*.

Delayed Shutdown

The delayed shutdown feature allows the Helix Server administrator to initiate a graceful shutdown sequence, giving media players time to report playback statistics before the shutdown or restart. The section "Implementing Delayed Shutdown" on page 70 explains how to set up this feature.

Additional Fields for Statistics Type 4

RealPlayer 11 and later report seven additional statistics values when you record statistics type 4 in the access log. These statistics help determine quality of service, especially when you run a low-latency broadcast. Additionally, statistics type 4, rather than types 1 and 2, is the new default for the access log. For more information, see "Statistics Type 4" on page 345.

IPv6 Support

Helix Server supports Internet Protocol version 6 addresses (IPv6) for most features. This allows you to run Helix Server on a dual-stack IPv4/IPv6 machine and take advantage of IPv6 addressing where possible.

> **For More Information:** The section "IP Version 6" on page 26 provides an overview of Helix Server's implementation of IPv6.

Bandwidth Detection for RealPlayer 11

Helix Server Version 11.1 implements a new method of server-side bandwidth detection that works with RealPlayer 11 and later. This feature allows RealPlayer to receive the optimal stream on different networks without the user manually changing the bandwidth configuration.

> **For More Information:** See "Automatic Bandwidth Detection" on page 29 for details about this feature.

Windows Media Push Broadcasting

In addition to pull broadcasting, Helix Server Version 11.1 supports push broadcasting with Windows Media Encoder version 9 and later. To enable push broadcasting, Helix Server defines a port that Windows Media Encoder uses to deliver the live stream.

> **For More Information:** See "Setting up a Windows Media Push Broadcast" on page 145.

TCP Preference for Media Players

The `PreferClientTCP` variable allows you to use TCP instead of UDP in RTSP streaming sessions as long as the media player lists TCP as a transport option. This feature must be licensed to be used. For more information, refer to the chapter on basic streaming features in *Helix Server Configuration and Registry Reference*.

Server-Side Rate Control

The rate control feature, which works with multi-rate container files, allows Helix Server to stream the appropriate bit rate to a media player based on the player's capabilities. This feature utilizes RDF files to inform Helix Server of

media player capabilities, and provides functions for managing stream congestion on a nework.

> **For More Information:** See "Implementing Rate Control" on page 74.

Support for Differentiated Services

Helix Server can set the bits for precedence and quality of service in IPv4 packets for many streaming media protocols. This allows networks that support IPv4 differentiated services to forward media packets to different nodes on the network according to different criteria.

> **For More Information:** See "Configuring Differentiated Services" on page 79.

Configurable RTSP Timeout Value

You can now set the amount of time that can elapse before Helix Server closes an idle RTSP connection. Refer to "Controlling Connections" on page 73 for information about setting this timeout value.

Logging Enhancements

Helix Server uses a new, more efficient method for writing log files that is based on the customized logging feature (now called *advanced logging*) introduced with version 9. The process for setting up log files through Helix Administrator is similar to the process in version 9.

> **For More Information:** See also "Compatibility with Previous Versions" on page 14.

Access and Error Logs Based on Advanced Logging Templates

The basic access and error logs are now predefined templates of the advanced logging feature. The Helix Administrator page for defining these log files is similar to previous releases. Within the configuration file, however, the basic access and error logs are defined along with the advanced logging templates. The `LogPath` and `ErrorLogPath` variables that previously indicated the location of these basic log files are now obsolete.

> **Tip:** The basic access log file includes a new logging style that captures information about bit rate and media format changes

during a presentation. See "Logging Style 6" on page 328 and "Bit Rate Adaptations" on page 336.

New Client Stats Templates for Media Player Statistics

The advanced logging feature includes a new template type called `Client Stats` that records media player information. You can use this template to generate periodic reports, such a separate report about the status of each connected media player every minute. This template type can also generate a report whenever a media player disconnects.

For More Information: See "Template Types" on page 354. The section "Generating Client Statistics Reports" on page 365 demonstrates a `Client Stats` report.

Removed Features in Helix Server Version 11.1

The latest version of Helix Server does not include the following features, which were present in earlier versions of Helix Server.

Progressive Networks Audio (PNA)

Helix Server Version 11.1 drops support for the proprietary PNA protocol used in earlier RealNetworks client software versions. Previous versions of Helix Server supported PNA for compatibility with older RealNetworks clients (RealPlayer 5 and earlier).

Automatic Ad Insertion

The ad insertion feature, which automatically created SMIL-based ads that displayed in RealPlayer, has been discontinued.

Distributed Licensing

Helix Server no longer supports distributed licensing, a feature in which a pool of Helix Servers shared a licensed feature set.

Legacy Bandwidth Negotiation

Helix Server Version 11.1 does not support legacy bandwidth negotiation. Before the introduction of SureStream RealAudio and RealVideo, bandwidth negotiation was handled by creating one file for each available bandwidth, and

placing all of the files in a directory that ended with .rm. Files were named according to the compression algorithm used to encode them.

Support for MPEG-1 and Vivo Video Formats

Helix Server does not stream MPEG-1, MPEG-2, or the Vivo video formats. It continues support for MP3, as well as the MPEG-4 versions of the MPEG standard.

Support for Pre-G2 Live Encoders

Helix Server no longer accepts live RealMedia streams from encoders earlier than RealProducer G2. As described in "Encoding with an Older Version of RealProducer" on page 142, it can accept live streams from RealProducer G2 through RealSystem Producer 8. For information about encoding live broadcasts with RealProducer 9 and higher, refer to "Setting Up Account-Based Broadcasting" on page 140.

Upgrade Issues

The following sections explain issues with upgrading to Helix Server Version 11.1 from an earlier version of Helix Server.

Compatibility with Previous Versions

Aside from the issues noted in the following sections, there are no known compatibility issues between Helix Server Version 11.1 and Helix Server versions 10 and 9, or RealSystem Server version 8. You can use a version 8, 9, or 10 configuration file with Helix Server Version 11.1, though no new features will be enabled. This allows you to migrate to a new version by installing the new software, using your old configuration file, and activating new features on an as-needed basis.

> **Tip:** RealNetworks recommends using identical versions of server products for server-to-server features like splitting.

ProcessorCount Variable Obsolete

The new `StreamerCount` variable replaces `ProcessorCount`, which should be removed from the configuration file. Refer to the chapter on basic streaming features in *Helix Server Configuration and Registry Reference*.

Default Logging of Statistics Type 4

By default, Helix Server records client statistics type 4 to its access log. The previous default was statistics types 1 and 2. If you wish to continue logging statistics other than type 4 statistics, you must modify the logging feature after you install Helix Server Version 11.1.

Note, too, that previous versions of Helix Server recorded only statistics type 4 if you selected statistics types 1, 2, and 4. This was because statistics type 4 included the same information found in statistics types 1 and 2. When you choose statistics types 1, 2, and 4 with Helix Server Version 11.1, however, you will log all three statistics fields.

> **For More Information:** See "Setting Up Basic Access and Error Logs" on page 349.

Access Logging Templates

As described in Chapter 15, the logging feature includes templates that replicate the standard access and error logs used in previous versions of Helix Server. These templates are turned on by default, and use the default values for log files in prior releases. If your current access log does not use the default logging style 5 or client statistics 4, you need to modify the templates after you upgrade to set the values used in your current log files.

> **For More Information:** See "Setting Up Basic Access and Error Logs" on page 349.

Media Player Session Templates

The new `Client Stats` template type records information about media player connections. A `Session` template no longer logs media player information. If you used `Session` templates to create reports when media players connected and disconnected, those templates no longer function with Helix Server Version 11.1. You need to recreate your reports using `Client Stats` templates.

> **For More Information:** See "Template Types" on page 354. The section "Creating Logging Templates" on page 360 explains how to set up a `Client Stats` report.

Binding Syntax for IPv6 Machines

The older syntax to bind to all IP addresses on a machine, `0.0.0.0`, binds only the IPv4 addresses on a dual-stack IPv4/IPv6 machine. To bind to all IPv4 and IPv6 addresses on a dual-stack machine, specify `any` as the binding option.

> **For More Information:** See "Binding to an IP Address" on page 66 for an explanation of how to set bindings and a list of additional binding options.

Upgrades from RealServer 7 or Earlier

Helix Server is fully compatible with RealServer 3 through 7, with the exception of the splitting feature. Because of the many changes to the splitting feature from version 8, all nodes along the stream—transmitter, relay, and receiver—must run Helix Server. If you are moving from a RealServer 7 or earlier product, note that links used for splitting have completely changed. Keep in mind that a source is now called a *transmitter*, and a splitter is now called a *receiver*.

> **For More Information:** See "Linking to Split Content" on page 208.

Default Installation Directory

On Windows, Helix Server installs into the following default location, which differs from the installation paths for previous versions of RealSystem Server:

```
C:\Program Files\Real\Helix Server
```

If you choose the default location, you'll need to move your existing content to the new directory tree, as described in "Upgrading in a Different Directory" on page 46.

CHAPTER

2

OVERVIEW

Designed for powerful and flexible media delivery, Helix Server Version 11.1 streams the widest variety of media, such as audio, video, animation, images, and text, to the broadest range of media players, including RealPlayer, Windows Media Player, and Apple QuickTime Player. If you are new to Helix Server, this chapter introduces you to Helix Server concepts and features.

Media Formats

Helix Server can stream on-demand clips and broadcast live events in more media formats than any other media server. Depending on its license, Helix Server can serve the file formats listed below. Although not exhaustive, the following list represents the major media formats available with Helix Server, which can deliver additional formats through plug-ins created by third-party developers.

RealNetworks:	RealAudio (.rm, .ra), RealVideo (.rm, .rmvb), RealPix (.rp), RealText (.rt)
Macromedia:	Flash (.swf)
Microsoft:	Windows Media (.asf, .wma, .wmv)
Apple:	QuickTime (.mov)
Standards-Based:	MPEG-4, MP3
Image Formats:	GIF (.gif), JPEG (.jpg, jpeg), PNG (.png)
Other:	AU (.au), AIFF (.aif, .ief), WAV (.wav)

Helix Server can deliver the same media formats on any of its supported operating systems, which include Windows and many UNIX variants such as Linux. This allows you to stream the media formats you want, using the operating system of your choice. Helix Servers running on different operating

systems are completely interoperable, allowing you to deliver homogeneous media services in a heterogeneous network environment.

> **For More Information:** The versions of media formats and media players supported by Helix Server are subject to change. Check **http://www.realnetworks.com/resources** for the latest information.

Understanding Media Formats and Codecs

To understand the types of media that Helix Server can stream, you need to understand various components such as the *file format* , *hint track*, and *codec*.

File Formats

Media formats are commonly named according to file formats, such as RealMedia, Windows Media, QuickTime, and MPEG. The file format specifies how information is packaged within the clip. For example, each media clip contains numerous informational fields in addition to the audio or video data. These fields provide the media player with the clip title, playing time, and so on. The file format specifies what information the clip may contain, and where within the file the information fields and media data are stored.

To stream a media clip, Helix Server must be able to read the file format to pull out necessary information and media data packets. It generally has a separate file format plug-in for each file type that it uses to read the file. If it does not possess the appropriate plug-in for a certain file format, Helix Server cannot stream the file. Although a file format plug-in enables Helix Server to open and read a file of a certain type, it does not allow it to decode the audio and video data, which is the job of a codec on a media player.

Hint Track

Many streaming media formats, including QuickTime and MPEG, contain a hint track that provides Helix Server with information about how to stream the clip's media packets. Although you can typically encode these clips without a hint track, doing so is recommended only for downloaded media rather than streamed media. If the format can include a hint track, add the track when encoding the clip to ensure that the clip streams well.

> **Note:** As discussed in "Support for Non-Hinted MPEG-4 Tracks" on page 21, Helix Server can stream some variations of the MPEG-4 format without relying on a hint track.

Codecs

Every streaming media clip is produced using a codec, which is shorthand for *coder/decoder*. The encoding software used to create the streaming clip employs a codec to compress the media data. On the receiving end, the media player uses the same codec to decode the media data and play the clip. If the media player does not have the correct codec, it cannot play the clip.

To illustrate these points, consider QuickTime, which is a file format that uses the file extension `.mov`. The most popular codec for encoding QuickTime clips is the proprietary Sorenson codec. However, QuickTime clips can also be encoded using a number of different codecs. A QuickTime clip can be encoded with the standards-based MP3 codec, for example, the same codec used with `.mp3` files.

Because Helix Server can read the QuickTime file format, it can stream any QuickTime clip to any player. Whether the media player can play it, however, depends on whether the player supports both the file format and the codec. QuickTime Player both reads the QuickTime format and includes the Sorenson codec. Although RealPlayer reads the QuickTime format, it does not include the Sorenson codec, so it can play a QuickTime clip only if it uses a standards-based codec such as MP3.

Selecting Formats to Stream

When determining which clip formats to stream, consider first the media players you want to reach. As explained in the preceding sections, the player must support the file format and possess the codec needed to decode the media packets. In most cases you need to pay attention to codec versions as well. For example, RealVideo 9 is a different codec from RealVideo 8. RealPlayer can play all versions of RealVideo, but older RealPlayers cannot play RealVideo 9.

To determine if Helix Server can stream a clip, open the clip in RealPlayer from your desktop. If RealPlayer can play the clip, Helix Server typically can deliver it, as long as the clip is in a streaming format rather than a format designed for download. Note, however, that Helix Server can also stream clips that RealPlayer does not play.

RealNetworks Formats

Helix Server streams RealAudio, RealVideo, RealText, RealPix, and SMIL (1.0 and 2.0). It supports all past codecs and file formats, meaning that you can deliver any existing RealMedia clip with the current version of Helix Server.

SureStream RealAudio and RealVideo

Because different viewers have different streaming bandwidths available to them, RealAudio and RealVideo clips can use SureStream technology to encode multiple streams at different bandwidths in a single clip. When a RealPlayer requests a clip, Helix Server delivers the stream suited for that player's connection speed. This way, each viewer receives the highest-quality stream possible. In addition, Helix Server and the player can switch between streams to compensate for changing network conditions.

> **For More Information:** For instructions on encoding SureStream RealAudio and RealVideo clips, see *RealProducer User's Guide*. See also the audio and video chapters of *RealNetworks Production Guide*.

Secure RealMedia

Secure RealMedia is a proprietary file format that uses the file extension `.rms`. You can package as secure RealMedia an unsecure clip encoded by a variety of codecs, such as RealAudio, RealVideo, MP3, H.263, H.264, MPEG-4, AAC, or AMR. The secure clip includes a globally unique identifier (GUID), as well as a key for the secured file.

SMIL

For media streamed to RealPlayer, the content creator may use Synchronized Multimedia Integration Language (SMIL) to coordinate multiple clips into a single presentation. A SMIL file, which has the file extension `.smil`, uses XML-based markup to lay out and time any number of clips played together or in sequence. It can also open HTML pages in RealPlayer, and create special effects such as fades during clip transitions. RealPlayer G2 through RealPlayer 8 can play SMIL 1.0 files. RealPlayer supports SMIL 1.0 and 2.0.

> **For More Information:** See *RealNetworks Production Guide* for instructions about using SMIL.

RealPix

Using RealPix, you can stream slideshows composed of still images in GIF, JPEG, or PNG formats. The RealPix markup language includes special effects, letting you fade between images, for example, or zoom in on an image detail. You can even use SMIL to coordinate your slideshow with a streaming soundtrack.

> **For More Information:** *RealNetworks Production Guide* contains a chapter on producing RealPix slideshows.

RealText

With RealText, you can create timed text clips that can stream alone or in combination with other media such as audio or video. This makes RealText a handy means for adding text to SMIL presentations. Using RealText, you can add subtitles to a video, for example, or provide closed-captioning.

> **For More Information:** See the RealText chapter in *RealNetworks Production Guide*.

MPEG Audio and Video

Helix Server streams the standards-based MPEG formats, including MPEG-4 and MP3, which is the audio layer of the MPEG-1 format. Helix Server accepts requests for streaming MPEG content over the RTSP control protocol, and can deliver the stream to any client that supports the RTP packet format. RealPlayer plays most of the supported MPEG formats.

MPEG-4 Formats

Helix Server streams hinted MPEG-4 content over RTSP/RTP. It delivers ISMA-compliant bit streams, and you can view MPEG-4 bit streams using any ISMA-compliant client that supports RTSP/RTP, such as RealPlayer and Apple QuickTime Player. MPEG-4 clips commonly use the file extension `mp4`.

Support for Non-Hinted MPEG-4 Tracks

Helix Server generally can stream MPEG-4 clips encoded with any codec, as long as the clips include a hint track. In addition, Helix Server can stream clips encoded with certain codecs whether or not the clips contain hint tracks. The following table lists the MPEG-4 audio and video codecs that do not need to

include hint tracks. For these codecs, Helix Server refers to the hint track if it is present, but still streams the media packets if the track is absent.

MPEG-4 Codecs That Do Not Require Hint Tracks

Codec	Type	MIME Type	Reference
H.263, PO, P3	Video	video/h263-2000	http://www.ietf.org/rfc/rfc2429.txt
MPEG-4 (SPL0)	Video	video/mp4v-es	http://www.ietf.org/rfc/rfc3016.txt
AAC, LC, and LTP	Audio	audio/mp4a-latm	http://www.ietf.org/rfc/rfc3016.txt
AMR-NB	Audio	audio/amr	http://www.ietf.org/rfc/rfc3267.txt
AMR-WB	Audio	audio/amr-wb	http://www.ietf.org/rfc/rfc3267.txt

Note: Through the configuration file, you can turn off hint track reading entirely for these codecs. For more information, refer to the on-demand delivery chapter of *Helix Server Configuration and Registry Reference*.

RTP-Delivered Formats

Helix Server includes broad support for the standards-based RTP packet format, which is used by default with QuickTime and MPEG, and optionally with RealMedia. This allows Helix Server to deliver on-demand clips or live broadcasts for virtually any media type that streams over the RTSP control protocol and the RTP packet format.

For More Information: The section “Packet Formats” on page 251 explains RTP.

Windows Media

Helix Server can stream Microsoft Windows Media audio and video to Windows Media Player versions 6.4 and later. Helix Server supports the MMS protocol, and can receive live streams from the Windows Media Encoder for unicasting and multicasting. Windows Media formats do not stream to RealPlayer.

Note: Helix Server supports multiple bit rate (MBR) encoding for on-demand clips and live streams for Windows Media version 7 only. It does not support MBR for Windows Media version 9 and later.

For More Information: For information about producing clips or live broadcasts in the Windows Media format, visit **http://www.microsoft.com/windows/windowsmedia/default.mspx**.

Windows Media Player 11 and Later

Windows Media Player 11 does not request media using the MMS protocol. When it encounters an MMS URL for on-demand or live content on Helix Server, Windows Media Player rejects the MMS URL and attempts to contact Helix Server over HTTP. If the request is successful, Helix Server streams the clip as a Windows Media stream cloaked as HTTP.

The ASXgen client launch utility, described in the section "ASXgen for Windows Media Player" on page 91, automatically provides an alternate HTTP URL that directs Windows Media Player 11 to the appropriate HTTP port on Helix Server. If you write ASX files manually, however, you need to include an alternate HTTP URL, as described in the section "Windows Media ASX File" on page 89.

Note: The HTTP connection used to deliver cloaked Windows Media streams is **not** managed the same as HTTP requests from browsers. The content is delivered only to the media player, and is protected against browser caching and user download.

QuickTime

Helix Server can stream hinted QuickTime clips to Apple's QuickTime Player 4 and later. It can deliver QuickTime clips encoded in all major proprietary and standards-based codecs, including Sorenson, Cinepak, Qualcomm PureVoice, and Qdesign. Helix Server can stream live QuickTime broadcasts from Sorenson Broadcaster, as well as simulated broadcasts from Playlist Broadcaster and the Helix Server SLTA utility.

For More Information: For QuickTime information, visit Apple's Web site **http://www.apple.com/quicktime/**. For more about Sorenson tools, visit **http://www.sorenson.com**.

Note: If you encode a QuickTime clip with a standards-based codec, such as h.261, h.263, or MP3, you can also stream the clip to RealPlayer 8 and higher. RealNetworks media players do

not play QuickTime clips encoded with proprietary codecs such as Sorenson, however.

Macromedia Flash

RealPlayer supports Macromedia Flash version 4, 3, and 2. Flash is well-suited for linear presentations that have a continuous audio track and animated images synchronized along a timeline. Such presentations can include demonstrations, training courses, product overviews, and movie trailers.

> **For More Information:** Learn more about Flash from Adobe's Web site at **http://www.adobe.com/**. See the Flash animation chapter in *RealNetworks Production Guide* for information about streaming Flash to RealPlayer.

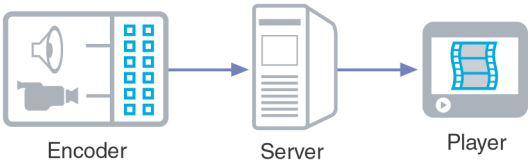
Encoding Tools

Helix Server delivers clips and live streams, but does not create them. The following sections explain the encoders and production tools that you can use with Helix Server.

Streaming Media Encoders

For delivering on-demand clips, the three major steps are encoding a clip with an encoding tool, streaming a clip through Helix Server, and playing a clip with a media player. Many encoders also accept live input, encoding it as a stream that is sent to Helix Server for live broadcast without being saved as a streaming clip first.

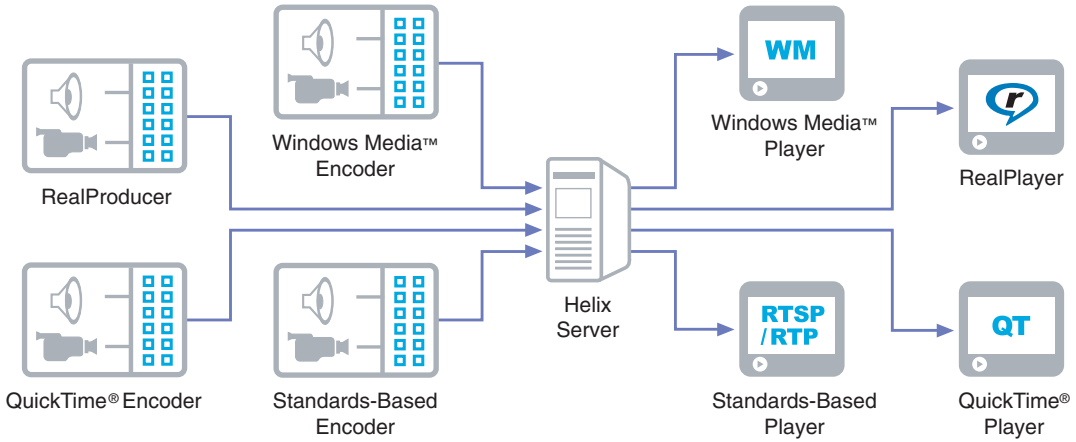
Encoding, Streaming, and Playing



For each media type, you use a specific tool (or family of tools) to encode audio and video as a streaming clip or live broadcast. RealProducer, for example, turns files in formats such as AVI and WAV into RealMedia clips. It can also encode live input from a camera or microphone. Tools like Microsoft Windows Media Encoder and Sorenson Broadcaster can encode audio and

video input for streaming to Windows Media Player and the QuickTime Player, respectively.

Universal Media Encoding and Delivery



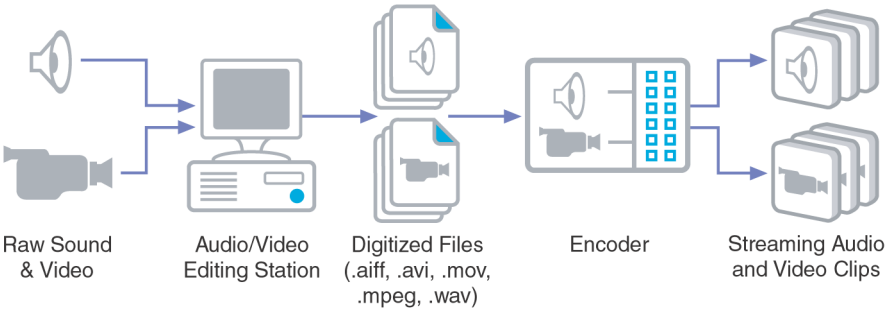
Tip: This manual uses the generic term *encoder* to refer to any software that creates live streams or prerecorded clips in a format that Helix Server can deliver.

For More Information: *RealProducer User's Guide* explains how to encode RealMedia clips.

Production Tools

Encoding a media clip or broadcast is the last step of a process that involves capturing, digitizing, editing, and optimizing audio or video data. A streaming media author uses various production tools to accomplish these jobs. These tools typically include video cameras, microphones, recording media such as tapes or CDs, mixing hardware, and audio and video editing software. You can use any tools you want to capture and edit audio and video input. You just need to ensure that your tools can save digitized files in formats that your encoding tools can accept.

Production Process for On-Demand Clips



For More Information: *RealNetworks Production Guide* contains a presentation planning chapter that explains these basic steps. See also that guide's audio and video chapters for tips on capturing and editing digital media.

Protocols

Helix Server streams media to clients over internal networks and the public Internet. Although Helix Server can deliver HTML pages, it's usually employed along with a separate Web server to coordinate the delivery of HTML pages and streaming media. The following sections summarize the supported communications protocols.

For More Information: For details about streaming protocols, see "Protocol Layers" on page 248.

IP Version 6

Helix Server supports both version 4 IP addresses (IPv4) and the newer version 6 IP addresses (IPv6). You can run Helix Server on a computer that uses one or more IPv4 addresses, as well as a machine that uses both IPv4 and IPv6 addressing. Note the following about using IPv6 addresses:

- Where possible, RealNetworks recommends that you specify DNS names instead of IP addresses. This allows Helix Server or another machine to resolve the domain name to an IPv4 or IPv6 address based on its network capabilities and the most efficient means of routing the request.
- Helix Server supports full IPv6 syntax, such as the following:
  - `1080:0:0:0:8:800:200C:417A`

It also supports the use of a double colon ("::") to compress fields containing only zeroes. You could therefore represent the preceding address as the following:

- `1080::8:800:200C:417A`

  Note: The double colon can appear at the beginning, middle, or end of the address. Only one double colon can appear within an address.

- Helix Server supports only the Classless Inter-Domain Routing (CIDR) format for IPv6 bitmasks, as shown in the following example:

  `2001:638:a01:2::/64`

  For More Information: Many Web resources explain IPv6 CIDR notation. See, for example, **http://en.wikipedia.org/wiki/Classless_inter-domain_routing**.

- Helix Server can stream on-demand content or live broadcasts using any streaming protocol to any media player that supports IPv6. For media URLs, however, RealNetworks strongly recommends using domain names to allow streaming to clients that handle only IPv4.
- The section "Binding to an IP Address" on page 66 explains how to bind Helix Server to various IP addresses on a multi-stack machine.
- The access control feature allows you to define access rules for both IPv4- and IPv6-based client connections. The section "IPv4 and IPv6 Access Rules" on page 265 explains access rule checking for the two protocols.
- Helix Server does not support IPv6 addresses for the following features:
  - Multicasts, which Chapter 8 describes.
  - SNMP, which is covered in Chapter 8.
  - Differentiated Services, described in the section "Configuring Differentiated Services" on page 79.

Real Time Streaming Protocol (RTSP)

RTSP is a standards-based streaming media protocol endorsed by the Internet Engineering Task Force (**http://www.ietf.org/**). It enables Helix Server to communicate with all versions of RealPlayer starting with RealPlayer G2, as well as the QuickTime Player and any RTSP-based MPEG player. Helix Server

also supports the RTP packet protocol, the standards-based companion to the RTSP control protocol.

Microsoft Media Services (MMS)

MMS is a proprietary control protocol used by Helix Server to communicate with Windows Media Player.

HyperText Transfer Protocol (HTTP)

Although HTTP is not a streaming media protocol, Helix Server uses HTTP in a number of ways. For example, it uses HTTP to deliver the Helix Administrator HTML pages that allow you to configure and run Helix Server.

On-Demand Streaming

On-demand streaming is the most common method of delivering media. Comparable to rented DVDs, on-demand clips are available at any time. The media is digitized, encoded in a streaming format, and stored on Helix Server. Each viewer who requests an on-demand clip receives a separate data stream from Helix Server. The clip starts at its normal, encoded beginning, and each viewer can fast-forward, rewind, or pause the presentation independently.

Basic Streaming Features

After installation, Helix Server requires no configuration to stream clips on-demand. You can simply place your clips in the main content directory, and link to them from a Web page. Chapter 5 explains how to write links to on-demand clips. It also explains some basic streaming features that you can use:

- "Adding a Mount Point for On-Demand Clips" on page 95 explains how to store clips anywhere on a network, defining mount points that indicate the clip location.
- You can create aliases to shorten long URLs or hide clip locations as described in "Adding a Mount Point for On-Demand Clips" on page 95.
- See "Browsing On-Demand Content" on page 100 to learn how to list all on-demand clips residing on Helix Server.

- As described in "Displaying Source Information" on page 101, you can allow RealPlayer viewers to download a clip's encoding information, as well as see the markup for SMIL files.

Automatic Bandwidth Detection

Helix Server uses a method of bandwidth detection that allows RealPlayer to receive the optimal stream when connected to different networks. This eliminates the need for the user to change the bandwidth configuration manually. For example, a RealPlayer on a laptop computer may automatically receive a 150 Kbps LAN stream when the computer is on an office network, a 512 Kbps stream when plugged into the user's home DSL line, and a 34 Kbps modem stream when connecting through a dial-up modem while the user travels.

Note the following about automatic bandwidth detection:

- This feature works only with RealPlayer 11 and later. Earlier versions of RealPlayer prompt the user to set the optimal bandwidth choice when they detect a change in the network configuration. Other media players are not supported.
- Automatic bandwidth detection works only when RealPlayer uses the default RTSP control protocol and RDT packet format, or a version of HTTP cloaking. It does not work with RTSP using the standards-based RTP packet format.

  For More Information: See "Packet Formats" on page 251 and "HyperText Transfer Protocol (HTTP)" on page 250.

- Automatic bandwidth detection is compatible with all methods of launching RealPlayer described in the section "Launching Media Players and Opening URLs" on page 88. This includes the Ramgen utility, Ram files, and SDP files.
- If RealPlayer reconnects to a different Helix Server as described in "Implementing Redundant Servers" on page 107, the second Helix Server performs a test to determine the optimal streaming bandwidth.
- Automatic bandwidth detection is turned on by default. You can turn it off through the `SendBWPackets` variable in the Helix Server configuration file.

For More Information: Refer to *Helix Server Configuration and Registry Reference*.

Network Bandwidth Control

Several Helix Server features can assist you in managing bandwidth when streaming on an intranet or the Internet:

- The section "Controlling Connections" on page 73 explains how to place limits on streaming, such as a cap on the total number of connected media players or the total amount of outgoing bandwidth.
- The server-side rate control feature allows Helix Server to stream the appropriate bit rate to a media player based on the player's capabilities. It also provides functions for managing stream congestion on a nework. For more information, see "Implementing Rate Control" on page 74.
- As described in "Configuring Differentiated Services" on page 79, Helix Server can set the bits for precedence and quality of service in IPv4 packets for several streaming media protocols. This allows networks that support IP differentiated services to forward media packets to different nodes on the network according to different criteria.

Features for Multiple Servers

Several features help you to administer a large network that contains multiple Helix Servers delivering on-demand content. Chapter 6 discusses these features, which include the following:

- Instead of having RealPlayer reconnect to the same Helix Server if clip delivery is interrupted, you can point RealPlayer to a backup server, as explained in "Implementing Redundant Servers" on page 107.
- When you have a large network, you may want to replicate media assets across different Helix Servers. The features described in "Content Caching" on page 110 let you do this.
- In addition to multiple Helix Servers, your network can contain multiple Helix Proxies. The section "Using a Media Proxy" on page 119 explains how you can control the content that a Helix Proxy caches or splits.

Authentication and Access Control

Keeping your media assets secure is an important function of Helix Server. To allow you to verify and control connection attempts, Helix Server offers the following features:

- The features described in "Controlling Connections" on page 73 allow you to limit the number of simultaneous connections, or the amount of outgoing bandwidth.
- Chapter 12 explains how to associate connection rules based on IP addresses. These rules can allow or deny connections to specific protocol ports.
- Authentication, which Chapter 13 covers, verifies the identity of a user requesting streaming media. This verification can take the form of asking for a user name and password. Or, it can be entirely hidden from the viewer.
- For Internet Service Providers, Helix Server works with existing user accounts and directory structures to make media files available for streaming. For each ISP account, you can allocate a minimum and maximum number of connections. Chapter 14 explains this feature.

Monitoring and Reporting

Helix Server offers several features that allow you to monitor current connections, and view past information stored in a log file:

- Logging, as described in Chapter 16 and Chapter 15, allows you to create reports that enable you to see trends, gather information, or check on the status of your Helix Server.
- As described in Chapter 17, Helix Administrator includes a real-time tool that displays activity on your Helix Server. The monitor displays information such as who's using your Helix Server, the peak use times, and which files are requested the most.

SNMP

Using Simple Network Monitoring Protocol (SNMP) version 1, 2c, or 3, you can monitor Helix Server from any SNMP-compliant management system. The SNMP feature allows you to monitor Helix Server performance and update server configuration remotely. It includes a master agent that acts as

an intermediary between Helix Server and the management system. The agent can run as an independent application or a Windows service.

> **For More Information:** Chapter 18 explains how to configure the SNMP feature.

Feature Availability

Depending on which Helix Server product you use, some of the features described in this manual may not be available to you, or may be limited in some way. Consult your license file for a list of which features are enabled in your version of Helix Server. If you'd like to augment your Helix Server's capabilities, contact RealNetworks or your reseller.

> **For More Information:** For instructions on reading license files with Helix Administrator, see "License File Information" on page 56.

Live Broadcasting

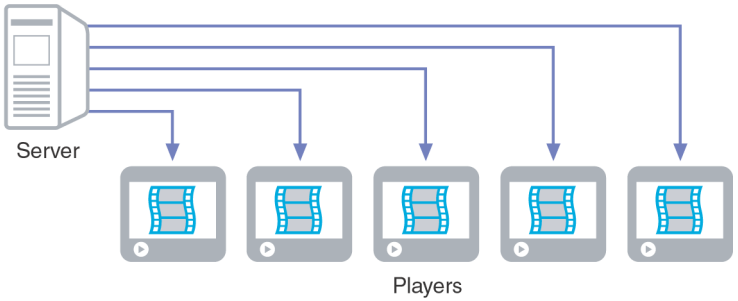
As with a live television event, a user can tune into a live Internet or intranet broadcast to join the presentation in progress. Because the event streams in real-time, the viewer cannot fast-forward or rewind through the broadcast. As the following sections explain, you can deliver live streams in several different ways, based on your needs and network capabilities.

> **Tip:** Most of the features described previously for on-demand delivery, including URL aliases, authentication, access control, and monitoring, work for live broadcasts as well.

Unicasting

Unicasting, which Chapter 7 covers, is the simplest method of live broadcasting. It works with most supported media formats and requires little setup. In unicasting, a media encoder delivers a live stream to Helix Server, which then delivers a separate broadcast stream to each media player. Because each player receives its own stream, unicasting is limited by your number of licensed client connections, as well as your outgoing bandwidth.

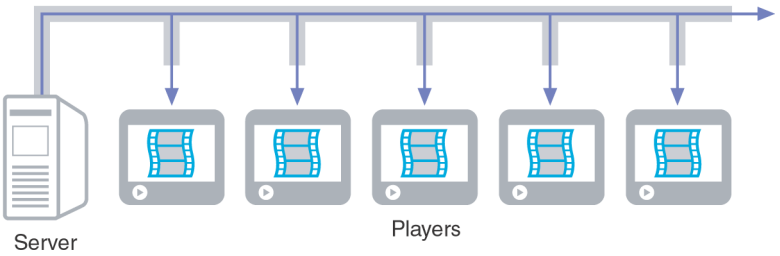
Unicasting



Multicasting

As explained in Chapter 8, multicasting dramatically reduces the bandwidth required for broadcasting, allowing many more viewers to participate. In a multicast, media players do not receive separate broadcast streams from Helix Server. Instead, they all connect to the same stream (or streams). Multicasting requires a multicast-enabled network, however, and is primarily suited for intranets, although multicasting on the Internet is possible.

Multicasting



Helix Server supports three kinds of multicasts:

- Back-channel multicasts maintain a low-bandwidth control channel between Helix Server and RealPlayer, enabling Helix Server to monitor how many players are connected. You can multicast any media format played by RealPlayer. In fact, you can make all unicasts simultaneously available as back-channel multicasts through the same broadcast hyperlink.
- Scalable multicasts do not maintain a control channel. Helix Server does not monitor the number of connected players, but it can broadcast a single stream to any number of players. It can also direct players to send

quality of service statistics back to it, or to a Web server, at the end of the broadcast. Helix Server supports RTP-based media players that comply with scalable multicasting standards, including RealPlayer and QuickTime Player.

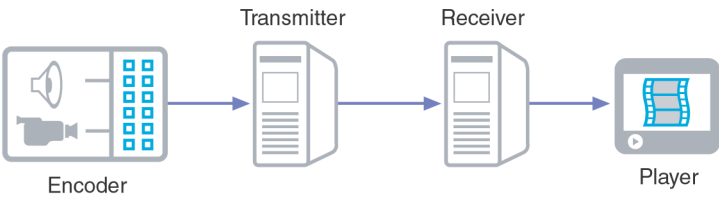
- Windows Media multicasts are scalable multicasts for Windows Media Player only. You can broadcast to any number of players, and direct client statistics to a Web server. Unlike with back-channel multicasts, you cannot make all Windows Media unicasts simultaneously available as multicasts.

Splitting

Whereas unicasting and multicasting deliver broadcast streams to media players, splitting transmits a broadcast stream in any media format from one Helix Server to another. As explained in Chapter 9, a Helix Server acting as a *transmitter* delivers a live media stream to other Helix Servers acting as *receivers*. Each receiver then broadcasts the stream to media players, either through unicasting or multicasting.

Splitting is a powerful feature that provides many ways to deliver a broadcast to any number of Helix Servers connected on an intranet or through the Internet. By increasing the pool of Helix Servers broadcasting an event, you can unicast to many thousands of viewers on the Internet, or multicast behind the firewalls of different facilities your organization maintains around the world.

Splitting



Low-Latency Broadcasts

When you stream a live broadcast in the RealMedia format, you can use a low-latency mode that automatically decreases the amount of data buffering on RealProducer, Helix Server, and RealPlayer. This reduces the amount of broadcast latency, meaning that viewers see events on RealPlayer only a few seconds after they are captured by a camera.

For More Information: See *RealProducer User's Guide* for instructions on encoding a low-latency stream. For information about using Helix Server in a low-latency broadcast, refer to "End-to-End Latency Reduction" on page 192.

Archiving

Live events don't exist as files, but you can use the archiving feature to write live content to a file as you broadcast. You can then stream the archive on-demand if you wish. Archiving a broadcast on Helix Server works only with RealMedia and MP3 broadcasts. For more information, see "Archiving Broadcasts" on page 131.

Redundant Encoders

For important live broadcasts, Helix Server can use multiple encoding sources. Should one source become unavailable, Helix Server switches automatically to the next source. Encoder redundancy works with any media format, is easy to set up, and has a few global parameters that the section "Using Broadcast Redundancy" on page 135 explains. This option is available for unicasts and back-channel multicasts.

Simulated Live Broadcasting

A broadcast event does not have to be a live stream delivered by a media encoder. Using the SLTA utility, which Chapter 10 covers, you can simulate a live broadcast using any number of prerecorded audio or video clips. This is a handy way to reach viewers in different time zones, for instance, or to stream encore presentations of live events. You might also broadcast clips that were never broadcast live. You can set up a playlist of songs, for example, and use SLTA to stream an Internet radio program. To the viewer, a simulated live broadcast appears to be a live event.

Feature Comparison

The Helix Server features described in the preceding sections work for all audio and video content, including RealMedia and MPEG, streamed to RealPlayer. Not all features are available when you stream other media formats to other media players, however. The next table summarizes the Helix Server

features available when you stream the following formats to the following media players:

- Windows Media (.asf, .wma, .wmv) to Windows Media Player
- QuickTime (.mov) or MPEG-4 (.mp4) to Apple QuickTime Player
- standards-based RTP-based formats, such as MPEG-4 (.mp4), to an RTP-based player

Helix Server Feature Comparison for Various Media Players

Helix Server Feature	Windows Media	Apple QuickTime	RTP-Based	Reference
HTTP cloaking for firewalls	**yes**	no	no	page 64
Launch utility for Web page links	**yes**	no	**yes**	page 90
Aliases in URLs	**yes**	**yes**	**yes**	page 97
Viewable clip source information	no	no	no	page 101
Reconnection to a redundant server	no	no	no	page 107
Cached content	**yes**	**yes**	**yes**	page 110
Proxied on-demand and live streams	**yes**	**yes**	**yes**	page 119
Unicasting	**yes**	**yes**	**yes**	page 129
Redundant live stream encoders	**yes**	**yes**	**yes**	page 135
Back-channel multicasting	no	no	no	page 161
Scalable multicasting	**yes**	**yes**	**yes**	page 164 page 171
Splitting from transmitter to receiver	**yes**	**yes**	**yes**	page 177
Simulated live broadcasts	**yes**	**yes**	**yes**	page 215
Access control by IP address	**yes**	**yes**	**yes**	page 263
User name and password validation	no	**yes**	no	page 269
Media player ID validation	no	no	no	page 297
Logging reports	**yes**	**yes**	**yes**	page 353
Online activity monitoring	**yes**	**yes**	**yes**	page 369

Helix Server Components

Helix Server runs on several server operating systems, including Windows and many UNIX variants. Although a few of its features are specific to Windows or

UNIX, Helix Servers on all operating systems are virtually identical in their setup and operation. The following sections describe the major components of Helix Server.

Plug-ins

Plug-ins provide the functionality of Helix Server's individual features, and reside in a `Plugins` subdirectory of your Helix Server installation directory. Helix Server uses one plug-in to stream RealVideo, for instance, and another to stream MPEG. Plug-ins also provide server features such as user authentication. Using Helix Server's open architecture, third parties can create additional plug-ins, enabling you to extend Helix Server's capabilities.

Helix Administrator

Helix Administrator is a secure, HTML-based interface that lets you run Helix Server through a frames-capable and Java-enabled browser located anywhere on your network. Helix Administrator divides Helix Server features into functional areas, and supplies easy-to-use forms and screens that allow you to configure Helix Server features, as well as monitor activity. See "Using Helix Administrator" on page 51 for details.

Configuration File

A human-readable, XML-based text file named `rmserver.cfg` stores Helix Server's configuration information. When you change Helix Server's configuration through Helix Administrator, this file is updated automatically. You can also edit the file manually, and maintain different files for different configurations, selecting the appropriate file when you start Helix Server. Because all configuration information is stored in this single file, you can create a master file that you propagate across multiple Helix Servers to set up an entire network quickly. Appendix A explains the configuration file syntax.

License File

You need one or more license files, which Helix Server reads at start-up, to enable your Helix Server features. RealNetworks distributes license files by e-mail.

Working with Other Professionals

Delivering streaming media clips or broadcasts typically requires the effort of several people. As the Helix Server administrator, you'll need to interact with content creators, as well as other network administrators. This guide points out cases where you need to provide Helix Server information to other professionals. The following sections summarize some important aspects of working with others to deliver streaming media presentations.

Encoding Professionals

The people who encode media, whether live broadcasts or on-demand content, need to know the address of Helix Server, protocols, port numbers, and mount points to use in links, especially if you've changed the settings from the defaults. *RealNetworks Production Guide*, which is written for content creators, explains the formats for linking to on-demand clips. However, you'll need to give content creators the URLs to broadcasts, which vary depending on the broadcast features you use.

Helix Server Administrators

If you have multiple servers in different facilities, you'll need to communicate certain information with other Helix Server administrators. When you use splitting, for instance, a Helix Server transmitter in one facility needs address and other information about a Helix Server receiver in another facility, and vice versa.

Firewall Administrators

If there are users within your network who either cannot receive presentations from Helix Server on the Internet, or who receive poor-quality streams, the information in Chapter 11 will help the firewall administrator understand what changes can be made to enhance the viewing experience.

PART

II

GETTING STARTED

In this section, you learn how to install and start Helix Server, use the Helix Administrator interface, and set the basic Helix Server configuration.

CHAPTER

3

INSTALLATION AND QUICK START

This chapter explains how to install Helix Server on Windows and UNIX platforms. It also introduces you to Helix Administrator, the Web-based tool for configuring Helix Server. As soon as you start Helix Server, it is ready to stream media, and the last section walks you through the processes for streaming clips and broadcasting live input.

Understanding Installation Issues

Before you install Helix Server, you need to make basic set-up and deployment decisions, as described in the following sections.

Firewalls and Helix Server

You need to choose where to place Helix Server in relation to firewalls—either your firewall or an outside organization's firewall—for optimal communication. Chapter 11 explains general issues involving firewalls. If your organization has a firewall, and you are not sure of its impact on Helix Server communication, be sure to read "Placing Helix Server in a Network" on page 252.

> **Tip:** If you have questions about which ports are available on your network to allow traffic through a firewall, consult with your firewall administrator.

Server and Proxy on the Same Machine

If you are installing Helix Gateway, which includes both Helix Server and Helix Proxy, RealNetworks recommends that you do **not** install both of these applications on the same machine. This prevents the applications from competing for processor power and memory. It is possible to install both applications on the same machine, however, if only one machine is available

and you have low-volume streaming needs. In this configuration, however, the installation procedure for Helix Server differs slightly from the procedure described in this chapter.

> **For More Information:** For instructions on installing Helix Server along with Helix Proxy on a single machine, refer to *Helix Server and Helix Proxy Troubleshooting Guide*.

Web Servers and Helix Server

RealNetworks suggests that you do not install Helix Server on the same physical machine that runs your Web server. This eliminates conflicts over ports, and helps to balance loads so that Helix Server is not affected by heavy Web server use, and vice versa. If you need to install Helix Server on the same computer as your Web server, observe the following precautions.

HTTP Port Resolution

Although HTTP is not a streaming protocol, Helix Server supports HTTP, primarily to handle media requests made by Web browsers, as well as to operate with the HTML-based Helix Administrator. Web browsers and media players typically make HTTP requests on port 80, and if a Web server and Helix Server reside on the same computer, they cannot both use port 80. There are two ways to avoid this port conflict.

Use a Nonstandard HTTP Port for Helix Server

During installation, you can specify a different HTTP port for Helix Server. In this case, though, all HTTP URLs to Helix Server must specify the port number so that clients make the request on the correct port. This creates potential for errors in writing URLs, and may limit client access if firewalls restrict HTTP requests to port 80.

> **For More Information:** See "Handling Communication through Nonstandard Ports" on page 65 and "Streaming to Client Software Behind Firewalls" on page 256.

Bind Helix Server to a Different IP Address

The second, better approach is to use two IP addresses for the same computer, one for Helix Server, the other for the Web server. This requires a *multi-homed* machine that has two or more network interfaces. In this configuration, you assign an IP address to each network interface, then bind Helix Server to one

of the IP addresses. In this way, Helix Server and your Web server both use port 80 on different network addresses.

Note, though, that Helix Server may fail to start after installation because of an HTTP port conflict if all of the following conditions are true:

- You install Helix Server on a multi-homed machine that also runs your Web server.
- Your Web server uses the IP address assigned to network interface 0.
- Your Web server uses port 80 for HTTP communications.
- You choose port 80 for Helix Server HTTP communications.

The conflict arises because Helix Server binds to network interface 0 after installation. If the Web server is using this address, the two servers will both try to claim port 80. (No problem arises, though, if the Web server uses a network interface other than 0.) The following procedure explains how to work around this problem.

➤ To prevent an HTTP port 80 conflict on a multi-homed machine:

1. When installing Helix Server, choose any unused port other than 80 for HTTP.
2. After installation, start Helix Server and bind it to a different IP address on your multi-homed machine, as described in "Binding to an IP Address" on page 66.
3. Change Helix Server's HTTP port to 80 as described in "Defining Communications Ports" on page 63.
4. Restart Helix Server.

Web Server MIME Types

Helix Server works with any Web server that supports configurable MIME types. The following table lists the recommended MIME types. Helix Server requires only that the Web server use the MIME types given below for the `.ram`

and .rpm extensions. See your Web server documentation for information about defining MIME types.

Web Server MIME Types and Extensions

MIME Types	Extensions
audio/x-pn-realaudio	ra, rm, ram
audio/x-pn-realaudio-plugin	rpm
application/x-pn-realmedia	rp
application/smil	smi, smil
application/sdp	sdp
image/gif	gif
image/jpg	jpg, jpeg
text/html	html, htm

For More Information: Helix Server can also deliver files over HTTP. For information about configuring its MIME types, see "Adding MIME Types for HTTP Communication" on page 70.

Installing Helix Server

To install Helix Server, you need a binary installation file and a license file that enables the Helix Server features. Although you can install Helix Server without the license file, Helix Server will not operate until you have obtained a valid license file. License files are delivered by e-mail after you download or purchase Helix Server.

Note: To install Helix Server as a Windows Service, you must have administrative access.

➤ To install Helix Server:

1. Launch the binary setup file you downloaded. If you have a Helix Server installation CD, open the folder named for the operating system you are using, and double-click the setup file.
2. Read the installation recommendations and press **Enter**.
3. Enter the path to the license file you received from RealNetworks, and press **Enter**. The installation process copies the license file to the License subdirectory under the main Helix Server directory. On startup, Helix Server reads that copy of the license.

4. Read the end-user license agreement, signifying your agreement to its terms by pressing **Enter**.

5. Enter a path where you want to install Helix Server, or accept the default path on Windows. Examples in this guide assume that you've chosen the default path.

   Note: On Windows, the default installation path for Helix Server differs from previous versions of RealSystem Server. For more information, see "Upgrading in a Different Directory" on page 46.

6. Enter a user name and password, and then confirm your password by entering it again. Your user name and password are required to access various Helix Server features, such as Helix Administrator. Choose a password that is difficult to guess, and that includes both letters and numbers. The password is case-sensitive.

7. In the next set of screens, you define ports that Helix Server uses for the RTSP, HTTP, and MMS protocols, as well as the port used by Helix Administrator. RealNetworks recommends accepting the default ports, unless those port values will cause conflicts with other applications. Note the following:

   - You can change the port settings later, as described in "Defining Communications Ports" on page 63.
   - If you use a nonstandard port for a streaming protocol, you will need to include the port number in URLs, as described in "Handling Communication through Nonstandard Ports" on page 65. For more information about streaming protocols, see "Application-Layer Protocols" on page 250.
   - You do not need to configure transport-layer protocols such as TCP and UDP. Helix Server and the client automatically select the transport protocol. After installation, you can restrict the UDP port range, as described in "Changing Port Assignments" on page 64. For background on TCP and UDP, see "Transport-Layer Protocols" on page 248.
   - You need the Admin port number to connect to Helix Administrator from a Web browser. As a security feature, the installer generates this port number randomly. RealNetworks recommends that you accept

the default, but you may change the value if you wish, or you know that the value will conflict with another port assignment. In either case, remember the port number, or record it in a secure location.

- On UNIX, the default value for the RTSP port is lower than 1024, meaning that you have to log in as `root` to start Helix Server if you accept the default value.

8. On Windows, the default installation sets up Helix Server as a service. This is recommended, but you can prevent this by unchecking the **Run as NT Service** box.

   Note: If you choose, you can later set up Helix Server to run as a service, as described in *Helix Server and Helix Proxy Troubleshooting Guide*.

   This installer page also presents the option to **Install SNMP Master as an NT Service**. If you check this box, the Simple Network Monitoring Protocol master agent is installed as a service. This optional feature is significant only if you have licensed the SNMP feature, which Chapter 18 explains.

9. In the final confirmation screen, review and accept the installation information to complete the installation process.

Upgrading in a Different Directory

If you are upgrading, and you install Helix Universal Server in a path that differs from that of your previous server installation, you need to move your existing content from the previous installation directory to the new directory after the installation. Content you need to move includes files in the `Content` and `Secure` directories, and, optionally, the `Logs` directory. If you are using authentication, you'll also need to move the files described in Appendix C.

If you plan to use a configuration file from an earlier server version, you need to edit the configuration information manually to reflect the new installation directory. Look for the variables that give full paths, and change their values accordingly.

Warning! Because editing the configuration file with a text editor can potentially disable Helix Universal Server, be sure to read Appendix A before attempting modifications.

Reinstalling Helix Server in the Same Directory

Reinstallation is generally not necessary, but if needed, you can reinstall Helix Server by repeating the installation procedure described in "Installing Helix Server" on page 44. A reinstallation does not affect media content, but it resets your Helix Server configuration values to their defaults. If you tailored your system configuration after the initial installation, the following tips allow you to retain your data and make your reinstallation process smoother:

- Back up the Helix Server configuration file (`rmserver.cfg`) and SNMP configuration file (`master.cfg`) to preserve the configuration information. After the reinstallation, replace the files created by the installer with your backups.
- Back up the `slta.cfg` file if you used that file for running SLTA. For more information on SLTA, see Chapter 10.
- Back up any authentication databases (`adm_b_db`, `con_r_db`, and so on) that you've revised or added. This step is necessary only if you've added more users and passwords for authentication than those added during installation. Appendix C explains authentication databases.
- Note the value of the `Admin` port (**Server Setup>Ports**). If you bookmarked Helix Administrator in your browser, specify the same `Admin` port during the reinstallation to keep the bookmark functional.
- A reinstallation does not affect cache files, access logs, or error logs. It is therefore not necessary to back up these files before reinstallation. These files typically reside in the `Cache` and `Logs` subdirectories of the main installation directory.

Running Helix Server

This section describes how to start and stop Helix Server on Windows and UNIX. For additional information about command-line options that you can use at start-up, refer to *Helix Server and Helix Proxy Troubleshooting Guide*.

Starting Helix Server

When you start Helix Server manually, you can select which configuration file you want to use. As described in "Restarting Helix Server" on page 53, you can use Helix Administrator to restart Helix Server following a configuration change.

Tip: The standard configuration file does not contain relative paths to components. However, a configuration file may be modified to include relative paths. In this case, you must start Helix Server from the directory that holds the configuration file to resolve the relative paths correctly. For this reason, the following sections show how to start Helix Server from its main installation directory, which contains the standard configuration file.

Starting on Windows

The following sections explain how to start Helix Server as a Windows service, from the **Start** menu, or from the Windows command line.

Changing the Windows Service Startup Parameters

In its default Windows installation, Helix Server is set up as a service named `Helix Server`. In this case, Helix Server always runs in the background, and you do not need to start it. You may wish to increase its memory maximum, however, as described in "Changing the Maximum Memory Usage of the Service" on page 49.

Starting Up from the Start Menu or Desktop

From the **Start** menu, select **Programs>Helix Server>Helix Server**. Or, double-click the Helix Server icon on the Windows desktop. This starts Helix Server with its default configuration file, `rmserver.cfg`, and a memory maximum of 256 MB.

To change the configuration file or to adjust the maximum memory usage, stop the server if it is running. Right-click the icon you use to start the server and select **Properties**. In the `Target` field, you can change `rmserver.cfg` to the name of a new configuration file. To increase the memory usage, add `-m 512` to the end of the field as shown in the following example, then restart the server:

```
"C:\Program Files\Real\Helix Server\Bin\rmserver.exe" "C:\Program
Files\Real\Helix Server\rmserver.cfg" -m 512
```

Note: You need to follow this procedure for each shortcut icon that you use to start Helix Server.

Starting Up from the Command Line

From the **Start** menu, open the command prompt. Navigate to the Helix Server folder, and enter the following command to start Helix Server with its default configuration file and standard memory use (256 MB):

```
Bin\rmserver rmserver.cfg
```

Optionally, you can use a different configuration file, as well as change the maximum memory allotment by including the `-m` parameter. After the `-m` parameter, specify the amount of memory in Megabytes (must be greater than 32). The following example allows Helix Server to use up to 512 Megabytes of memory:

```
Bin\rmserver rmserver.cfg -m 512
```

Changing the Maximum Memory Usage of the Service

By default, the Helix Server service uses a maximum of 256 Megabytes of memory. Follow the next procedure to change this amount.

➤ To change Helix Server memory usage:

1. Choose **Start>Settings>Control Panel**.
2. Double-click **Administrative Tools**
3. Double-click **Services**.
4. Locate `Helix Server` in the list, highlight it, right-click, and choose **Stop**.

   Note: Your service name may be different if you set up the service after installing the software, a procedure described in *Helix Server and Helix Proxy Troubleshooting Guide*.

5. Highlight `Helix Server` on the list, right-click, and choose **Properties**.
6. Navigate to the **General** tab.
7. In the **Start parameters** field, enter the `-m` parameter followed by the maximum amount of memory in Megabytes. For example:

   ```
   -m 512
   ```

8. Click **OK**.
9. Highlight `Helix Server` on the list, right-click, and choose **Start**.

Starting on UNIX

If you performed a default installation of Helix Server, the RTSP port is set lower than 1024, requiring the user who starts Helix Server to log in as `root`. If you do not want Helix Server to inherit `root` privileges, you can switch Helix Server to another user and group name immediately after it starts up. For instructions, refer to "Setting UNIX User and Group Names" on page 68.

You can start Helix Server as an application or as a background process. The following procedure uses the default configuration file (rmserver.cfg), but you can specify a different file.

➤ To start Helix Server on UNIX:

1. Start any command shell.
2. Navigate to the main Helix Server installation directory.
3. Choose one of the following options:
   a. Start Helix Server in the background with the following command:
      ```
      Bin/rmserver rmserver.cfg &
      ```
   b. Start Helix Server as an application:
      ```
      Bin/rmserver rmserver.cfg
      ```
   c. Optionally, you can adjust the amount of memory that Helix Server can use from the default value of 256 MB. Do this by including the -m parameter, where the number after -m specifies the amount of memory in Megabytes (must be greater than 32). The following example starts Helix Server as an application:
      ```
      Bin/rmserver rmserver.cfg -m 512
      ```
      The next example starts Helix Server as a background process:
      ```
      Bin/rmserver rmserver.cfg -m 512 &
      ```

Process ID (PID)

Helix Server creates a text file that records the current value of the process ID of the parent Helix Server process, rmserver. The file is stored in the directory indicated by the PidPath variable, and is named rmserver.pid at installation. If PidPath is omitted from the configuration file, Helix Server stores the information in the directory specified by the LogPath variable.

Stopping Helix Server

It's generally not necessary to stop Helix Server when it's running. If you make configuration changes that require a restart, you can restart through Helix Administrator, as described in "Restarting Helix Server" on page 53.

> **Note:** As described in "Implementing Redundant Servers" on page 107, you can redirect RealPlayers to other Helix Servers to finish their media sessions.

For More Information: The section "Implementing Delayed Shutdown" on page 70 explains how to allow media players to report playback statistics before the shutdown commences.

Shutting Down on Windows

If Helix Server was started as a Windows service, stop it through the **Services** control panel. Give the **Start>Settings>Control Panel** command. Then double-click **Administrative Tools** and **Services**. Locate Helix Server on the list (your service name may be different), highlight it, and click **Stop**.

If you started Helix Server manually, switch to the command window and press **Ctrl+c**. You can also use the Task Manager (**Ctrl+Alt+Del**) to end the Helix Server application.

Shutting Down on UNIX

To stop Helix Server on UNIX, obtain the parent process identification number, and then issue the kill command with that process number. The process ID is stored in the rmserver.pid file, which is usually kept in the Logs directory. (The PIDPath variable in the configuration file specifies this location.) You can perform both actions with one command. From the command line, navigate to the directory that contains the Helix Server PID file, and type the following, where *pidfile* is the name of the PID file:

```
kill `cat pidfile`
```

Using Helix Administrator

Helix Administrator is Helix Server's HTML-based, graphical user interface. It allows you to modify and manage Helix Server from a browser anywhere on your network. The following sections explain how to start Helix Administrator and use it to manage your Helix Server configuration.

Note: You can configure most Helix Server features by using Helix Administrator. However, some advanced features require manual editing of the Helix Server configuration file, as described in *Helix Server Configuration and Registry Reference*. Integrating Helix Server with other systems may also require manual configuration changes. For this information, refer to your systems documentation.

Starting Helix Administrator

To start Helix Administrator, you need to know the port number it uses, as well as the user name and password selected during Helix Server installation. The password is stored in the `MonitorPassword` variable of the configuration file. For background on the configuration file, see Appendix A.

➤ To start Helix Administrator:

1. Start Helix Server if it is not already running.
2. Click the browser shortcut added to the desktop by the Helix Server installer, or open the following location in your Web browser:

   `http://`*`address:AdminPort`*`/admin/index.html`

   If your browser is on the same computer as Helix Server, you can typically use the `localhost` address:

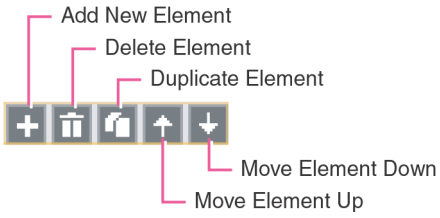
   `http://localhost:`*`AdminPort`*`/admin/index.html`
3. Enter the user name and password chosen during installation. The password is case-sensitive.
4. Click **OK** to start Helix Administrator.

> **Tip:** You can create additional user names and passwords to let other people access Helix Administrator. For more information, see "Administrator Authentication" on page 270.

Using the Interface

Helix Administrator consists of HTML pages that you use to configure Helix Server. The left-hand frame groups features into functional areas, as described in "Helix Administrator Sections" on page 54. Pages that display in the right-hand frame typically consist of forms that include fields and pull-down lists. In pages that list multiple elements, you can use the control icons depicted in the following illustration.

Helix Administrator Controls



When you change configuration information on a Helix Administrator page, click **Apply** at the bottom of the page to save the changes. An arrow appears next to the **Apply** button and the page title tab to indicate that changes require saving. A confirmation dialog appears when you click **Apply**. Note that Helix Administrator discards changes if you navigate to a different page before clicking **Apply**. As well, clicking **Reset** returns the current page to its stored values.

> **Tip:** If you are familiar with earlier versions of RealSystem Server, note that you no longer have to click an **Edit** button to update an element definition. You simply enter the new element information in the appropriate field, and click **Apply** at the bottom of the page to save the change.

Restarting Helix Server

Some configuration changes you make in Helix Administrator require a Helix Server restart, which breaks open connections for live events or clips streamed on demand. It's best, therefore, to make these changes during periods of low use. The Helix Administrator interface indicates feature changes that require a Helix Server restart. It also prompts you when a change requires a server restart when you click **Apply**. Click the **Restart Server** button to restart Helix Server.

> **Note:** As described in "Implementing Redundant Servers" on page 107, you can redirect RealPlayers to other Helix Servers to finish their media sessions.

> **For More Information:** The section "Implementing Delayed Shutdown" on page 70 explains how to allow media players to report playback statistics before the restart commences.

Queuing Changes for a Later Restart

It is not necessary to restart Helix Server immediately after you make a configuration change. In this case, the **Pending Changes** flag appears in the upper-right corner of Helix Administrator. This flag reminds you that all pending changes will go into effect the next time Helix Server is started.

Helix Administrator Sections

Helix Administrator's left-hand navigation pane groups Helix Server features under functional areas such as **Broadcasting**. Click the name of a functional area to expand or collapse the list of features it contains. The following tables summarize all features, and point you to the sections of this manual that explain each feature. Features vary according to your operating system and your license agreement, so you may not see all features listed here.

Server Setup

The server setup features let you configure the basic functions of Helix Server. Many of these features are preconfigured at installation.

Server Setup Features

Feature	Function	Reference
Ports	Define ports for communications protocols.	page 63
IP Binding	Select IP addresses Helix Server uses.	page 66
MIME Types	Create additional Web serving MIME types.	page 70
Connection Control	Limit connections by type or bandwidth.	page 73
Redundant Servers	Define failover servers for on-demand content.	page 107
Mount Points	Create mount points for on-demand content.	page 95
URL Aliasing	Shorten long URLs by creating aliases.	page 97
HTTP Delivery	Define Web serving directories.	page 68
Cache Directives	Control proxy caching and splitting.	page 119
Delayed Shutdown	Add time to close connections on a shutdown.	page 70
User/Group Name	Create UNIX user and group name.	page 68

Transport Settings

The transport settings affect how Helix Server delivers streams.

Transport Settings

Feature	Function	Reference
Differentiated Services	Mark IP packets for specific treatment.	page 79
Rate Management	Set up server-side congestion control.	page 74

Security

The security features let you limit connections to Helix Server, as well as set up user name and password validation for content viewers.

Security Features

Feature	Function	Reference
Access Control	Limit media player connections by IP address.	page 263
User Databases	Select authentication databases.	page 282
Authentication	Create authentication passwords and realms.	page 278 page 284
Commerce	Define commerce rules.	page 287

Logging and Monitoring

The logging and monitoring features let you view current Helix Server activity, as well as review past, recorded activity.

Logging and Monitoring Features

Feature	Function	Reference
Server Monitor	Display statuses of current connections.	page 369
Basic Logging	Define the basic access and error log files.	page 321
Advanced Logging	Create templates for advanced log reports.	page 353
SNMP	Implement Simple Network Monitoring Protocol	page 377

Broadcasting

Using the broadcasting features, you can unicast live events in any media format.

Broadcasting Features

Feature	Function	Reference
RealNetworks Encoding	Broadcast in the RealMedia format	page 138
QuickTime and RTP Encoding	Broadcast QuickTime or RTP-based media.	page 147
Windows Media Encoding	Broadcast Windows Media.	page 143
Live Archiving	Archive RealMedia broadcasts.	page 131
Broadcast Redundancy	Define backup encoder features.	page 135

Broadcast Distribution

Broadcast distribution builds on the basic broadcasting features, enabling you to multicast live events, as well as distribute broadcast streams to different Helix Servers.

Broadcast Distribution Features

Feature	Function	Reference
Transmitter	Set up a splitting transmitter.	page 200
Receiver	Define a splitting receiver.	page 205
Scalable Multicasting	Multicast to large numbers of RealPlayers.	page 164
Back-Channel Multicasting	Multicast to RealPlayers using a control channel.	page 161
Windows Media Multicasting	Multicast in the Windows Media format.	page 171
Session Announcement	Publicize a multicast automatically.	page 175

Content Management

The content management section groups useful features for managing on-demand clips.

Content Management Features

Feature	Function	Reference
Content Caching	Distribute on-demand content to various servers.	page 110
ISP Hosting	Provide streaming services for ISP customers.	page 303
Content Browsing	List all content stored on your Helix Server.	page 100
View Source	Make source markup and clip information available.	page 101

License File Information

The text-based license file resides in the License subdirectory of Helix Server's installation directory. It is in an XML format that you can read with any text editor. Making any changes invalidates the file, however. You can also display the license file through Helix Administrator by clicking **About**. You generally do not need to do anything with the license file, as long as Helix Server reads it correctly on startup.

Tip: If you have multiple license files, Helix Administrator shows the values for all of them at once. In this case, you need to read each file individually and calculate additive features, such as the total number of licensed streams.

Note: If all license files are invalid, Helix Server reports an error message and shuts down. To resolve this, contact RealNetworks for a valid license file.

Testing Your Installation

Click the **Media Samples** link in the upper-right corner of Helix Administrator to display a page containing links to sample clips. You can quickly test your installation by playing these clips if RealPlayer or another supported media player is installed on your computer. To play RealVideo 10 in RealPlayer, for example, click **Play RealVideo 10 Sample**.

If your Helix Server machine does not include a supported media player, you can play a sample clip from another machine on your network by logging into Helix Administrator from that machine. You can also open a clip directly in a media player. In RealPlayer, for example, give the **File>Open Location** command, then enter a media clip URL such as this:

```
rtsp://helixserver.example.com/realvideo10.rm
```

For More Information: Chapter 5 explains media clip URLs.

Quick Start Tutorials for Streaming Media

This section gives you step-by-step instructions for performing the basic tasks of streaming a prerecorded clip, and setting up a simple broadcast. These steps will familiarize you with these basic procedures. Keep in mind that there are many options for encoding, streaming, and broadcasting, as described in this guide and *RealProducer User's Guide*.

Quick Start Requirements

To perform these tasks, you'll need the following software and hardware on a computer other than the one that runs Helix Server:

- RealPlayer
- multimedia equipment:

- CD player and software
- sound card
- speakers

- RealProducer

  This section provides instructions for using the latest version of RealProducer. Although you can use an earlier version of RealProducer, the encoding steps will be different, and you should refer to your user's guide or online help for encoding instructions.

- Web browser
- HTML editor or text editor for creating an HTML page (optional)

> **Note:** RealProducer and RealPlayer are included with some Helix Server packages. They are also available in free download versions from the RealNetworks Web site at **http://www.realnetworks.com** and **http://www.real.com**.

Creating and Streaming a Clip on Demand

This section explains how to encode and stream a simple music clip. To do this, you'll need a music CD and RealProducer.

Step 1: Encode a Music Clip

This step encodes a streaming music clip directly from a music CD. Perform this step on the computer that has RealProducer installed.

➤ To encode the music clip:

1. Place a music CD in the computer's CD tray, and play it using the computer's CD player.

   > **Note:** RealJukebox and RealPlayer do not initialize the audio device needed for encoding. If one of these programs launches to play the CD, stop the playback, start the computer's general CD player, and play the CD.

2. Start RealProducer and give the **File>New Job** command.
3. In the left-hand input section, click the **Devices** radio button, and select the audio device from the **Audio** pull-down list.

4. Choose **File>Add Destination File**, set the file name `ondemand.rm`, and choose a directory for the clip.
5. Click the **Encode button**, wait at least one minute, and click **Stop**.

Step 2: Transfer the Music Clip to the Content Directory

Copy the `ondemand.rm` clip you created in the preceding step to the Helix Server `Content` directory. On Windows, the path is:

```
C:\Program Files\Real\Helix Server\Content
```

On UNIX, installation locations may vary, but paths look like this::

```
/usr/local/Real/HelixServer/Content
```

Step 3: Write a Web Page Link (Optional)

Create a link for the clip in an HTML page served by your Web server. Use the following link format, in which you substitute your Helix Server's computer name or IP address for *address*:

```
<a href="http://address:HTTPport/ramgen/ondemand.rm">Click here</a>
```

You do not need to include the HTTP port number if you selected port 80 during the installation. Here is an example:

```
<a href="http://helixserver.example.com/ramgen/ondemand.rm">Click here</a>
```

If your RealPlayer is on the same machine as your Helix Server, you can typically use the local host address:

```
<a href="http://localhost/ramgen/ondemand.rm">Click here</a>
```

> **For More Information:** The `/ramgen/` parameter, which is described in "Launching Media Players and Opening URLs" on page 88, causes the Web browser to start RealPlayer.

Step 4: Play the Clip

If you added the link to a Web page, browse the page and click the link. If you did not create a Web page link, launch RealPlayer, give the **File>Open Location** command, and enter the following URL:

```
rtsp://address:RTSPPort/ondemand.rm
```

You do not need to include the RTSP port number if you selected port 554 during the installation. Here is an example:

```
rtsp://helixserver.example.com/ondemand.rm
```

If your RealPlayer is on the same machine as your Helix Server, you can typically use the local host address:

```
rtsp://localhost/ondemand.rm
```

Broadcasting a Stream

This section explains the basic steps for broadcasting a stream without creating a clip. Perform this step on the computer that has RealProducer installed.

Step 1: Encode a Music Stream

This step sets up RealProducer to encode a continuous stream from the music CD, and send the stream to Helix Server.

➤ To create a live stream:

1. Place a music CD in the computer's CD tray, and play it using the computer's CD player.

   Note: RealJukebox and RealPlayer do not initialize the audio device needed for encoding. If one of these programs launches to play the CD, stop the playback, start the computer's general CD player, and play the CD.

2. Start RealProducer and give the **File>New Job** command.
3. In the left-hand input section, click the **Devices** radio button, and select the audio device from the **Audio** pull-down list.
4. Choose **File>Add Destination Server**. In the dialog box, leave the default settings, except for the following:
   a. Enter any destination name, such as **My Helix Server**.
   b. Use the stream name `live.rm`.
   c. For the server address, enter the IP address of the computer on which you installed Helix Server. If your RealProducer and Helix Server are on the same IPv4 machine, you can use `127.0.0.1`. For an IPv6 machine, use `::1`.
   d. In the **User Name** and **Password** boxes, type the same user name and password you use for logging in to Helix Administrator.
5. If you encoded an on-demand clip as described in the preceding tutorial, select that clip name in the **Destination** box, then click the trash icon to

delete it. Your destination server name should be highlighted in the **Destination** box.

6. Click the **Encode button**. After you have verified the broadcast in the following steps, you can turn off the encoding by clicking **Stop**.

Step 2: Write a Web Page Link (Optional)

Create a link for the clip in an HTML page served by your Web server. Use the following link format, in which you substitute your Helix Server's computer name or IP address for *address*:

```
<a href="http://address:HTTPport/ramgen/broadcast/live.rm">Click here</a>
```

You do not need to include the HTTP port number if you selected port 80 during the installation. Here is an example:

```
<a href="http://helixserver.example.com/ramgen/broadcast/live.rm">Click here</a>
```

If your RealPlayer is on the same machine as your Helix Server, you can typically use the local host address:

```
<a href="http://localhost/ramgen/broadcast/live.rm">Click here</a>
```

> **For More Information:** The `/ramgen/` parameter, which is described in "Launching Media Players and Opening URLs" on page 88, causes the Web browser to start RealPlayer.

Step 3: Play the Broadcast

If you created a Web page link, browse the page and click the broadcast link. If you did not create a Web page link, launch RealPlayer, give the **File>Open Location** command, and enter the following URL:

```
rtsp://address:554/broadcast/live.rm
```

You do not need to include the RTSP port number if you selected port 554 during the installation. Here is an example:

```
rtsp://helixserver.example.com/broadcast/live.rm
```

If your RealPlayer is on the same machine as your Helix Server, you can typically use the local host address:

```
rtsp://localhost/broadcast/live.rm
```

> **Note:** There may be few seconds of delay before playback commences. This slight broadcasting latency helps to ensure reliability.

CHAPTER

4

SERVER SETUP

This chapter describes basic Helix Server setup. These functions include binding to IP addresses and specifying ports. You may not need to change any of these settings depending on your system's configuration and the values you chose during installation.

Defining Communications Ports

After you install Helix Server, you can change the ports used for protocols such as RTSP and HTTP, as well as for features such as Helix Administrator. RealNetworks recommends that, whenever possible, you use the default communications ports, which are "well-known" ports that Web browsers and media players use by default when contacting Helix Server. The following table lists the default ports for the protocols that Helix Server uses.

Recommended Port Numbers

Protocol or Feature	Default Port	Purpose
RTSP	554	RTSP-based communication between Helix Server, RealPlayer, QuickTime Player, and RTP-based players.
MMS	1755	MMS-based communication between Helix Server and Windows Media Player.
HTTP	80	HTTP-based communication between Helix Server and Web browsers, as well as media players behind firewalls that necessitate HTTP cloaking.
Admin	(random)	Communication with Helix Administrator. The value is randomly generated during installation, and is required in the browser URL when connecting to Helix Administrator.
Monitor	9090	Communication with the Server Monitor. For more information on this feature, see Chapter 17.

For More Information: For more information about these protocols, as well as topics such as HTTP cloaking, see Chapter 11.

Changing Port Assignments

You can easily change ports through Helix Administrator. This requires a Helix Server restart, and if you change the Admin port, you'll also need to log into Helix Administrator again with the new port value.

➤ To change Helix Server port settings:

1. Click **Server Setup>Ports**.
2. Change values by entering new numbers for various ports. You can also change global options, described below, that affect how media players use Helix Server ports.
3. Click **Apply**. If you receive an error message that the port is used for UDP communications, choose another port, or restrict the UDP range as described below.

Enable Ramgen Port Hinting URLs

Set to **Yes** by default, this option instructs Helix Server to add protocol port information to all Web page links that use the `/ramgen/` mount point to launch RealPlayer. RealNetworks recommends that you leave this feature set to **Yes**. For additional information, see "Handling Communication through Nonstandard Ports" on page 65.

Enable HTTP Fail Over URL for ASXgen

If you set this option to **Yes**, Helix Server automatically includes an alternate HTTP URL to the streaming content in all Web page links that use the `/asxgen/` mount point. This allows Windows Media Player to request the content through HTTP if MMS communication is blocked.

UDP Resend Port Range

Initially blank, this option instructs Helix Server to use the specified range of UDP ports for media player replies. Enter a minimum range of two ports for each CPU on the Helix Server machine. "Media Players" on page 259 lists the standard UDP port ranges. For background information, see "Working with Firewall Technologies" on page 253.

Handling Communication through Nonstandard Ports

If you set the port for a communications protocol such as HTTP, RTSP, or MMS to a nonstandard value, media players may not be able to communicate to Helix Server because they won't know which port to use. When you choose the standard ports, which are set up by default during installation, clients always communicate on the correct ports. If you have set nonstandard ports, you can use port hinting, or add port numbers to URLs.

Port Hinting for RealPlayer and RealPlayer 8 through Ramgen

Port hinting tells a media player which communications ports are used for various protocols. This works only for RealPlayer 8 and later. If you keep the **Enable Ramgen Port Hinting URLs** feature at its default value of **Yes**, hinting occurs automatically when you launch RealPlayer through a Web page link that uses the Ramgen utility.

> **For More Information:** For instructions on using Ramgen, see "Using a Client Launch Utility" on page 90. For background information, see "Streaming to Client Software Behind Firewalls" on page 256.

Port Hinting for RealPlayer and RealPlayer 8 through a Ram File

When using a Ram file to launch a presentation for RealPlayer, content creators can include port hints with the `cloakport` parameter. The player then attempts to connect to the server using the specified protocol and ports. This is useful if you have multiple Helix Servers that use different ports. Content creators do not then have to know exactly which port is used on each machine. In the following example, RealPlayer attempts to connect through RTSP on port 554. If that doesn't work, it tries RTSP on port 550, and so on. You can specify up to four ports:

```
rtsp://helixserver.example.com/video1.rm?cloakport="554 550 1012 9120"
```

> **Note:** You'll need to inform content creators about the `cloakport` parameter, which is not discussed in RealNetworks production guides. For more about the Ram file, see "Using Metafiles" on page 88.

Port Numbers in URLs

You need to include the specific, nonstandard port number in URLs in cases where port hinting and the `cloakport` parameter don't work. This includes

communication with Windows Media Player, QuickTime Player, and RealPlayers earlier than version 8. This also applies to Web page links that use the HTTP protocol. For all protocols, you add the port number after the Helix Server address, separating the two with a colon. For example:

```
rtsp://helixserver.example.com:500/video1.rm
```

> **Note:** You'll need to alert content creators to the use of nonstandard ports so that they can write hyperlinks correctly. The section "Writing Links to Content" on page 85 provides more information about Helix Server URLs.

Binding to an IP Address

When Helix Server starts, it uses the IP address assigned to the first network interface it finds on the computer—network interface 0. In a computer with multiple network interfaces—often referred to as a *multi-homed* machine—you can configure Helix Server always to use specific IP addresses. Through this feature, you can select individual IP addresses to use, whether IPv4 or IPv6. Or, you can bind to all IP addresses on the machine.

> **For More Information:** For more about IPv6 addresses, refer to "IP Version 6" on page 26.

Using Localhost

By default, Helix Server binds to the IPv4 or IPv6 *localhost* address (also called the *loopback* address), which enables a simulated network connection from Helix Server to a client installed on the same computer. When using this address, which is useful for testing, no information is sent over the network, but it appears as if the connection came from the network. You can express the IPv4 localhost address in dotted decimal form as `127.0.0.1`. For IPv6, the localhost address is `0:0:0:0:0:0:0:1`, which can be shortened to `::1`.

Capturing All Addresses

You can use the IP binding feature to capture all addresses for Helix Server's use. To do this, specify one of the following options and delete all other address entries.

Syntax for Capturing All IP Addresses

Binding	Shorthand
All IPv4 addresses	`0.0.0.0`
All IPv6 addresses	::
All IPv4 and IPv6 addresses	*

> **Tip:** Helix Server automatically binds to all addresses and to localhost. For most installations, RealNetworks recommends binding to all addresses.

Modifying IP Addresses

You bind Helix Server to the machine's IP addresses using Helix Administrator.

➤ To reserve IP addresses for Helix Server:

1. In Helix Administrator, click **Server Setup>IP Binding**.
2. Click the "+" icon and type the IP address or binding option that you want Helix Server to use into the **Edit IP Address** box. Do not enter a DNS name.

   > **Note:** When you bind Helix Server to one or more specific IPv4 or IPv6 addresses, it also binds to the localhost address automatically. You can disable the automatic binding to localhost by specifying an IP address with the `--hbi` heartbeat check option when starting Helix Server. For more information, refer to *Helix Server and Helix Proxy Troubleshooting Guide*.

3. Repeat this procedure for each address on this machine that you want Helix Server to use.

   > **Warning!** If you use an option to capture all IP addresses of a certain type, as described in "Capturing All Addresses" on page 67, do not specify any other addresses of that type. For

example, if you specify `0.0.0.0`, do not include any other IPv4 addresses. If you do, Helix Server will not be able to start up.

4. Click **Apply**.
5. Restart Helix Server.

Setting UNIX User and Group Names

By default, Helix Server on UNIX uses the user and group names of the person who starts it. After startup, though, it can immediately switch to a different user and group setting. This lets you start Helix Server as `root`, so that it can capture port 554 for RTSP communications, then assume a different user and group identity. The user and group names must be predefined through the operating system, and must have write permission for Helix Server's `Logs` and `Secure` directories.

➤ To change the group or user names:

1. In Helix Administrator, click **Server Setup>User/Group Name**.
2. Type the user name or ID number in the **User Name or ID** box. The default is `%-1`, which means Helix Server uses the name of the user who logged in and started Helix Server.
3. Type the user name or ID number in the **Group Name or ID** box. The default is `%-1`, which means that Helix Server uses the group name of the user who logged in and started Helix Server.
4. Click **Apply**.

Configuring HTTP Communications

Although not intended to be used as a Web server, Helix Server can serve any content over HTTP. The following sections cover configuration options for extending HTTP delivery to specific content paths, and for defining MIME types. You generally do not need to change these settings unless you intend to use Helix Server for Web serving functions.

Allowing HTTP Delivery

By default, Helix Server restricts HTTP delivery to defined directories. This protects streaming media content from HTTP downloading, which may place

copies of downloaded clips in Web browser caches. To serve content in a new location over HTTP, you specify which mount points allow HTTP requests. The following mount points (and the subdirectories of their base paths) are preconfigured to allow HTTP delivery.

Mount Points that Allow HTTP Requests

Mount Point	Function
`/admin`	Delivering Helix Administrator pages. See "Starting Helix Administrator" on page 52.
`/ramgen`	Launching RealPlayer from a browser link. See "Ramgen for RealPlayer" on page 90.
`/httpfs`	Delivering any media or HTML page through HTTP.
`/viewsource`	Sending HTML pages that list source information and markup. See "Displaying Source Information" on page 101.
`/encfs`	Negotiating port settings between Helix Server and RealProducer.
`/asxgen`	Taking Windows Media requests from browsers. See "Using a Client Launch Utility" on page 90.
`/nscfile`	Launching Windows Media multicasts. See "Defining a Multicast Channel" on page 172.
`/scalable`	Taking scalable multicast requests from browsers. This is added automatically when you set up multicasting. See "Linking to a Scalable Multicast" on page 171.

Note: You typically do not need to change the settings of the preconfigured mount points. For general information about mount points, see "Mount Points" on page 86.

➤ To allow HTTP delivery for a mount point:

1. Define the mount point as described in "Adding a Mount Point for On-Demand Clips" on page 95. Subdirectories beneath the mount point's base path must not contain secure material. If they do, users will not be prompted for their name and password.
2. Click **Server Setup> HTTP Delivery**.
3. Click the "+" icon to add a new path.
4. Change the name in the **Edit Path** box to that of the mount point you created. Include only the initial forward slash, as in `/htmlpages`.
5. Click **Apply**.

Adding MIME Types for HTTP Communication

Because Helix Server acts as a Web server for certain features, it has its own MIME types section. You can modify Helix Server's list of MIME types if you plan to deliver over HTTP a data type not listed in the following table.

Helix Server MIME Types and Extensions

MIME Type	Extension
audio/x-pn-realaudio	ram
image/gif	gif
image/jpg	jpg, jpeg
text/html	html, htm
text/plain	txt
video/quicktime	mov

➤ To add a MIME type for HTTP serving:

1. Click **Server Setup>MIME Types** in Helix Administrator.
2. Click the "+" icon to add a new MIME type.
3. Enter the MIME type in the **Edit MIME Type** box.
4. Enter a file extension, or a list of comma-separated extensions, in the **Extensions** box. Do not precede file extensions with a period.
5. Click **Apply**.

Implementing Delayed Shutdown

The delayed shutdown feature lets you initiate a Helix Server shutdown or restart in a manner that allows media players to report playback statistics. This feature is useful for services in which it is necessary to know how much of a clip a media player received before the shutdown. If you do not implement this feature, Helix Server terminates all streams and shuts down immediately upon request, receiving no playback statistics from media players.

> **For More Information:** The sections "Stopping Helix Server" on page 50 and "Restarting Helix Server" on page 53 explain how to initiate a shutdown or restart, respectively.

Defining a Delayed Shutdown

The following procedure describes how to implement the delayed shutdown feature and define the shutdown intervals. This feature is not turned on by default.

➤ To define a delayed shutdown:

1. In Helix Administrator, click **Server Setup>Delayed Shutdown**.
2. For **Player Disconnect Interval**, enter the time in seconds that elapses between the shutdown request and cessation of all media streams. A value of `30`, for example, gives media players 30 seconds of streaming time. This allows players nearing the end of a clip the opportunity to finish the session normally. A value of `0` stops streams immediately, but still allows media players time to report playback statistics.
3. The **Shutdown Proceed Time** sets the number of seconds during which Helix Server terminates active streams and receives statistics from media players before shutting down. A value of `30` is typically sufficient. You may want to set a higher value if your Helix Server regularly streams more than 1,000 simultaneous clips.

   Tip: The shutdown delay is the sum of the two intervals. If you set 30 seconds for both the player disconnect interval and the shutdown proceed time, for instance, Helix Server shuts down 60 seconds after the request is given

4. If you select `Yes` in the **Log Player Termination Status** pull-down list, Helix Server records in its error log the number of successful and unsuccessful media player terminations that occurred before the shutdown. It records the following statistics:
   - Number of media players that ended their media sessions normally during the player disconnect interval.
   - Total number of media players whose media sessions were terminated by the shutdown.
   - Number of media players that successfully logged playback statistics in the access log file.
   - Number of media players that ignored the termination request.

   Note: The **Log Player Termination Status** setting affects only aggregate statistics for media player termination written to the

error log. It does not affect the collection of playback statistics from each media player.

For More Information: See "Basic Error Log" on page 322 for a description of the error log.

5. If you set the **Allow New Client Connections During Shutdown** pull-down to `No`, Helix Server accepts no new media requests once the shutdown request is given. If you set this value to `Yes`, Helix Server accepts new requests until the player disconnect interval elapses. For example, if you define a long disconnect interval, such as 300 seconds, you may want to allow connections during shutdown. This lets media players connect and receive all or part of a clip until the player disconnect interval expires.
6. Click **Apply**.

Tip: If you want to implement this feature with the predefined values, simply click **Apply** on the shutdown page.

Notes on Delayed Shutdown

Keep the following points in mind when implementing delayed shutdown:

- The delayed shutdown feature functions with RealPlayer, QuickTime Player, and Windows Media Player. Other media players may or may not respect the request to terminate playback and report playback statistics.
- On Linux and Sun Solaris, delayed shutdown occurs when you terminate Helix Server using `killall` or `pkill`, respectively. However, a `kill -9` command terminates Helix Server processes immediately without the shutdown delay.
- On Windows, delayed shutdown occurs when you terminate Helix Server using `Ctrl+c` or a `Stop Service` command. Stopping Helix Server through the Task Manager (`Ctrl+Alt+Del`) ends the processes immediately without the shutdown delay, however.
- During a shutdown, Helix Server does not notify an encoder transmitting a live stream of the shutdown. An encoder pushing a live stream to Helix Server may therefore try to reconnect to the server.

For More Information: See Chapter 7 for information about encoders and live broadcasts.

- If Helix Server is acting as a transmitter in a splitting arrangement, it does not notify receivers that the stream is about to end. It terminates the stream during the final shutdown.

  For More Information: Chapter 9 explains splitting.

- Delayed shutdown is not initiated if Helix Server shuts down or restarts due to a hardware malfunction or a software error, such as running out of memory.

Controlling Connections

Helix Server allows you to limit the number and types of media client connections made. As well, you can set a limit on the bandwidth that Helix Server uses for streaming. All settings are optional.

For More Information: For instructions about requiring a user name and password for client connections, see Chapter 13. See also the access control feature described in Chapter 12.

➤ To limit connections:

1. In Helix Administrator, click **Server Setup>Connection Control**.
2. In the **Maximum Client Connections** box, enter an integer to specify the total number of media players that can connect simultaneously. Once this limit is reached, players attempting to connect receive an error message, and will not be able to connect until other players disconnect.

   You can specify a number from `1` to `32767`. This number can be less than or equal to the number of streams permitted by your license, which is summarized in the **Maximum Licensed Client Connections** field. If the value is `0` or blank, Helix Server uses the number of streams specified by your license.
3. To stream media only to RealNetworks clients, set the **RealPlayers Only** pull-down to `On`. Other media players cannot then connect.
4. To stream media only to RealPlayer Plus and RealPlayer, set the **RealPlayer Plus Only** pull-down to `On`. Other media players, as well as any free versions of RealPlayer that predate RealPlayer, cannot then connect.

5. For **Maximum Bandwidth**, you can enter a number in Kilobits per second (Kbps). Helix Server refuses additional media player connections once this bandwidth limit is exceeded.

   Tip: Your Helix Server license may allow more bandwidth than is suitable for your network. Check with your network administrator to determine the right number to use.

6. The **Connection Timeout** field defines the number of seconds that can elapse without a client (either a media player or a proxy) sending any feedback to Helix Server. After this time expires, Helix Server queries the client for its status. If the client does not respond, Helix Server closes the connection.

   Note: This timeout value applies only to RTSP-based connections.

7. Click **Apply**.

Implementing Rate Control

The rate control feature allows Helix Server to adjust the bit rate of a prerecorded clip streamed to a media player based on information about the player's buffering characteristics, as well as periodic status reports from the player. This allows Helix Server to compensate for fluctuating bandwidth that may occur due to network congestion.

Note: The rate control feature does not function with live broadcasts.

Media Players that Support Rate Control

The rate control feature is the standard method by which Helix Server adjusts the streaming bit rate for the following media players:

- Desktop RealPlayer 11

Use of the rate control feature, however, further depends on three factors:

- Helix Server must have information about the media player's buffering characteristics. As described in "Buffer Modeling" on page 75, Helix Server adjusts streaming rates based on a model of the player's buffering state.

The section "Device Capability Exchange" on page 76 explains how Helix Server determines the media player's buffering characteristics.

- Helix Server must receive periodic status reports from the media player to gauge network congestion and ensure that its buffering model is accurate. For information, refer to "Receiver Reports" on page 77.
- Helix Server can shift streaming rates only if the requested clip is encoded in a multi-rate format. The section "Media Formats Used with Rate Control" on page 77 lists the acceptable formats.

If a media player does not meet the conditions required for server-side rate control or client-side stream switching through SureStream, Helix Server delivers a single-rate stream.

Buffer Modeling

To implement appropriate rate control for each media player, Helix Server maintains a model of each player's data buffer. This allows it to gauge how much data is in the player's buffer and change streaming rates appropriately:

- If the data in a player's buffer falls too low, Helix Server may shift to a lower-bandwidth stream. A low buffer indicates that the network cannot deliver data to the buffer as quickly as the media player removes packets from the buffer. Shifting to a lower bandwidth causes the player to take packets out of the buffer at a slower rate. This, in turn, allows Helix Server to build up the buffer at the slower network rate. This data rate downshifting therefore helps prevent rebuffering, an undesirable condition in which the player must halt the presentation to refill an empty buffer.
- Conversely, if the player's buffer fills, Helix Server may shift to a higher-bandwidth stream. A full buffer indicates that the network is capable of delivering data faster than the media player uses it. Shifting to a faster encoding rate causes the player to consume packets in the buffer faster. By upshifting to a higher data rate, Helix Server delivers a higher-quality user experience, and prevents data packets from being lost on the network because the media player's buffer was too full to accept them.

Helix Server does not shift data rates with every fluctuation in network capacity. It does so only when it detects a persistent change in network capability that necessitates downshifting or facilities upshifting. In some cases, rebuffering may be unavoidable. The rate control feature helps to

minimize rebuffering by adjusting the streaming rate to the optimal choice, given the network's current conditions.

Device Capability Exchange

To model a media player's buffering status, Helix Server must know the player's buffering capabilities. Using the capabilities exchange feature, Helix Server can learn the media player's buffering capacity. The capabilities exchange feature is set up by default and typically does not need to be modified. Helix Administrator does not include fields to define capabilities exchange.

RDF Files

When a media player contacts Helix Server over HTTP or RTSP, it indicates the Internet location of an RDF file, which stands for Resource Description Framework. The XML-based RDF file, which uses the file extension `.rdf`, describes the player's streaming capabilities. Helix Server contacts the server and downloads the RDF file, caching it for use when delivering streams to additional media players of the same type.

> **Note:** A media player can send Helix Server the RDF file location in the `x-wap-profile` header of an HTTP request or any RTSP message. The media player can also send supplemental information in the `x-wap-profile-diff` header.

RDF File Overrides

Not all media players indicate locations of RDF files. As well, the information provided in the player's RDF file may not be appropriate for all situations. To overcome these problems, you can assign a media player a specific RDF file through the Helix Server configuration file. For information about editing the configuration file, refer to the rate control chapter of *Helix Server Configuration and Registry Reference*.

Server Components for Capabilities Exchange

Helix Server includes the following components used with capabilities exchange:

- The `ClientProfiles` directory is created under the Helix Server directory during installation. This directory holds the RDF files, and should not be used for any streaming content.

- The predefined /profiles/ mount point allows Helix Server to read from and write to the ClientProfiles directory. This mount point is not used in URLs for streaming content.

  For More Information: For background on how mount points work, refer to "Mount Points" on page 86.

Receiver Reports

Because Helix Server controls stream rates based on its own buffer model for the media player, it must receive periodic updates from the player to ensure that its model is accurate. Media players that use the RTP packet format periodically send Helix Server status reports that allow it to determine the stream packet loss rate, as well as the time required for packets to travel through the network. Using this information, Helix Server verifies its player buffer model and changes the streaming rate as necessary.

Note the following about receiver reports:

- For rate control to function, a media player must send RTCP receiver reports to Helix Server every one to five seconds.

  For More Information: For more on receiver reports, refer to RFC 1889 at **http://www.ietf.org/rfc/rfc1889.txt**.

- Although Helix Server can function with players that send receiver reports less frequently, longer reporting intervals decrease Helix Server's ability to adjust streaming rates in a timely manner.
- RealNetworks media players communicate to Helix Server using the proprietary RDT packet format. They report information similar to the RTCP receiver report, but do so more frequently. For information on RDT, refer to "Packet Formats" on page 251.

Media Formats Used with Rate Control

For rate control to function, Helix Server must deliver a clip that includes multiple streams encoded at different bit rates. For RealNetworks media players, you can use SureStream.

SureStream Clips for RealNetworks Players

As described in "SureStream RealAudio and RealVideo" on page 20, a SureStream clip encodes multiple streams at different bit rates. The

mechanism used to control the shifting of bit rates during playback varies according to the media player version:

- For desktop RealNetworks media players earlier than RealPlayer 11, Helix Server uses its older, client-side stream selection mechanism. This causes RealPlayer to request upshifting or downshifting based on its buffer state.
- For RealPlayer 11 and later, Helix Server manages all stream shifting based on its rate control model. This model allows delivery of all existing SureStream clips.

Defining Rate Control

Several fields in Helix Administrator allow you to configure rate control features that affect how Helix Server adjusts streams on a congested network. These fields affect only the default rate control settings within the configuration file. To change the values for a specific media player, you must edit that player's configuration settings manually.

> **For More Information:** Refer to the rate control chapter in *Helix Server Configuration and Registry Reference*.

➤ To define rate control:

1. In Helix Administrator, click **Transport Settings>Rate Control**.
2. The **Player Report Bandwidth Percentage** field reserves bandwidth for periodic feedback reports from the media player. The value is an integer that, scaled down by a factor of 10,000, represents a percentage of the media's stream rate. For example, with a clip streaming at 20 Kbps, the default value of `200` equals 2 percent of the stream speed, or about 410 bits per second of additional bandwidth reserved for player reports.

   > **Note:** Setting a higher value provides the media player with more bandwidth to send reports, and may increase quality of service on a highly congested network. An increased frequency of reports is not guaranteed, though, and different media players may respond differently to this setting.

3. The **Server Report Bandwidth Percentage** field holds an integer value that reserves bandwidth for Helix Server to send status reports to media players. These reports indicate basic server statistics such as the number of packets sent in the current session. The field value is scaled down by a

factor of 10,000. Hence, the default value of `100` equals 1 percent of the clip's streaming bandwidth, or about 205 bits per second for a clip encoded to stream at 20 Kbps.

> **Tip:** Some media players may use these server reports to modify the streaming session. Other players may ignore the reports.

4. The **Maximum Packet Number** field sets the maximum number of packets sent in a transmission scheduling window. The value is an integer in the range `0` to `1024`. The default value is `3`. For a 20 Kbps clip, for example, the scheduling window is approximately 6.6 seconds if the packet size is 1 Kb.
5. For **Excess Available Bandwidth Percentage**, use an integer that defines a percentage of the media streaming rate that Helix Server can fully utilize. The value must be greater than 100. For example, the default value of `110` enables Helix Server to use 22 Kbps of network bandwidth when streaming a clip encoded at 20 Kbps. Helix Server may use the additional 2 Kbps to fill a media player's buffer, for example, based on feedback from the player.
6. In the **Maximum Bandwidth Per Connection** field, define the total amount of bandwidth in bits per second that Helix Server can use for each connection. The default is `24000`, which is approximately 24 Kbps. When determining which stream from a multi-rate clip to deliver to a media player, Helix Server takes into account this maximum rate, the allowable excess bandwidth percentage, and the bandwidth percentages reserved for client and server reports.

   > **For More Information:** To set a limit on the total amount of outgoing bandwidth, refer to "Controlling Connections" on page 73.

7. Click **Apply**.

Configuring Differentiated Services

The differentiated services feature allows you to assign priorities to IPv4 packets that carry streaming media data over the RTSP protocol, and the RTP or RDT packet formats. These priority values can affect how the packets are forwarded through the network by routers that support differentiated services. For example, these values can cause routers to delay the forwarding of

low-priority packets when network traffic is high. Using differentiated services can thereby increase a network's efficiency by dividing packets into different classes that are assigned different delivery criteria.

> **Note:** The differentiated services feature functions only with IP version 4 (IPv4). It does not support IP version 6 (IPv6) packets. The priority assignments are ignored if IPv6 is used.

> **For More Information:** The Helix Server implementation of differentiated services complies with IETF standards described on the Web pages **http://www.ietf.org/rfc/rfc2474.txt** and **http://www.ietf.org/rfc/rfc2475.txt**.

Network Requirements for Differentiated Services

Marking packets for differentiated services has an effect only on networks in which each node through which a packet passes is configured to read the IPv4 packet's differentiated services field (`DSFIELD`), and forward the packet based on the field's encoded criteria. This limits the benefits of differentiated services criteria to an intranet on which each router is enabled for these services, and the network administrator has defined rules for forwarding packets based on the `DSFIELD` values.

> **Note:** You can still stream media on networks that are not enabled for differentiated services, such as the Internet. However the priority values have no effect on packet forwarding, and each packet is treated the same.

IP Header Bit Values

The settings of six bits within each IPv4 packet header define the differentiated services criteria.

Precedence Bit Settings

Three bits within an IP packet head define a precedence, which categorizes each packet. The following table lists the eight possible bit combinations that define each precedence. A precedence setting has no inherent meaning. Setting an IPv4 packet to the `Priority` precedence, for example, has significance only within a network that is configured to handle packets of this precedence in a specific way under certain network conditions. This packet handling can vary

from network to network, and is wholly defined by each network administrator.

Precedence Bit Settings

Precedence	Value	Bit Setting
Routine	0	000
Priority	1	001
Immediate	2	010
Flash	3	011
Flash override	4	100
CRITIC/ECP	5	101
Internetwork control	6	110
Network control	7	111

Quality of Service Bit Settings

Following the three precedence bits, three bits define the quality of service assigned to each precedence. As shown in the following table, these bit settings instruct network routers about the packet's desired throughput, reliability, and delay. As with the precedence values, the quality of service values are inherently arbitrary. For example, how a network handles a packet marked as "low delay" depends on how the network measures congestion, as well as the rules that the network administrator has defined for handling "low delay" packets.

Quality of Service Bit Settings

Service Category	Bit Setting
Delay	0–normal delay 1–low delay
Throughput	0–normal throughput 1–high throughput
Reliability	0–normal reliability 1–high reliability

Configuring Differentiated Services

Follow the next procedure to configure differentiated services for streaming media IPv4 packets.

➤ To define differentiated services:

1. In Helix Administrator, click **Transport Settings>Differentiated Services**.
2. In the top set of fields, you can set the precedence and quality of service for streaming media packets in the RTP and the RDT formats. In the bottom set of fields, you can define the precedence and quality of service for packets of the RTSP control protocol. These settings do not affect packets for other protocols, such as MMS and HTTP.

   For More Information: For more on RTP and RDT, see "Packet Formats" on page 251. For information on RTSP, see "Application-Layer Protocols" on page 250.

3. For a chosen protocol, select the packet precedence in the **Precedence** pull-down list. As described in "Precedence Bit Settings" on page 80, your network needs to define the characteristics for handling this precedence.
4. After setting a precedence for the packets, define their quality of service in the **Delay**, **Throughput**, and **Reliability** pull-down lists. As described in "Quality of Service Bit Settings" on page 81, your network needs to define the rules for handling the selected quality levels.
5. Click **Apply**.
6. Restart Helix Server.

   Note: On Linux distributions, Helix Server must run as `root` to use the differentiated services feature. This restriction does not apply to other UNIX operating systems or Windows.

PART

III

STREAMING CLIPS

In this section, you'll learn how to stream prerecorded clips, and create links to media content. You'll also learn about useful features that you can employ if you have a large network of servers.

CHAPTER

5

CLIP DELIVERY

Helix Server is ready to stream prerecorded clips right after installation. As with any Internet server, you request content from Helix Server using a hypertext URL, which you typically add to a Web page. This chapter explains link formats, introducing you to *mount points*. It shows you how to launch a media player from a Web page, and covers additional features that help you deliver prerecorded clips.

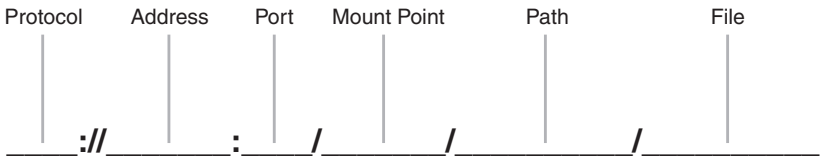
Writing Links to Content

A Web page hyperlink to content on Helix Server launches a media player and streams some content, whether a prerecorded clip or a live broadcast. A typical link to a media clip or a broadcast served by Helix Server includes the server address, protocol port (optional), mount points, path, and file name:

`protocol://address:port/mount_points/path/file`

The following illustration shows the parts of a typical link. Not every link includes the same components, however.

Parts of a Link



The following table describes common link components.

Helix Server URL Components

Component	Specifies
protocol://	The protocol used to initiate streaming. This is rtsp://, mms://, or http://. For more information on these protocols, see "Application-Layer Protocols" on page 250.
address	Address of Helix Server. Use one of the following: – dotted IPv4 decimal address, such as 163.112.11.12. – colon-delimited, IPv6 hexadecimal address enclosed in square brackets, such as [2001:db8::1]. – machine and domain name, such as helixserver.example.com. Where possible use a domain name so that media players can resolve the name to an IPv4 or IPv6 address as appropriate.
port	Port where Helix Server listens for requests on the specified protocol. This is not required if Helix Server uses the default, well-known ports. For more on ports, see "Defining Communications Ports" on page 63.
mount_points	One or more mount points that invoke Helix Server features. You can stream simple, on-demand clips using just a forward slash after the address and port. For more information, see "Mount Points" on page 86.
path	An optional path relative to the Content directory. If the file is in the Content directory, no path is required. Broadcast URLs do not include a path. For more information, see "Creating Content Subdirectories" on page 94.
file	The name of the presentation, including the extension. The file can be a clip or a metafile used to launch the clip, as described in "Launching Media Players and Opening URLs" on page 88. Broadcasts also use a file name specified by the encoder, even if no file is created on the computer.

Mount Points

All URLs to clips or live streams served by Helix Server include at least one mount point that looks like a directory listing in the request URL:

```
<a href="http://helixserver.example.com/ramgen/video1.rm">Play Video</a>
```

In the preceding URL, the /ramgen/ mount point does not correspond to a physical path on the Helix Server computer. Instead it's a virtual path that invokes a Helix Server feature. In this case, the mount point tells Helix Server to use its Ramgen feature to send the browser a MIME stream that launches RealPlayer.

> **For More Information:** For more on Ramgen, see "Using a Client Launch Utility" on page 90.

Content Directory Mount Point

Using Ramgen launches RealPlayer and passes it the media URL. The URL that RealPlayer uses, however, does not include the /ramgen/ mount point, which has already served its purpose. Additionally, the URL is reconfigured to use the RTSP streaming protocol. Hence, RealPlayer receives the following URL for requesting the streaming content:

```
rtsp://helixserver.example.com/video1.rm
```

This URL also contains a mount point, though it's easy to miss because it's simply the slash that precedes the clip name:

```
.../video1.rm
```

This mount point indicates that the clip resides in Helix Server's main content directory. This directory, named Content, is located in the Helix Server installation directory. A directory linked to a mount point is known as the mount point's *base path*. In a default installation on Windows, the base path for the "\" mount point is the following:

```
C:\Program Files\Real\Helix Server\Content
```

On UNIX, installation locations may differ, but the content directory falls under the main Helix Server directory:

```
/usr/local/Real/HelixServer/Content
```

The "/" mount point therefore masks the actual path to the streaming media content on the Helix Server computer.

> **For More Information:** "Streaming Clips On Demand" on page 94 contains additional information about paths and mount points for streaming on-demand clips.

Multiple Mount Points

A URL may have several mount points. Here, the /ramgen/ mount point launches RealPlayer, whereas the /broadcast/ mount point indicates that the content is live rather than prerecorded:

.../ramgen/broadcast/live.rm

When a URL includes /broadcast/, Helix Server does not search its content directory for a clip named live.rm. Instead, it knows that the URL is to a live stream sent by RealProducer. The configuration information for the /broadcast/ mount point gives Helix Server the information it needs to find the content, such as what server port RealProducer uses to deliver the stream.

Helix Server predefines mount points for many features. In some cases, you need to define your own mount points to set up a feature. When a feature requires a mount point, this guide's instructions on implementing that feature explain how to configure the mount point, as well as how to add the mount point to URLs to invoke the feature. In many cases, URLs must list mount points in a specific order for the various features to work.

Launching Media Players and Opening URLs

A clip that streams over the RTSP or MMS protocol must use a URL that starts with rtsp:// or mms:// rather than with http://. Because browsers cannot make RTSP or MMS requests, you cannot link a Web page directly to a streaming media clip that uses a streaming protocol. You resolve this problem either by adding a metafile, or using a client launch utility on Helix Server.

> **Tip:** Most media players also have an **Open** command that lets viewers enter a media URL directly without clicking a link. In this case, the URL is the same as that used in a metafile. Although viewers can play clips this way, this method is not recommended because it is awkward and prone to errors.

Using Metafiles

A metafile is a simple text file that links to a Web page through a standard <a href> link that specifies the HTTP protocol. The metafile contains the streaming media URL, and uses a file extension that causes the Web browser to launch a media player as a helper application. The browser then passes the metafile to the media player, which requests the streaming content over a streaming protocol. Each media player uses a different type of metafile.

RealNetworks Ram File

For RealPlayer, metafiles are known as *Ram files* because they use the file extension `.ram`. If the media is embedded in a Web page, though, the file extension is `.rpm`. When the viewer clicks the Web page link to the Ram file, the browser launches RealPlayer and gives it the Ram file, which contains the URL to the streaming clip, such as the following:

```
rtsp://helixserver.example.com/video1.rm
```

RealPlayer can also play SMIL presentations that coordinate multiple clips. In these cases, the SMIL file lists the RTSP URLs to the clips. The Ram file gives the RTSP (or HTTP) URL to the SMIL file.

> **For More Information:** The presentation delivery chapter of *RealNetworks Production Guide* covers the Ram file syntax, and explains how content creators can use the Ram file to pass playback parameters to RealPlayer.

> **Tip:** A Ram file feature called a *cloakport switch* helps you to mitigate the effect of restrictive firewalls on RTSP clients. This switch passes Helix Server port information to the client in case the RTSP connection is blocked by a firewall. See "Handling Communication through Nonstandard Ports" on page 65.

QuickTime Reference Movie

Although Apple's QuickTime Player also receives streaming content over RTSP, it has its own metafile extension and syntax. You use the QuickTime **MakeRefMovie** tool to create a reference movie that contains the RTSP URL or URLs to your QuickTime presentation on Helix Server.

> **For More Information:** See the QuickTime documentation for information on **MakeRefMovie**.

Windows Media ASX File

ASX files are the Windows Media equivalent to RealNetworks Ram files. They are text files that use the file extension `.asx`. They list the URLs to Windows Media clips that stream over the MMS or HTTP protocol. Within the ASX file, the requested content uses an `mms://` link that points to the content on Helix Server. You must also include an `http://` link to the same content that gives the Helix Server HTTP port number. For example:

```
<ASX Version = "3.0">
   <ENTRY>
      <Ref href = "mms://helixserver.example.com:1755/wmvideo.wmv"/>
      <Ref href = "http://helixserver.example.com:8080/wmvideo.wmv"/>
   </ENTRY>
</ASX>
```

> **For More Information:** See your Windows Media documentation for instructions about writing ASX files. The section "Windows Media Player 11 and Later" on page 23 explains the requirement for the HTTP URL in the ASX file.

Using a Client Launch Utility

As an alternative to writing a metafile, you can use a client launch utility that allows you to link your Web page directly to a streaming clip. In this case, the Web page URL used to request the streaming clip must include the mount point for the launch utility. This causes Helix Server to launch the appropriate media player, and stream the clip using the player's preferred streaming protocol. Helix Server includes launch utilities for RealPlayer and Windows Media Player. QuickTime has no equivalent utility.

Ramgen for RealPlayer

Helix Server's Ramgen utility is preconfigured with a /ramgen/ mount point that launches RealPlayer. To use the Ramgen utility, you include the /ramgen/ mount point in a Web page URL to a streaming clip, just after the Helix Server address. To enable the browser to make the request, you use the HTTP protocol rather than RTSP:

```
<a href="http://helixserver.example.com/ramgen/video1.rm">Play RealMedia</a>
```

When Helix Server receives a request that contains the /ramgen/ mount point, it sends a MIME stream to the browser that launches RealPlayer. It then streams the clip or SMIL file to RealPlayer using the RTSP protocol. Hence, the preceding Ramgen link is equivalent to linking to a Ram file from a Web page:

```
<a href="http://www.example.com/launch_video.ram">Play RealMedia</a>
```

and having the launch_video.ram file request the clip over RTSP:

```
rtsp://helixserver.example.com/video1.rm
```

Note: To circumvent restrictive firewalls, Helix Server adds the cloakport switch when using Ramgen. For more information, see "Handling Communication through Nonstandard Ports" on page 65. See also "Streaming through NAT Firewalls" on page 92.

For More Information: Content creators can learn about Ramgen from the presentation delivery chapter of *RealNetworks Production Guide*. Note that Ramgen does not support as many playback parameters as a Ram file, which makes the Ram file a more powerful tool for content creators.

ASXgen for Windows Media Player

ASXgen is the equivalent to Ramgen for launching Windows Media Player. Helix Server comes configured with a mount point named `/asxgen/`, which you add to a Web page link instead of writing an ASX file. When Helix Server receives a request that contains the `/asxgen/` mount point, it sends a MIME stream that causes the browser to launch Windows Media Player.

To stream to Windows Media Player 6.4, add the `.asx` extension to an ASXgen link for an `.asf`, `.wmv`, or `.wma` clip. The extra `.asx` extension is not necessary if you are streaming exclusively to Windows Media Player 7 and later, however. In the following example, the URL is for `video2.wmv.asx`, even though the clip on Helix Server is `video2.wmv`. When it receives the request, Helix Server strips off the extra `.asx` extension:

```
<a href="http://helixserver.example.com/asxgen/video2.wmv.asx">Play Windows Media</a>
```

Once Windows Media Player launches, Helix Server streams the requested content using the MMS protocol. To circumvent restrictive firewalls, Helix Server also transmits an HTTP URL that duplicates the MMS link, enabling HTTP cloaking for Windows Media Players behind firewalls. This link is also used by Windows Media Player 11 and later, which no longer support the MMS protocol.

For More Information: For information on enabling HTTP cloaking for Windows Media Player, see "Changing Port Assignments" on page 64. See also "Streaming through NAT Firewalls" on page 92. The section "Windows Media Player 11 and Later" on page 23 explains that player's use of HTTP.

Streaming through NAT Firewalls

If your Helix Server resides behind a Network Address Translation (NAT) firewall, client launch utilities and metafiles may fail because they instruct the media player to contact the firewall rather than Helix Server. To prevent this, you can add to the configuration file a HostName variable that supplies media players with the Helix Server IP address.

In the metafile or launch utility URL, you then add usehostname as a query string parameter, separated from the URL by a question mark (?). For example:

```
<a href="http://helixserver.example.com/ramgen/video1.rm?usehostname">
Play the Video</a>
```

The usehostname value can come anywhere within a string of multiple query parameters. Use an ampersand (&) to separate it from other parameters, as demonstrated in the following metafile example:

```
rtsp://helixserver.example.com/video1.rm?cloakport="550 9120"&usehostname
```

> **For More Information:** For information about configuring the HostName variable, refer to the chapter on basic streaming features in *Helix Server Configuration and Registry Reference*.

Tips for Launching Media Players

Keep the following points in mind when linking Web page to streaming media clips.

Do Not Use Both Metafiles and Launch Utilities

Metafiles and the Ramgen and ASXgen utilities are mutually exclusive. If you launch content with a Ram file, for example, make sure that the requested URL does **not** contain the /ramgen/ parameter.

Incorrect URL in a Ram File

```
rtsp://helixserver.example.com/ramgen/video1.rm
```

Correct URL in a Ram File

```
rtsp://helixserver.example.com/video1.rm
```

Conversely, when using Ramgen or ASXgen, make sure that your Web page link points to a streaming clip, not to a metafile.

Incorrect File Request in a Web Page URL Using Ramgen

```
<a href="http://helixserver.example.com/ramgen/play.ram">Play RealMedia</a>
```

Correct File Request in a Web Page URL Using Ramgen

```
<a href="http://helixserver.example.com/ramgen/video1.rm">Play RealMedia</a>
```

Use the Correct Protocol

Remember that media clips and broadcasts stream over RTSP or MMS, but Web browsers can make requests only over HTTP. Specifying an incorrect protocol can prevent a presentation for launching, or degrade its playback. When using metafiles, ensure that URLs use streaming protocols.

Incorrect Protocol in a Ram or SDP File

```
http://helixserver.example.com/video1.rm
```

Correct Protocol in a Ram or SDP File

```
rtsp://helixserver.example.com/video1.rm
```

When using Ramgen or ASXgen, make sure that your Web page link uses HTTP, rather than a streaming protocol such as RTSP or MMS.

Incorrect Protocol in a Web Page URL Using Ramgen

```
<a href="rtsp://helixserver.example.com/ramgen/video1.rm">Play the Video</a>
```

Correct Protocol in a Web Page URL Using Ramgen

```
<a href="http://helixserver.example.com/ramgen/video1.rm">Play the Video</a>
```

Use Streaming Media URLs in SMIL Files

Content creators may create SMIL presentations for RealPlayer. In this case, you treat the SMIL file as if it were a streaming media clip. That is, you link to the SMIL file from a Ram or SDP file:

```
rtsp://helixserver.example.com/presentation.smil
```

or in a Web page link that uses Ramgen:

```
<a href="http://helixserver.example.com/ramgen/presentation.smil">...</a>
```

The SMIL file itself then contains URLs to the various clips. These are typically RTSP URLs such as those you write in a Ram or SDP file:

```
<video src="rtsp://helixserver.example.com/video1.rm" region="video_region".../>
```

> **For More Information:** *RealNetworks Production Guide's* clip source tags chapter explains SMIL URLs.

Communicate with Content Creators

You will need to give some information to content creators so that they can transfer their content to Helix Server and write valid hyperlinks. The main

reference manual for content creation, *RealNetworks Production Guide*, explains protocols, Ram files, and the Ramgen utility. However, content creators must rely on the Helix Server administrator for the following information:

- Helix Server address
- whether the port must be specified along with the protocol
- mount points (other than `/ramgen/`) to use to implement specific Helix Server features
- method for transferring content to Helix Server, such as file copy or FTP
- paths to the streaming content when it is transferred to Helix Server

Streaming Clips On Demand

Helix Server is ready to stream clips as soon as it starts up. All you do is place a clip in the `Content` directory and write a link as described in "Writing Links to Content" on page 85. The following sections provide information about additional, optional ways to configure Helix Server to stream clips on demand.

Creating Content Subdirectories

On your Helix Server computer, you can create subdirectories under the predefined `Content` directory, which is described in "Content Directory Mount Point" on page 87. On Windows, for example, you might create a `video` subdirectory in the `Content` directory. In a default Helix Server installation on Windows, the full path would be the following:

```
C:\Program Files\Real\Helix Server\Content\video
```

In an installation on UNIX, it may look like this:

```
/usr/local/Real/HelixServer/Content/video
```

A metafile link to a video clip in this subdirectory would include the `video` subdirectory, but not the path to the main `Content` directory:

```
rtsp://helixserver.example.com/video/video1.rm
```

If, for example, the link occurred in a Web page and used Ramgen, the `video` subdirectory listing would follow the `/ramgen/` mount point (and any other mount points) in the URL

```
<a href="http://helixserver.example.com/ramgen/video/video1.rm">Play</a>
```

Tips for Using Content Subdirectories

- Just by examining a link, you can't determine which parts of a link refer to mount points, and which parts refer to subdirectories. You must be familiar with the mount points set up on Helix Server to recognize the mount points in use.
- Helix Server looks through the list of mount points before it looks for directory names. Should a mount point have the same name as an actual directory, Helix Server will use the mount point and ignore the actual directory.
- You can create aliases for long subdirectory listings, as described in "Setting Up Aliases" on page 97.
- When broadcasting live events, you can take advantage of mount points and directories that have the same names to display a standby message if the broadcast has not begun, or is interrupted. See "Playing a Standby Message" on page 153.

Adding a Mount Point for On-Demand Clips

If possible, place all of your streaming clips in the `Content` directory or its subdirectories. If necessary, though, you can stream content from other directories, disks, or networked machines. Instead of using symbolic links or aliases from the `Content` directory to other directories, however, you create new mount points for on-demand clips. Each mount point specifies a base path where Helix Server locates the files. You then use the new mount point in URLs for on-demand clips located in the new directory.

➤ To create a new on-demand mount point:

1. In Helix Administrator, click **Server Setup>Mount Points**.
2. Click the "+" icon to add a new mount point.

   Tip: You can also edit an existing mount point by selecting it from the list, changing the information, and clicking **Apply**. You can change the base path of the `Content` directory, for example, by editing the `RealSystem Content` mount point definition.

3. Enter a new, unique name to replace the generic mount point name that appears in the **Edit Description** box.

4. Specify the mount point as it will appear in the request URL in the **Mount Point** box. Do not use spaces in the name, and enclose the name with forward slashes, as in `/video_clips/`.
5. In the **Base Path** box, enter the full path to the directory that stores the content.
6. If content resides in an external data store such as a Network File System (NFS) or a storage area network (SAN), you may see a performance increase by selecting **Network** from the **Base Path Location** box.
7. Change **Cacheable by Caching Subscribers** to `No` if you are using this server as a content-caching subscriber, and you do not want this subscriber to search content publishers for this mount point. For more on this feature, see "Content Caching" on page 110.
8. Click **Apply**.
9. If you intend to serve content under this mount point through HTTP URLs, set up HTTP delivery as described in "Allowing HTTP Delivery" on page 68. This is generally not recommended for streaming clips, though, because HTTP delivery caches content in browsers, and allows viewers to save clips to disk.

   Note: Even if you do not allow direct HTTP requests for the content, Helix Server can deliver the content to media players over HTTP as necessary to work around firewall restrictions. Media players do not cache the content, however.

Using On-Demand Streaming with Other Features

This following table describes the ways in which on-demand streaming works with other features.

On-Demand Streaming Used with Other Features

Feature	Interaction with On-Demand Streaming
Simulated Live Broadcasting	You can use the Simulated Live tool (SLTA) to broadcast on-demand files as if they were live. See Chapter 10.
Live Archiving	If you have used the live archiving feature to convert live streams to on-demand files, you can then create links to these files and deliver them on demand. You can also use the archived files to recreate a live presentation using SLTA.

(Table Page 1 of 2)

On-Demand Streaming Used with Other Features

Feature	Interaction with On-Demand Streaming
Helix Proxy	Helix Server is configured to allow Helix Proxy to cache on-demand content streamed by Helix Server. To prevent certain on-demand clips from being cached, see "Preventing Streams from Being Cached or Split" on page 123.
Access Control, Authentication	Any access control or authentication rules you set up for Helix Server are automatically used when users request on-demand content with RealPlayer. QuickTime Player support is limited to basic user name/password authentication.
Monitoring	You can view which presentations are streaming at any time with the Server Monitor. Click the **Connections** tab or the **Files** tab to see which files are in use.
Logging	Presentations streamed from Helix Server, whether on-demand or live, can be recorded in a basic access log. QuickTime and Windows Media Player connections are included.

(Table Page 2 of 2)

Setting Up Aliases

Aliases are substitutions for actual file names and directory paths used in URLs. With an alias, you can mask resources or simplify published URLs for any links that use any protocol. Suppose that the following example is a path to a file located on the physical disk of a Helix Server:

```
C:\Program Files\Real\Helix Server\Content\music\pop\video1.rm
```

In this sample, `video1.rm` resides a few subdirectories below the `Content` directory. The following might be the unaliased URL for this file:

```
rtsp://helixserver.example.com/music/pop/video1.rm
```

Using aliasing, you can assign the content `music/pop/video1.rm` to any value, such as the following:

Content:	`music/pop/video1.rm`
Alias:	`example.rm`

Helix Server would then play the `video1.rm` clip when a viewer requested the following URL:

```
rtsp://helixserver.example.com/example.rm
```

Helix Server replaces only the alias portion of the URL. If you enter characters in the URL before or after the alias, that part of the URL remains unchanged. For example, consider the following changes in the preceding resource-to-alias value pair:

Content:	music/pop/
Alias:	example

For Helix Server to play video1.rm, you would use the following URL:

```
rtsp://helixserver.example.com/example/video1.rm
```

Here, Helix Server replaces the alias with the exact characters in the resource, leaving the file name intact. If the preceding URL did not contain the file name, the resolved alias would lead to a subdirectory, not a clip.

Preventing Alias Blind Spots

You need to be careful to avoid content blind spots—or content that cannot be accessed by a URL—by using aliases that match resources. Suppose that you had two files named video.rm and audio.rm in Helix Server's Content directory, and you created the following alias/resource pair:

Content:	audio.rm
Alias:	video.rm

The following URL would always play the audio.rm file:

```
rtsp://helixserver.example.com/video.rm
```

You would not be able to form a URL to play video.rm because Helix Server would always translate the characters video.rm into audio.rm. The only way to write a URL to play the actual video.rm clip would be to create an alias that uses the video.rm clip as its resource.

Tips for Defining Aliases

Observe the following considerations and limitations when setting up aliases:

- Aliases are for use only in URLs. Do not use an alias within a field in a Helix Administrator page. If you need to specify a mount point to enable a Helix Server feature, for example, use the actual mount point, not an alias.

- You can use an alias in an HTTP URL to media content that uses the `/ramgen/` or `/asxgen/` mount point. However, aliases do not function in HTTP URLs used by browsers to request Web pages.
- Do not use more than one alias in a URL.
- Do not create an alias that contains only numbers. You can use one or more numbers within an alias that also contains letters, however.
- Slashes in aliases have no special meaning. Hence, neither a forward slash nor a backslash indicate any type of directory structure.
- When it searches for an alias-to-resource match, Helix Server parses requests from left to right.
- Helix Server always selects the most complete alias match. In other words, Helix Server evaluates its entire list of aliases before selecting a match.

Creating an Alias

Creating an alias is a simple process that the following procedure describes.

➤ To map an alias to a resource

1. In Helix Administrator, click **Server Setup>URL Aliasing**.
2. Click the "+" icon to add a new alias.
3. Enter any descriptive text in the **Alias Description** box.
4. In the **Alias** box, enter the alias you want to map to a resource.
5. In the **Resource** box, enter the portion of the URL to which the alias corresponds.

Using Aliases with Other Features

The following table summarizes the Helix Server features affected by aliasing.

Aliasing Used with Other Features

Helix Server Feature	Interaction with Aliasing
Broadcast Redundancy	Not supported. URLs for broadcast redundancy must use actual resource names, not aliases.
Logging	An access log records actual resource names, not aliases.
Ram files	You can use aliases in Ram files.

(Table Page 1 of 2)

Aliasing Used with Other Features

Helix Server Feature	Interaction with Aliasing
Ramgen and ASXgen	You cannot combine the /ramgen/ or /asxgen/ mount point with an alias.
HTTP Protocol	Helix Server does not support aliased HTTP URLs. Aliases work only with the RTSP and MMS protocols.
Splitting: transmitters	Aliases are not supported. Transmitters always cite the actual name of the live source.
Splitting: receiver	You can use aliases in URLs
Multicast: backchannel	You can use aliases in URLs.
Helix Administrator	Aliases not supported.
Unicasting/On-demand streaming	You can use aliases in URLs.
Windows Media delivery	An alias is required for pull-splitting Windows Media content. Push-splitting does not require an alias.

(Table Page 2 of 2)

Browsing On-Demand Content

Helix Administrator's content browsing feature allows you to list all on-demand clips residing on Helix Server. You specify the mount points and types of clips, such as all clips or only the RealVideo clips, that you want to see. Helix Administrator then generates an HTML page listing and linking to all clips that fit those criteria.

➤ To create a content browsing list:

1. Click **Content Management>Content Browsing** in Helix Administrator.
2. The **Browsable Mount Points** box is predefined with the primary on-demand mount point (/), which corresponds to the Helix Server `Content` directory and its subdirectories.
3. If you've created other mount points as described in "Adding a Mount Point for On-Demand Clips" on page 95, you can add them to the **Browsable Mount Points** box by selecting them through the pull-down list.
4. The default value * in the **Extensions to Browse** box generates a content list for all clips and files, including SMIL files and metafiles. To limit the types of files shown, enter the file extensions in the box. Separate multiple extensions with a comma. For example, to browse just MP3 and RealMedia clips, you enter this:

```
mp3, rm
```

5. Click **Apply** to save the changes.
6. Click the **Browse Content** link to generate a separate HTML page of all the selected content. The list includes content for all mount points shown in the **Mount Points to Browse** Box, not just the highlighted mount point. The list includes mount points, directories, and files, with file sizes listed in bytes. Clicking a link in the **Play** column plays a clip.

Displaying Source Information

Just as browsers have "view source" commands that display HTML markup, the desktop RealPlayer has a view source command (**View>Clip>Clip Source**) that displays SMIL markup and clip information for all content it plays. When a viewer gives this command, Helix Server generates an HTML page that contains the SMIL markup (if used), as well as information about clips and live broadcast streams. RealPlayer displays this information in a native browser window. With RealPlayer 7 and 8, source information displays in the viewer's default Web browser.

> **Tip:** The view source command is a handy way to learn a clip's streaming bit rate. Just play the clip in RealPlayer, and use the view source feature to display the clip's encoding statistics. SureStream RealAudio and RealVideo clips display all available bit rate streams.

Default Security Precautions

By default, source information is not divulged for any clip or presentation that requires user name and password validation. For other presentations, the source information hides clip locations. For example, when the viewer requests the source of a SMIL file, the browser's address box displays a URL that shows random numbers and letters in place of the SMIL file path:

```
http://helixserver.example.com:80/viewsource/template.html?ABcdlkj293847
```

The source page displaying SMIL markup links to separate pages that give information about each clip used in the presentation, including clip size, buffer time, and bit rate, but not the full path. To protect the location of content, the SMIL source page omits the full path to the clips, showing

ellipses (...) instead. For example, the actual SMIL file on Helix Server may have a source tag that gives the full URL to a video clip:

```
<video src="rtsp://helixserver.example.com/presentation/presentation.rm" .../>
```

In the SMIL source page that viewers see, the full path is hidden:

```
<video src="rtsp://.../presentation.rm" .../>
```

When you stream with Helix Server, you have full control over what information you display for which clips. You can override defaults, for example, and display path information for all clips, or only for those in selected directories. Other methods of playing clips do not offer this full control, however:

- When viewers play clips residing on their local computers, RealPlayer generates the source pages itself, and always shows the full paths and all clip information.
- If a Web server delivers clips, RealPlayer generates the source pages itself, hiding the paths to media clips, but showing all other clip information.

Selectively Displaying Source Information

By default, the view source feature is turned on for all on-demand clips and broadcasts, except those that require user name and password validation. The full paths to clips and files are not displayed, however. On a system-wide basis, you can turn off the view source feature, or decide to show full path information for all clips that do not require password validation. Or, you can set up selective rules to define exactly which on-demand clips and broadcasts display source information.

To enable selective rules, you define paths to content directories and broadcast mount points. The main on-demand mount point (/) is predefined to let you set up a source rule for the entire `Content` directory. You can add more rules for `Content` subdirectories, other on-demand mount points and subdirectories, and broadcast mount points such as `/broadcast/`. In applying view source rules,

Helix Server looks for the closest match. The examples in the following table illustrate how these rules work.

Examples of View Source Rules for the Main Content Directory

Path	View?	Result
/	No	Only clips in the Content/news/ subdirectory (and any subdirectories under that) show source information.
/news/	Yes	
/	No	Only clips in the Content/news/daily subdirectory (and any subdirectories under that) show source information.
/news/	No	
/news/daily/	Yes	
/	Yes	All clips in the Content directory except those in the news/ subdirectory show source information. However, clips in the news/daily/ subdirectory (and any subdirectories under that) show source information.
/news/	No	
/news/daily/	Yes	

For More Information: See Chapter 7 for information about broadcast mount points.

Changing View Source Settings

The following procedure explains how to change the view source settings. You need to change these settings only if you want to modify the defaults to turn off view source entirely, for example, or to set up selective viewing rules.

➤ To change view source settings:

1. In Helix Administrator, click **Content Management>View Source**.
2. Set the appropriate features in the pull-down lists of the **Master Settings** section, as shown in the following table. These settings let you temporarily or permanently establish system-wide viewing rules.

View Source Master Settings

Menu	Option	Action
View Source	Use Settings Above	Apply selective view source rules.
	Disable View Source	Show no source information system-wide.
	Enable View Source	Show source information system-wide.
Hide Paths	Use Settings Above	Apply selective path rules.
	Show All Paths	Show paths for viewable clips.
	Hide All Paths	Hide paths for viewable clips.

If you want to enable selective viewing rules, select `Use Settings Above` as the master view source setting. You can use the master setting `Hide All Paths` or `Show All Paths` with selective rules. If you want to hide paths on a case-by-case basis, though, select `Use Settings Above`.

3. Perform this step if you want to define viewing rules, as described in "Selectively Displaying Source Information" on page 102. The **View Paths** box is predefined with the primary on-demand mount point (/), which corresponds to the `Content` directory in the main Helix Server installation directory. Do the following to define a new path:
   a. Click the "+" icon to create a path listing.
   b. Change the name in the **Edit Path** box to the path that corresponds to the view source rule. If you want to set up a source rule for clips in a `Content/news/` subdirectory you've created, for example, you enter `/news/`. (The mount point "/" corresponds to the `Content` directory.)
4. In the **View Source** and **Hide Paths** pull-down lists, select the appropriate settings for the highlighted path. If you set **View Source** to `No`, the **Hide Paths** setting does not matter.
5. Click **Apply** to save your changes.

View Source Used with Other Features

The following table describes how the view source feature interacts with other Helix Server features.

View Source Used with Other Helix Server Features

Feature	View Source Interaction
Broadcasting	The view source feature applies to both on-demand and live content.
SLTA	On-demand files that are converted to live streams through SLTA show the same information as other live files.
Splitting	View source is disabled for users who are receiving a broadcast through a push splitting source.
Authentication	The view source feature is automatically disabled for all secure content.
Reporting	A record is created in the basic access log when a user makes a view source request. See the table "GET Statements for On-Demand Content" on page 338.

CHAPTER

6

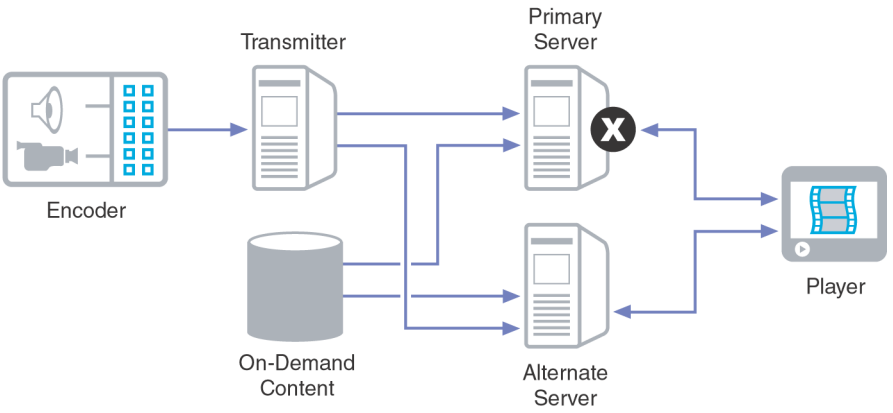
MULTIPLE SERVERS

There are many features that help you to manage large networks containing more than one Helix Server. This chapter covers content caching, server redundancy, and proxies.

Implementing Redundant Servers

Normally, if an RTSP connection between RealPlayer and Helix Server breaks, RealPlayer attempts to reconnect to the same server. With a redundant server setup, Helix Server sends RealPlayer a list of alternate servers when it sets up the RTSP control channel. If a break occurs, RealPlayer uses its alternate list to connect to another Helix Server that has the same content. If multiple alternates are available, RealPlayer selects one randomly. The following illustration depicts a RealPlayer failing over to an alternate Helix Server.

Redundant Servers



Note: The redundant server feature works only with RealOne Player and later. It does not function with earlier versions of RealPlayer, or any other media players.

Redundant Server Requirements

There are three criteria for setting up redundant servers. You need to ensure that the same content is available on each server, and that each server defines a list of redundant servers, along with the rules for when to use them.

Cloned Content

Generally, all servers in a redundant cluster must offer the same content, whether on-demand or live. If a redundant server offers only some of the primary server's content, you'll need to set up rules on the primary server to define exactly which failures prompt RealPlayer to reconnect to the alternate server.

> **For More Information:** You can use the content caching feature, described in "Content Caching" on page 110, to distribute on-demand content between servers.

Alternate Server Lists

Each Helix Server defines a list of servers available for redundant failover. Typically, each Helix Server in a redundant cluster identifies the other servers in the cluster. Helix Server "A" fails over to Helix Server "B" and Helix Server "C," for example, while Helix Server "B" fails over to servers "A" and "C". If connections to Helix Server "A" fail, each disconnected RealPlayer chooses server "B" or "C" at random. This helps to balance the load if a heavily-used server becomes completely unavailable.

It is not necessary for each server to fail over to every other server in the cluster, however. Helix Server "A" might fail over to Helix Server "B", which fails over to Helix Server "C", which fails over to Helix Server "A". Because you define each server's failover list separately, you have a lot of flexibility over how to set up your redundant cluster.

Failover Rules

On each Helix Server in a redundant cluster, you create rules to designate which alternate servers are used when a content stream fails. Because rules are attached to mount points, you can have URLs that use different mount points fail over to different servers, or not fail over at all.

Helix Server searches for and applies rules in an assigned order. For this reason, you should put longer, more specific rules first. Generally, the shorter the rule, the broader its scope. For example, assigning the root mount point

(/) to an alternate would result in *all* disconnects for on-demand content being redirected to that alternate.

The rule system also allows you to forbid the use of the redundant servers feature on a path-by-path basis. When you exclude a path, Helix Server does not pass a list of alternate servers to RealPlayer while setting up the RTSP control channel. If RealPlayer experiences a disconnect when playing that content, it tries to reconnect to the same Helix Server.

Setting Up Redundant Servers

The following procedure explains how to set up a redundant server list. Carry out this procedure on each server in your redundant cluster.

➤ To set up an alternate server:

1. In Helix Administrator, click **Server Setup>Redundant Servers**.
2. In the **Alternate Servers** box, click the "+" icon to add an alternate server. Then enter the following information:
   a. Enter any text that describes the alternate server in the **Description** box.
   b. Enter the alternate server's RTSP port in the **Port** box.
   c. In the **Host Name** box, enter the host name (preferred), IPv4 address, or IPv6 address of the alternate server.
   d. Repeat this step for all redundant servers.
3. In the **Redirect Rules** box, click the "+" icon to set up a redirect rule:
   a. Change the generic path name in the **Edit Rule Path** box to an actual path or mount point on your server. To use an alternate server as a failover server for all content, for example, enter the main content mount point:

      /

   b. Below the **Alternate Servers** box, choose from the pull-down list the servers that will act as failover servers for content served under this rule. The menu lists the servers already defined in the **Alternate Servers** section.
   c. Repeat this procedure to define other redirect rules as necessary.
4. To exclude a path from any rule, click the "+" icon in the **Exclude Paths** section, and enter the path in the **Edit Path** box. For example:

```
/archive/
```

5. Click **Apply**.

Content Caching

Content caching enables you to transfer on-demand streaming media content in any format between two or more Helix Servers. This dynamic distribution improves playback quality by propagating the content closer to the viewer. It also reduces delivery cost by caching content at the network's edge. Distributing content is beneficial if you have large numbers of centrally located, on-demand media files, as well as a number of Helix Servers distributed across your network.

> **Note:** Caching on servers and proxies works basically the same, but the usage is different. A proxy is used to get through a firewall. A server caches only static content, and works as described in this section. Caching works for all data types. However, if you cache a SMIL file, the content caching feature does not parse the file and cache the clips used in the SMIL presentation.

Understanding Content Caching

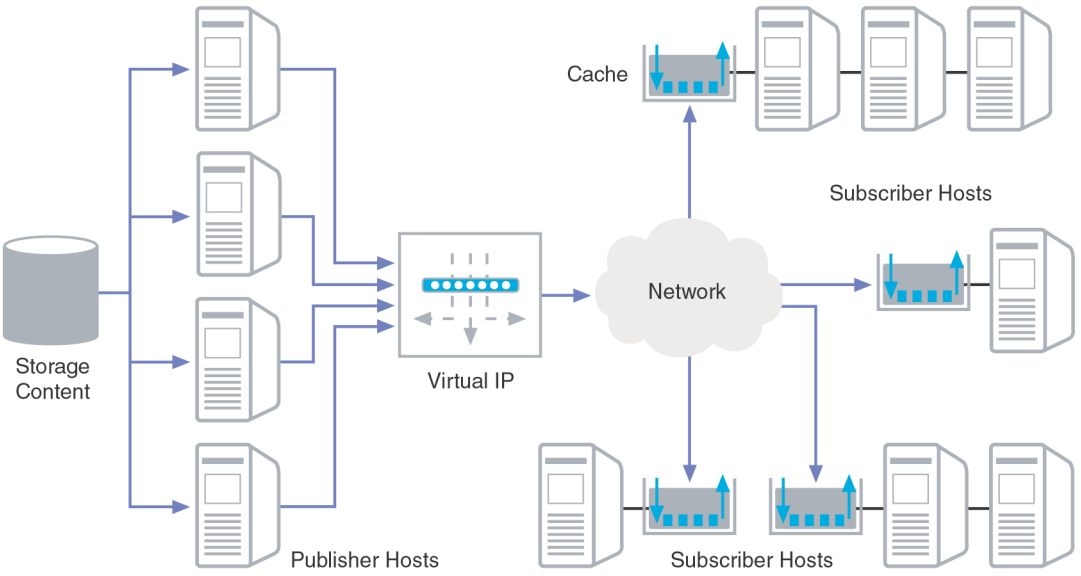
The following sections provide an overview of the content caching feature.

Publishers and Subscribers

In a caching network, you typically manage the bulk of your on-demand media from one or more Helix Servers configured as *caching publishers.* Other Helix Servers, called *caching subscribers*, contact the publisher for on-demand content. Each subscriber maintains a portion of physical disk space for caching on-demand content originally deployed on the publisher.

When a media player requests on-demand content from a subscriber, the subscriber attempts to fulfill the request from its cache. If the content is not cached, the subscriber forwards the request to the publisher. As the publisher fulfills the request, the subscriber loads the content into its cache, and streams it to the media player. The following illustration shows a cluster of publishers behind a load balancing solution, delivering content to subscribers in different locations.

Content Caching



Load Balancing

RealNetworks recommends that you implement a load balancing solution for Helix Servers acting as publishers. The minimum requirement is *DNS rotation*, sometimes known as *round robin DNS*. This technique involves associating multiple IP addresses to a single host name in the DNS record of the host. When the DNS server receives a request for the host name, it rotates through the IP addresses. Thus, the load for the host name is distributed between the IP addresses. Load balancing solutions often use a virtual IP address scheme.

> **For More Information:** For recommendations about placing a cluster of Helix Servers behind a virtual IP address, see "Working With A Virtual IP Address" on page 254.

Content Loading and Removal

Once publishers and subscribers are set up, content caching requires little administration. Initial setup can include preloading the cache. Subscribers then prune their caches with a "least recently used" (LRU) mechanism, which removes first those files that have not been requested for the longest time when the cache fills. This way, popular content remains readily available. Removing content from the publisher stops the content from being streamed

throughout the network. A subscriber streams only the content that resides on the publisher, even if the content is still available in the subscriber's cache.

Security

Only trusted subscriber Helix Servers may cache content from the publisher. During installation, Helix Server sets up a content caching realm used on the publisher. The person who administers the publisher adds a unique user name and password to the content caching realm, and distributes these credentials to any subscribers set up to access the publisher. In addition to maintaining a secure channel between publisher and subscriber, the administrator of the publisher retains some control over the files served by the subscriber, even if those files already exist in the subscriber's cache.

> **For More Information:** For information on the search logic between subscriber and publisher, see "Search Logic for Content Caching Rules" on page 115.

Bandwidth Conservation

Content caching ensures that only those portions of media files needed to fulfill a request are transferred from the publisher to the subscriber. In the case of SureStream media requests, only the byte ranges corresponding to the client-subscribed bit rates are transferred. In other words, when a client requests a SureStream clip from the subscriber, the subscriber determines the client bandwidth and requests from the publisher only the portion of the clip that best matches the request. This ensures that the connectivity between the content subscriber and content publisher is used efficiently, and that only requested data resides in the cache. If another client requests content that a subscriber has just cached, but at a different rate, the subscriber adds the new bit rate data to the file.

> **Note:** If the subscriber administrator preloads a SureStream clip, however, the entire file is loaded in the cache.

Links to Cached Content

Links to cached content use the same structure as links to regular on-demand content, with the following exceptions:

- The subdirectory structure in the link pointing to the *subscriber* must match the directory structure on the *publisher*

- A rule that defines the publisher's subdirectory structure must appear on the subscriber.

  For More Information: For more information, as well as specific examples of links, see the section above, "Defining Subscriber Rules" on page 114.

Setting up Content Caching Publishers

There is little to set up on a Helix Server content caching publisher. You must set up authentication. You must forward the user names and passwords of the `SecureCDist` realm to the subscriber administrator, and indicate the publisher's RTSP port and content mount points. This is so subscribers can create rules for publishers to use when caching content.

For More Information: For more on rules, see "Defining Subscriber Rules" on page 114.

Setting up Authentication on the Publisher

During installation, Helix Server sets up a default content caching realm, creates a database of users, and populates the database with a default user name and password, which is the same user name and password used for initial access to Helix Administrator. You need to distribute a user name and password for the `SecureCDist` realm to the administrators of subscriber Helix Servers for use when setting up subscriber Helix Servers.

For More Information: See "Setting Up Realms" on page 284 for information about authentication realms.

Setting up Content Caching Subscribers

You set up subscriber Helix Servers after you set up the publisher. Only two steps are required:

1. Define the publisher Helix Servers associated with this subscriber, as well as a publishing hierarchy. For more information, see "Identifying the Publisher" on page 114.
2. Define rules for how subscribers access publishers based on the URL used by the client requesting content. For more information, see "Defining Subscriber Rules" on page 114.

Note: As an option, you can also preload the subscriber cache with files from the publisher. See "Manually Populating a Cache" on page 117.

Designating Content for Content Caching

Designate content for caching by setting the content caching mount point flag. Content located under this mount point is then available to the subscriber. The subscriber then enters this mount into a subscription rule.

➤ To set a content caching mount point flag:

1. Click **Server Setup>Mount Points**.
2. Select the mount point where content available for content caching subscribers resides.
3. Select **Yes** from the **Cacheable by Caching Subscribers** box.

Identifying the Publisher

Each subscriber Helix Server must have at least one publisher Helix Server.

➤ To define the publisher:

1. Click **Content Management>Content Caching**.
2. Ensure that `Yes` appears for **Enable Content Fetching**. This enables the content caching feature.
3. Click the add new publisher button. A generic publisher description appears in the **Publisher Description** box.
4. In the **Host** box, enter the host name, IPv4 address, or IPv6 address, for this publisher.
5. In the **Port** box, enter the RTSP port of the publisher. (Most Helix Servers use `554`).
6. In the **User Name** and **Password** boxes, enter the user name and password supplied by the administrator of the content caching publisher.
7. Click **Apply**. This adds a publisher node to the subscriber.

Defining Subscriber Rules

Content subscriber rules afford you more control over how content is distributed along your network. You can think of each rule as a subdirectory

structure on the publisher that appears in request URLs. For example, consider the following URL:

```
rtsp://subscriber.example.com/subdirectory/myfile.rm
```

In this example, `/subdirectory/` is meant to be a subdirectory on the publisher that you add to the subscriber as a rule. When an incoming URL arrives at the subscriber, Helix Server attempts to match whatever string it finds immediately following the host name.

Search Logic for Content Caching Rules

1. Search for the string in the local on-demand content directory (non-content caching) and its subdirectory structure. If a match is found, serve the file. If not, continue the search.
2. Search for the string in the content caching cache. If a match is found, serve the file. If not, continue the search.
3. Search for the string in the rules associated with publisher nodes from the top down. When a match is found, the string is handed off to the publisher. If no match is made, the request fails.
4. If the publisher finds the file, it serves the file to the subscriber's cache. The subscriber then serves the file to the media player from its cache. If the publisher does not find a match, the request results in an error.

In practice, however, if Helix Server would have found a matching file in its cache, it would have immediately checked the publisher to ensure the file was still available. If the file had been removed from the publisher, the subscriber would not serve the file to the player, even though the file resides in cache. In this way, the administrator of the publisher has some control over the content available to subscribers.

For any of this to work, the string that appears in the URL must match a rule on the subscriber. That rule, in turn, must match the subdirectory structure on the publisher. Thus, the administrator on the subscriber must carefully coordinate with the administrator on the publisher. You can use the asterisk (*) as a wildcard character in rules. In fact, Helix Server adds an implicit wildcard to the end of each rule it processes.

Perhaps the easiest method of setting up content caching is not to use any subdirectory structure at all. Just use the root of the publisher Helix Server for storage all on-demand content, and then create a "/" rule on the subscriber. Consider the following example, meant to be typed directly into RealPlayer:

rtsp://cdist_subscriber.helixserver.com:554/**myfile.rm**

This URL combined with a content caching rule for "/" allows for the best of both worlds: *all* incoming URLs to the subscriber are matched against local content, and then run through the content caching regimen.

➤ To define content subscriber rules:

1. Click **Content Management> Content Caching**.
2. Ensure that Yes appears for **Enable Content Fetching**. This enables the content caching feature.
3. Ensure that at least one publisher is set up. If the correct publisher does not appear in the list, go to the procedure above, "Identifying the Publisher" on page 114, before continuing with this procedure.
4. Click the add new rule button. A generic rule name appears in the **Publisher Description** box. You may edit this name to whatever you like.
5. In the **Rule Path** box, enter the mount points flagged as **Cacheable by Caching Subscribers** on this subscriber. If there is an additional subdirectory structure under the mount point that will appear in your links, then add that structure to the rule, too.
6. Ensure that Yes is selected in the **Enable Rule** box.
7. Use the **Add Publisher to This Rule** pull-down list to add publishers to this rule. Repeat this step to add additional publishers. Make sure rules are valid for all publishers. Helix Server searches for matches on publishers from top to bottom. If a publisher connection times out, Helix Server attempts to connect to the next publisher in this list.
8. Click **Apply**.

Defining the Size of the Cache

The subscriber must have a cache to store files received from the publisher. By default, Helix Server comes configured with a cache of 1000 megabytes. The size of the cache must be at least 11 megabytes. There is no maximum size restriction.

The cache is automatically pruned based on an least recently used (LRU) mechanism. This means that infrequently requested content is first discarded when the cache reaches its maximum size. If you restart Helix Server, the LRU index is rebuilt based the contents of the cache at the time of restart.

Although there is no functionality to prune the cache with Helix Administrator, there are still ways of manipulating cache content. As has been discussed earlier, if you remove content flagged for content caching from the publisher, the subscriber will not serve that content to players, even if the content remains in cache. Eventually the content will be pruned by the LRU mechanism.

A more direct approach would be to simply delete files directly from the physical disk that holds the cache. Never do this while Helix Server is running, however. Helix Server stores cache contents in memory. Thus, deleting files while Helix Server is running will create a consistency problem. Shut down Helix Server and then delete.

➤ To define the size of the subscriber Helix Server's cache:

1. Click **Content Management> Content Caching**.
2. In the **Maximum Cache Size** box, change the default value of 1000 megabytes to the size required. The cache has a minimum size limit of 11 megabytes. If a lower number is entered, Helix Server ignores the value and uses 11 megabytes of disk space. The theoretical maximum limit for cache size is about nine quadrillion megabytes, but cache sizes in excess of ninety-nine million megabytes cannot be set in Helix Administrator.

Defining the Cache Location

The physical location of the subscriber is the value entered as the base path of the `/fsforcache/` mount point. You can change the base path of this mount point, but you should not change the name of the mount point itself.

➤ To define the location of the subscriber Helix Server's cache:

1. Click **Server Setup>Mount Points**.
2. In the list on the left, select `"RNCache Local File System"`.
3. To change the base path, type the new location of the cache in the **Base Path** box.
4. Click **Apply**.

Manually Populating a Cache

Administrators of subscriber Helix Servers can manually make requests for content. This populates the cache with whatever content the administrator

requires. One reason to preload content on the subscriber is so that when clients first start to request files, the publisher is not inundated with requests that all must be fulfilled over the content caching network.

When files are preloaded using the **Fetch These Clips** box described below, Helix Server requests the entire file. In other words, Helix Server requests all bit-rates for pre-loaded SureStream files. Once the files are pre-loaded into the cache, Helix Server treats it like any other file in the cache, pruning based on an LRU mechanism.

> **Note:** In addition to preloading the cache from the network, you can also create a disk image of the content you want in cache, and then use that image to propagate cache contents from subscriber to subscriber. You should shut down Helix Server beforehand, however. For more information about where Helix Server stores the physical cache on disk, see "Defining the Cache Location" on page 117.

➤ To populate the subscriber cache manually:

1. Click **Content Management> Content Caching**.
2. Click **Populate Cache**.
3. In the **Fetch These Clips** box, enter the file names of the clips that you want loaded into the cache. Helix Server uses the rules you have in place to fetch content. Hence, use a subdirectory structure that reflects rules you have already defined for publishers.

   For example, if `publisher1` has static content in its `/news/` directory that you want propagated to subscribers, you would add a `/news/` rule to `publisher1`. To fetch content from `publisher1`, you would enter `/news/`*`filename`* in the **Fetch These Clips** box (where *filename* substitute the actual names of the files for pre-loading on the subscriber).

   > **Note:** To manually populate a subscriber's cache, content caching subscriber must be completely configured, including having a least one publisher, one rule, and the **Enable Content Fetching** turned on. For more information, see "Setting up Content Caching Subscribers" on page 113.

Content Caching Used With Other Features

This section describes the ways in which content caching works with other features. As a general rule, content caching is comparable to on-demand streaming in how it works with other features.

Content Caching Used with Other Features

Other Feature	Notes
Live Delivery Methods: Unicasting, Simulated Live Broadcasting, Splitting, and Multicasting	Content caching and live broadcasting are mutually exclusive methods of delivering content
Helix Proxy No Cache Directives	Cache directives are not honored for content caching. These are applicable only for proxy and intermediary devices.
Access Control and Individual User Authentication	Content is subjected to any conditional access rules that have been established on the subscriber Helix Server.

Using a Media Proxy

You can use a proxy to cache or split all on-demand and live content hosted on Helix Server. Using proxies optimizes bandwidth by rebroadcasting content on behalf of clients. For most content, this type of optimization poses no problem. However, there may be some broadcasts or on-demand presentations that you do not want cached or split by a proxy. This section discusses how to define directives for on-demand content and live broadcasts that prohibit proxy devices from caching or splitting streaming media.

> **For More Information:** See *Helix Proxy Administration Guide* for information on using Helix Proxy.

Understanding Media Proxies

A media proxy is generally installed on an intranet or on an Internet Service Provider's (ISP) network. Media players, such as RealPlayer, are configured to use a proxy for all streaming media requests. When a person requests streaming media, the media proxy acts as an intermediary agent, making the request on behalf of the client. The proxy fulfills the request by locally caching

on-demand media, or splitting a live broadcast. In this manner, the proxy conserves bandwidth between it and Helix Server.

RealNetworks certifies third-party media proxies with Helix Proxy technology. This certification ensures two things:

- media cached locally on the proxy is secure
- proxy-fulfilled media requests are always accounted for at the origin Helix Server.

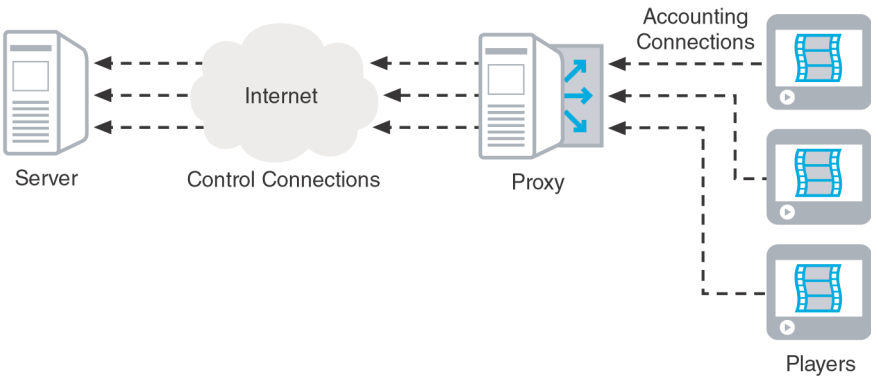
Reports for proxy-fulfilled media requests can be accessed in real-time or through the basic access log. Make sure to choose a RealNetworks certified proxy for your network.

How Servers and Proxies Work Together

To use Helix Proxy, all media players on the network must be configured to request media through the proxy. When the player makes the initial request for media, a two-way TCP control channel is established. The control channel is for the player software and Helix Server to send information to one another, like rewind and pause commands or user names and passwords. Helix Proxy forwards this initial request to the Helix Server on behalf of the player. Helix Server verifies the existence of the requested file, and whether the client is authorized to view it.

Meanwhile, Helix Proxy monitors these control connections, so that it can account for all clients requesting content from the Helix Server. For Helix Proxy, the control channel is an *accounting channel*. It's the same TCP connection used by the player and server, but Helix Proxy uses it in a different way. The following diagram illustrates multiple players connecting to a Helix Server through Helix Proxy.

Establishing the Accounting Connection



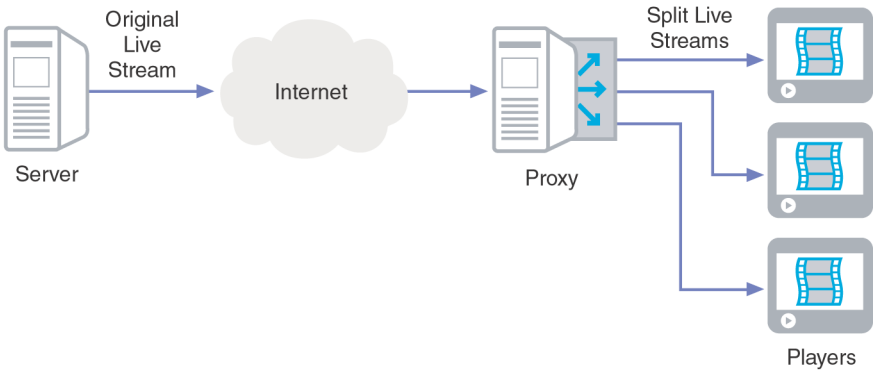
How Helix Proxy Streams Content

With each player connected to Helix Server in this manner, Helix Proxy can begin streaming media. Helix Proxy streams media based on the type of content being requested, and whether the Helix Server administrator has placed any cache directives on content.

Helix Proxy Replicating Live Content

If the stream is live, Helix Proxy uses pull splitting to get an initial stream from Helix Server. This stream is used to fulfill the initial RealPlayer request, as well as any additional requests. The source Helix Server sends only a single stream to Helix Proxy. The following diagram illustrates multiple live streams replicated by Helix Proxy.

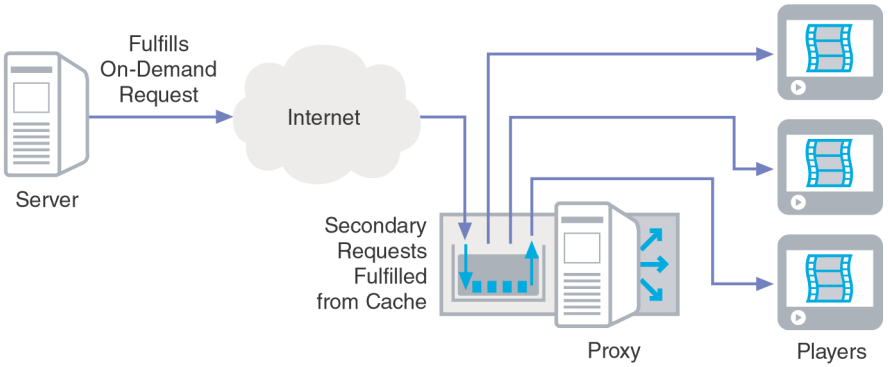
Helix Proxy Replicating Live Streams



Helix Proxy Delivering On-Demand Content

If the stream is on-demand, Helix Proxy first tries to fill the request from its media cache. If the content is not yet stored in the cache, Helix Proxy pulls the content from the source Helix Server, simultaneously serving the client and filling the cache. The following diagram illustrates requests for multiple on-demand streams fulfilled by Helix Proxy.

Helix Proxy Streaming On-Demand Content from Cache



Cache Directives Restricting Helix Proxy

For live or on-demand content, you have full control over what content is available to Helix Proxy. If you have cache directives in place, Helix Proxy functions differently than described above, depending on the type of content being restricted.

Cache Directives and On-Demand Content

If cache directives restrict on-demand content, Helix Proxy does not cache content, nor does it fulfill requests for restricted media from its cache. Because the origin Helix Server fulfills requests for restricted content, there is no bandwidth optimization.

Cache Directives and Live Broadcasts

If cache directives restrict a mount point for a live broadcast, Helix Proxy does not rebroadcast a single stream pulled from the origin Helix Server. Instead, it pulls a stream to fulfill each client request. Because the origin Helix Server fulfills requests for restricted content, no bandwidth optimization occurs.

Restricting Proxies from Caching or Splitting Content

Unless you specify otherwise, all material on your Helix Server is available to Helix Proxy. Helix Server has these options for restricting which Helix Proxies can cache streams:

- preventing some paths, files, or mount points from being cached
- preventing certain Helix Proxies from accessing your Helix Server
- preventing all caching

Preventing Streams from Being Cached or Split

You can restrict paths, files, or mount points that Helix Proxies can utilize. If Helix Server receives a request for material included in the **Deny Cache Requests for these Resource Paths** list, it streams the file directly to the client rather than allowing it to be cached and retransmitted. Helix Server records the transaction in the basic access log, and reports a bytes sent size of 0 bytes in the cached requests log file.

For example, you might choose to prevent material in authenticated content locations from being cached. Or, you might put the path to time-sensitive clips on this list so that it cannot be stored by Helix Proxy.

> **Note:** Media caching software makes more streams available on your Helix Server. If you limit clips to be cached, you also limit how many clients you can serve.

➤ To prevent Helix Proxy from caching material on Helix Server:

1. In Helix Administrator, click **Server Setup>Cache Directives**.
2. Under the **Deny Cache Requests for these Resource Paths** list, click the "+" icon. A generic path name appears in the list.
3. In the **Edit Paths** box, type the name of the path, file, or mount point for which you want to restrict content.
   - For example, if a subdirectory of the `Content` directory contained a directory named `news`, you would add `/news` to the **Deny Cache Requests for these Resource Paths** list. If you only wanted to prevent the late-breaking news clip from being cached, you would add the path and the file name instead (`/news/breaking.rm`).
   - If you wanted to restrict live streams from being rebroadcast, you would add the mount point to be restricted.

4. Click **Apply**.

Preventing Helix Proxy from Accessing Helix Server

You can indicate that certain Helix Proxies are not allowed to cache any of your material. To do this, you must know the IP address of the machine on which Helix Proxy is installed.

> **Tip:** To find out which incoming requests are coming from a Helix Proxy, look in the `rmaccess.log` file. You can identify Helix Proxy request with `the client_id field` of the `rmaccess.log file`.

➤ To prevent certain Helix Proxies from making requests:

Create an access rule for the Helix Proxy you want to restrict. In addition to specifying the IP address, indicate the port number to which access should be denied (usually 554).

> **Note:** To learn about limiting access to your Helix Server according to the IP address of any other computer, see "Understanding Access Control" on page 263.

Preventing All Caching

➤ To prevent all caching of all material from all clients and Helix Proxies:

1. In Helix Administrator, click **Server Setup>Cache Directives**.
2. In the **Deny All Cache Requests** box, select **Disabled**.
3. Click **Apply**.

Cache Control Used with Other Features

This section describes how Helix Server interacts with Helix Proxy and other media proxy software.

How Helix Server Features Work with a Media Proxy

Helix Server Feature	Notes
On-Demand Streaming	You can mark on-demand content as non-cacheable, on a per-file or per-folder basis. Otherwise, all on-demand clips are automatically available to media caching software.
Live Broadcasts	You can create a directive for mount points, so that live broadcasts are not split by media caching software.
Access Control	You cannot restrict the IP addresses of an individual client computer. However, you can restrict the IP address of the Helix Proxy that is requesting material on behalf of clients. See "Preventing Helix Proxy from Accessing Helix Server" on page 124.
Authentication	Before allowing clips to be cached, Helix Server verifies whether the client is valid. If the requested material is marked as secured, it then performs any necessary authentication checks. Authenticated material can be stored in a Helix Proxy cache, but the client will be authenticated with the source Helix Server every time it tries to access the stored clip.
ISP Hosting	All on-demand material served on behalf of ISP-hosted customers can be cached, unless you mark those directories as non-cacheable (see "Preventing Streams from Being Cached or Split" on page 123).
Monitoring	The Java Monitor will show the IP address of the caching software as it plays a clip. The caching software is not identified as such. Rather, it appears to be a client.
Reporting	To find out which incoming requests are coming from a Helix Proxy, look in the `rmaccess.log` file. You can identify Helix Proxy request with the `client_id field` of the `rmaccess.log` file.
Ad Streaming	All material served through the ad streaming feature is marked as non-cacheable. There is nothing you can do to prevent this.

PART

IV

BROADCASTING

This section explains how to broadcast events live. Once you understand the basics of live broadcasting, you can learn how to distribute your broadcast as widely as possible, using as little bandwidth as possible.

CHAPTER 7

UNICASTS

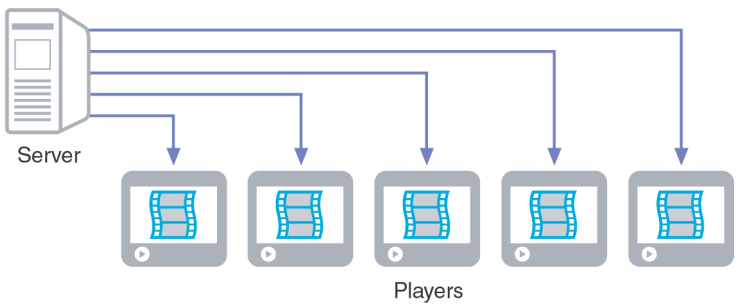
A unicast is the simplest way to broadcast a live event to viewers. Helix Server can unicast events in RealMedia, Windows Media, and QuickTime formats, as well as in a number of RTP-based formats such as MPEG. You can use live unicasting for audio and video feeds delivered on the Internet or private intranets.

> **Tip:** See also Chapter 10 for information about delivering a prerecorded clip as if it were a live event. This is a good way to test broadcasting before streaming a live event.

Understanding Unicasts

Basic broadcasts are called *unicasts* because Helix Server delivers a separate stream to each media player. An encoder such as RealProducer encodes a real-time event in a streaming format. The encoder delivers the stream to Helix Server, which then delivers the live stream to each media player, as shown in the following illustration. Viewers typically receive the broadcast by clicking a link in a Web page, just as they do for an on-demand clip. The link's format, though, instructs Helix Server to deliver a live stream rather than a prerecorded clip.

Unicasting



Licensing Limitations

Helix Server can deliver multiple, simultaneous broadcasts in any combination of supported media formats, as long as the total number of players does not exceed the licensed maximum. On a Helix Server licensed for 100 Megabits of streaming data, for example, you can unicast the same live presentation in RealMedia format at 40 Megabits and in Windows Media format at 40 Megabits. At the same time, you might unicast 20 Megabits of an entirely different live presentation to MPEG players.

Bandwidth Constraints

In unicasting, each broadcast stream uses bandwidth, so you are also limited by Helix Server's available, outgoing bandwidth. Unicasting from a single Helix Server is generally suited, therefore, for light- to medium-volume broadcasts. For events with a large number of viewers, you can use splitting (see Chapter 9), multicasting (see Chapter 8), or a combination of the two, to deliver a large number of broadcast streams, or to conserve bandwidth.

> **Tip:** If you broadcast to RealNetworks media players on a multicast-enabled intranet, set up back-channel multicasting as described in Chapter 8. Players can then connect to a unicast in multicast mode, which saves on network bandwidth. If a player cannot receive the multicast, it requests a unicast.

Broadcast Trial Runs

When you broadcast live content, you don't get a second chance. It's good practice to perform a trial run to ensure that the equipment works properly and that the broadcast results are what you expect. Because you can't edit a live broadcast the way you can a prerecorded file, it's important to set your audio levels and plan your video shots carefully in advance.

During both the trial run and the live broadcast, view the broadcast output with the appropriate media player. When the player connects, check that the buffering time does not exceed 15 seconds. Throughout the presentation, keep an eye on the broadcast quality. If you experience problems during your trial run, you may need to reduce the number of streams if you are using a multiple-stream technology such as SureStream. Or, you may need to run the encoder on a more powerful computer.

Live Unicasting Used with Other Features

Live unicasting works with all other Helix Server features. There are some considerations for each feature, however, as described in the following sections.

Firewalls

Live broadcasts use standard streaming protocols, so highly restrictive firewalls may block broadcasts. Firewalls are described in Chapter 11.

Access Control and Authentication

Before any client is allowed to receive any broadcast, Helix Server checks the client's IP address to see whether the client is allowed to receive a broadcast. If the address is acceptable, Helix Server looks at the location of the file to see if it is in a secure location. If so, Helix Server challenges the user (or media player) for identification. Once the client passes the tests, Helix Server connects the client to the live broadcast.

> **For More Information:** See Chapter 13 and the section "Securing Broadcasts" on page 277 for more information on setting up authentication for live broadcasts.

Monitoring

You can view all streams and connections to broadcasts through the Server Monitor in Helix Administrator. Chapter 17 has more information.

Logging

The basic access log records all client connections to live broadcasts. The basic error log records any errors encountered by clients. See Chapter 15 for details.

Archiving Broadcasts

Helix Server is preconfigured to archive broadcasts that use the RealMedia or MP3 format. It does not archive other broadcast formats, such as MPEG-4, Windows Media, and QuickTime. Archived files function just like prerecorded clips, and you can stream them on demand immediately after they are recorded. Before you begin broadcasting with RealMedia or MP3, you may want to define archiving as described in this section.

> **Note:** When you archive a low-latency broadcast, Helix Server does not record the latency flag. A simulated live broadcast using the archive therefore experiences the standard end-to-

end latency. For background on live latency, refer to "End-to-End Latency Reduction" on page 192.

Tip: RealProducer can also archive a stream on its local computer as it delivers the stream to Helix Server. You can create the archive on that machine, on Helix Server, or both. Typically, though, the Helix Server computer has more disk space for archiving.

Selectively Archiving Broadcasts

Helix Server is preconfigured to archive broadcasts to its `Archive` subdirectory. You just have to turn the archiving feature on for the "*" rule as described below. However, you may also want to set up different archiving rules. This allows you to archive only some broadcasts, as well as create archive files for different broadcasts in different directories.

The key to selective archiving is the encoder source path. When you broadcast a live stream, you can define an optional path name through the encoder interface. For example, you might define an encoder source path such as `news/`, along with the stream name `live.rm`. The `news/` source path does not correspond to an actual directory path on either the encoder or Helix Server computer. It's just a name sent with the stream name that enables you to use various features, such as selective archiving rules.

You can set up any number of archiving rules. You might archive only the broadcasts that use certain source path names, for instance. Or, you can archive all broadcasts *except* those that use certain source path names. As well, you can archive all broadcasts, using the source path names only to indicate different archive directories. The following table shows possible combinations of using the general archiving rule, "*", along with two selective archiving rules that correspond to `news/` and `talk/` source paths.

Examples of Archiving Rules

Rule	Setting	Destination	Result
*	Enabled	/Archive/	All broadcasts except those with the news/ and talk/ source paths are archived in the /Archive/ directory. Broadcasts using the news/ source path are archived under the /News/ mount point or directory, whereas those using the talk/ source path are archived under /Talk/.
/news	Enabled	/News/	
/talk	Enabled	/Talk/	

(Table Page 1 of 2)

Examples of Archiving Rules (continued)

Rule	Setting	Destination	Result
*	Disabled	(any)	Only broadcasts using the news/ or talk/ source paths are archived. All archive files are created in the /Archive/ directory.
/news	Enabled	/Archive/	
/talk	Enabled	/Archive/	
*	Enabled	/Archive/	All broadcasts except those using the talk/ source path are archived. Archive files are created in the /Archive/ directory, unless they use the news/ source path. In that case, they are archived under /News/.
/news	Enabled	/News/	
/talk	Disabled	(any)	

(Table Page 2 of 2)

Setting Up Archiving

Broadcast archiving is not turned on by default. The following procedure explains how to activate archiving, define new archiving rules, and set up archiving options.

➤ To set up live archiving:

1. In Helix Administrator, click **Broadcasting>Live Archiving**.
2. Each entry in the **Source Paths** box sets up a different archiving rule. Helix Server predefines the "*" rule, which automatically archives all broadcasts to the same directory. To archive all broadcasts, simply enable the "*" entry as described in the next step. To archive broadcasts selectively, create a new source path:
   a. In **Source Paths**, click the "+" icon to add a new path.
   b. Under **Edit Source Path**, enter the broadcast path name sent by RealProducer. For more information, see "Selectively Archiving Broadcasts" on page 132.
3. From the **Archiving** list, select Enabled to activate the archiving rule selected in the **Source Paths** box.

   Tip: If you want to archive only selected broadcasts, be sure to choose Disabled for the "*" rule, and enable only the selective rules.

4. In the **Destination Path** box, indicate where Helix Server stores the archived files for the selected rule. Encase the text in forward slashes, as in /Archive/. Helix Server matches the entry to one of the following elements, searching for a match in the following order:

- **Mount point.** To archive files to another machine, set up a new on-demand mount point as described in "Adding a Mount Point for On-Demand Clips" on page 95.

  Warning! Archiving does not work if the mount point uses `Network` instead of `Local` for **Base Path Location**. For more information, refer to "Adding a Mount Point for On-Demand Clips" on page 95.

- **Directory path.** Directory names are relative to the `Content` directory. If the directory does not exist already, Helix Server will create it. The default location is the `Archive` subdirectory under `Content`.

5. To create multiple archive files limited by size, enter the maximum size for each file in Megabytes in the **Limit Archive Files By Size** box. Archive files are numbered sequentially in the base file name. For example, if the broadcast stream is named `live.rm`, the first archive file is named `live0.rm`, the second is `live1.rm` and so on.

   Note: A number *after* the file extension indicates an earlier broadcast that used the same stream name as a later broadcast. See "Maintaining Archives for Repeated Broadcasts" on page 135 for more information.

6. To create multiple archive files limited by broadcast time, enter the appropriate time in the **Limit Archive Files By Time** boxes. To create a new archive file every 30 minutes during the broadcast, for example, enter `30` in the **Minutes** box. As with files limited by size, archive files limited by broadcast time are numbered sequentially.

   Tip: Generally, you limit archives by size or time. You can select both methods, however, to create archive files according to the first limit reached. For example, you can create a new archive file whenever the preceding file reaches 30 Megabytes in size, or has recorded 15 minutes of the broadcast, whichever comes first.

7. Click **Apply**.

Maintaining Archives for Repeated Broadcasts

When you repeat a broadcast, such as a live news show rebroadcast each day, you can use the same stream name, and archive each broadcast in the same directory. In this case, Helix Server automatically renames older archive files by appending a unique number after the file extension. For example, if the archive file `dailynews.rm` exists in the archive directory when Helix Server begins to archive a new `dailynews.rm` stream, Helix Server moves the existing archive to a name such as `dailynews.rm.86400`. The appended number is related to a timestamp, so larger numbers indicate newer files.

Streaming Archived Files

An archived broadcast file functions just like an on-demand clip, and you can stream it by writing links to it as described in Chapter 5. You can also use SLTA, which Chapter 10 describes, to rebroadcast archive files as if they were live events. It is easiest to stream the archive from the directory where you saved it, but you can also move it to a different directory. A Web page link to a RealMedia archive that resides in the default `Archive` directory looks like this:

```
http://helixserver.example.com/ramgen/Archive/concert.rm
```

If you saved several archive files for a single broadcast, streaming the entire event requires that you provide a Web page link to each archive. However, you can also list all archive files in order within a single Ram file to stream the entire archived broadcast. For more on Ram files, see "Using Metafiles" on page 88.

> **Tip:** When you list archives in order within a `<seq>` tag in a SMIL file, the sequence acts like a unified broadcast. That is, the RealPlayer timeline slider does not reset when each archive file begins to play. For more information, see the SMIL sequences chapter of *Introduction to Streaming Media*.

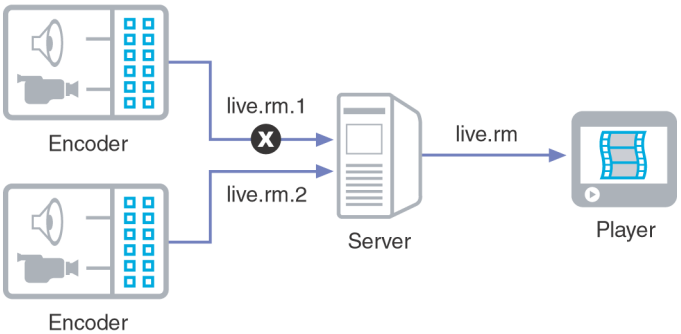
Using Broadcast Redundancy

When broadcasting any media format, you can specify one or more backup encoders. Each encoder delivers the same stream name marked with a unique delimiter, such as `.1` or `.2`. When the encoders connect to Helix Server to deliver their live event streams, they form a queue based on their connection order. Consider this example:

live.rm.2 connects first
live.rm.3 connects second
live.rm.1 connects third

Under normal circumstances all media players receive the stream live.rm, and have no knowledge of the encoder sending the stream. In the preceding example, live.rm originates as live.rm.2. If the encoder delivering live.rm.2 fails, media players reconnect to the next live.rm stream in the queue, which is live.rm.3. If live.rm.2 returns, it goes to the bottom of the queue. A subsequent failure of live.rm.3 causes media players to connect to live.rm.1, and so on.

Broadcast Redundancy



Support for encoder redundancy is enabled automatically, and requires no Helix Server setup. The section on broadcasting a certain media format explains how to implement redundancy for that media format. RealNetworks recommends that you read the following sections, however, to understand better how encoder redundancy works.

> **Tip:** To provide optimal redundancy, each encoder should be as independent as possible. For example, use multiple video cameras connected to separate computers that use different power supplies and network connections.

Modifying Encoder Redundancy Settings

Helix Server is preconfigured to support redundant encoders for all media types. The following procedure explains how to change overall encoder redundancy settings in case you want to modify how this feature works with any media encoder.

➤ To modify encoder redundancy settings:

1. Click **Broadcasting>Broadcast Redundancy**.
2. If you want to turn off broadcast redundancy, choose `Disabled` from the **Broadcast Redundancy** pull-down list. Otherwise, leave this set to `Enabled`.
3. The character specified in the **Delimiter** pull-down list separates the stream names from the source numbers. For example, the primary RealMedia stream might be `live.rm.1`, while the backup is `live.rm.2`. The period delimiter is recommended, because it is the most extensible character in UNIX shells. However, you can choose any one of the following:

   ^ (carat)

   ' (single quote)

   ~ (tilde)

4. The **Timeout** box sets the number of seconds that Helix Server waits for an interrupted stream to return before switching to a backup stream. The default value is 2, but you can set a range from 0 to 30.
5. The **Reconnect** pull-down list specifies how viewers using RealPlayer and RealPlayer 8 receive the backup stream should the primary stream become unavailable. The default value `Auto` causes the media player to switch to the new stream automatically. Choosing `Manual` displays the message defined in the **Error Message** field, and requires users to click the **Play** button to switch to the new stream.

   Note: Viewers must reconnect manually, regardless of the setting in the **Reconnect** list if they are using a version of RealPlayer earlier than RealPlayer 8, or using any other media player, such as Windows Media Player or QuickTime Player.

6. The **Mount Point** box defines the mount point to use in links to indicate that broadcast redundancy is used. The default value is `/redundant/`.
7. The **Error Message** field holds the text of the message that appears if **Reconnect** is set to `Manual`. This message should tell users how to connect to the new stream. The default message is:

   `Broadcast timed out; click Play button to restart.`

8. Click **Apply** to save the changes. You'll need to restart Helix Server if you changed the mount point.

Using Broadcast Redundancy with Other Features

The following table explains how redundant broadcasting interacts with other Helix Server features.

Broadcast Redundancy Used with Other Features

Other Features	Notes
SLTA	You can use redundant sources with SLTA.
Archiving	Helix Server archives the incoming source, but doesn't store the source number. If sources are `live.rm.1` and `live.rm.2`, for example Helix Server archives a file named `live.rm`, without the delimiter and number.
Splitting	A transmitter that uses the redundant encoders feature sends out its streams just like any other live broadcast. To create a system with multiple layers of backups, you should configure multiple transmitters to name their broadcasts with the same file name, plus the delimiter and a unique number.
Multicasting	Redundant live sources can be used with a back-channel multicast. Redundancy does not function with a scalable multicast.
Access Control, Authentication	Use access control and authentication the same as with any other live content.
Java Monitor	Redundant sources are displayed like any other live broadcasts.
Reporting	A record is created in the basic access log for any client that connects to a live broadcast originating from redundant sources. You can't determine which redundant source is in use, only that the redundancy feature is in use. See "GET Statements" on page 337 to see how redundant sources appear in the basic access log.

Broadcasting RealMedia

To broadcast RealAudio or RealVideo, you use RealProducer to capture and encode the live event. If you have an older version of RealProducer, such as RealProducer 8.5, you can continue to use it to deliver live streams, as described in "Encoding with an Older Version of RealProducer" on page 142. The section "Setting Up Account-Based Broadcasting" on page 140 explains

how to use RealProducer in a mode that emulates broadcasts originating from earlier versions of RealProducer.

> **Tip:** RealProducer can deliver live streams in *push mode* or *pull mode*. In these modes, RealProducer acts like a Helix Server transmitter in a splitting arrangement. These modes are more powerful than the connection methods described in the following sections. They require more setup, though, and are described in Chapter 9.

Broadcasting with SureStream

Using SureStream technology, you can broadcast RealMedia at multiple bandwidths using any encoder from RealProducer G2 to the latest version. In a SureStream broadcast, each RealPlayer selects an encoding appropriate for its connection speed. To reach multiple bandwidth audiences without using SureStream, you run separate RealProducer encoders (typically on separate machines), broadcasting each stream under a different stream name.

Note that when you archive a live broadcast that uses SureStream, Helix Server creates several temporary files that correspond to different SureStream streams. It merges these temporary files into the final archived file (or files) when the broadcast finishes. The merging time may take longer than the broadcast. Be sure not to stop Helix Server or delete the temporary files before the merging completes.

> **For More Information:** For more on archiving, see "Archiving Broadcasts" on page 131.

Using SMIL in Broadcasts

You can use SMIL 1.0 or SMIL 2.0 to add prerecorded content to a live broadcast. Within a SMIL file, you treat a broadcast like any other clip, furnishing a URL that points RealPlayer to the live stream instead of an on-demand clip. You can assign broadcast streams to SMIL regions, and group a broadcast with on-demand clips using `<seq>` or `<par>` tags.

SMIL can deliver an on-demand RealPix slideshow along with live RealAudio, for example, when both are in a `<par>` group. It cannot synchronize the on-demand clip with the live stream, however. This is because the on-demand clip's timeline starts when the viewer requests the presentation, whereas the broadcast stream's timeline starts when the broadcast begins.

To illustrate this, suppose that viewer A requests the presentation 2 minutes after the broadcast begins, and viewer B requests it 4 minutes after the broadcast begins. At 10 minutes into the broadcast, both viewers hear the same audio, but viewer A's RealPix clip is at its 8-minute mark, whereas viewer B's clip is at its 6-minute mark. Hence the relationship between the two timelines varies for each viewer.

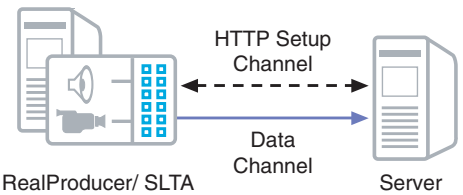
For More Information: For more on SMIL, see *RealNetworks Production Guide*.

Setting Up Account-Based Broadcasting

If you are a broadcasting beginner, you can use RealProducer in its *account-based* mode. This mode, which is a simplified subset of the newer push mode, emulates the broadcasting connection used in earlier versions of RealProducer. Although not as robust as the full-fledged push mode, it requires little setup, so it provides a good introduction to broadcasting RealMedia.

When RealProducer connects to Helix Server in account-based mode, it uses an HTTP connection to Helix Server to discover the best way to communicate. Because this connection is authenticated, RealProducer needs a valid user name and password on Helix Server. RealProducer then sets up a data channel based on the most reliable network path it finds to Helix Server.

RealProducer in Account-Based Mode



For More Information: See "Encoder Validation" on page 270 for information about setting up a user name and password to enable RealProducer to connect to Helix Server.

Warning! Do not change the unicasting configuration if a broadcast is in progress.

➤ To set up account-based broadcasting:

1. Click **Broadcasting >RealNetworks Encoding**. The necessary settings reside in the **9.0 Producer** section.
2. In the **Port Range** box, select the range of ports on Helix Server where RealProducer sends its live feed. Default values are `50001` to `50050`. Two ports are required for each live feed, so the default values allow up to 25 simultaneous connections. You may need to change or limit the port range to work around firewall restrictions, which are described in Chapter 11.
3. RealProducer needs a valid user name and password to connect to Helix Server. In the **Authentication** pull-down list, select the realm that holds user names and passwords to authenticate the encoder connection. The default realm for RealProducer is `SecureRBSEncoder`.

   Warning! Do not choose a realm that uses Windows NT LAN Manager. NTLM authentication is not supported between Helix Server and RealProducer.

4. If you need to add a user name and password pair to validate the encoder connection, click the **Create user names and passwords** link.

   For More Information: The section "Managing Users and Passwords" on page 278 describes how to add user names and passwords to a realm.

5. Click **Apply** to save the changes.

   Note: You don't define a mount point with account-based broadcasting. The predefined mount point used for all account-based broadcasts is `/broadcast/`.

Starting the Account-Based Broadcast

RealProducer initiates an account-based broadcast by contacting Helix Server and delivering the encoded stream. RealProducer has a server connection wizard in which you select push broadcasting, then specify an account-based login. You need to supply the Helix Server address, as well as a user name and password set up under Helix Server authentication.

In RealProducer, you designate a stream name, and can include a path if you want to archive the stream selectively. If you use multiple RealProducers for

broadcast redundancy, start each encoder at the same time (or as close as possible). Ensure that each stream has the same stream name, a delimiter (a period by default), and a unique stream integer, starting with 1. For example, your stream names might be the following:

primary stream: `live.rm.1`

backup stream: `live.rm.2`

> **Note:** Even though each stream has a delimiter and unique integer after its stream name, the link to the broadcast uses just the common stream name (`live.rm`).

> **For More Information:** See *RealProducer User's Guide* for instructions on setting up the encoder for broadcasting. To create the broadcast URLs, refer to "Linking to Unicasts" on page 151. The section "Using Broadcast Redundancy" on page 135 explains the use of multiple encoders.

Encoding with an Older Version of RealProducer

Helix Server supports broadcast connections from earlier RealNetworks encoders, from RealProducer G2 to RealProducer 8.5 Although you typically do not need to change any Helix Server settings, follow one of the procedures below to ensure that your Helix Server setup is correct.

➤ To set up encoding with RealProducer G2 through 8.5:

1. Click **Broadcasting >RealNetworks Encoding**. The necessary settings reside in the **G2 to 8.5 Producer** section.
2. If you wish, you can change the **Mount Point** setting from the default `/encoder/`. Links to the broadcast include this mount point.
3. The **Port** box sets the port on which Helix Server listens for the live broadcast stream. If you change this value, be sure to give the new value to the person running RealProducer.
4. The **Timeout** box sets the number of seconds that Helix Server waits after a break in packet reception to shut down the connection and terminate the broadcast. The default value is 30 seconds.
5. To validate the encoder connection, select the name of the appropriate realm from the **Authentication** box. The default realm is `SecureEncoder`.

6. If you need to add a user name and password pair to validate the encoder connection, click the **Create user names and passwords** link.

   For More Information: The section "Managing Users and Passwords" on page 278 describes how to add user names and passwords to a realm.

7. Click **Apply** to save the changes. You'll need to restart Helix Server if you changed the mount point.

Starting the Legacy Broadcast

In legacy broadcasting, RealProducer initiates the broadcast by contacting Helix Server. You supply the Helix Server address, listen port, and a user name and password (if required). You designate a stream name, and can include a path if you want to archive the stream selectively. Legacy broadcasts can also use a backup encoder, with the two encoders specifying number-delimited stream names, such as `live.rm.1` and `live.rm.2`.

For More Information: See your *RealProducer User's Guide* for instructions on setting up the encoder for broadcasting. To set up broadcast URLs, see "Linking to Unicasts" on page 151. The section "Using Broadcast Redundancy" on page 135 explains the use of multiple encoders.

Broadcasting Windows Media

Helix Server is preconfigured to deliver live broadcast streams in the Windows Media format. It includes support for the MMS protocol, a Windows Media broadcast mount point, and an ASXgen mount point for launching Windows Media Player. Broadcasting in Windows Media format therefore requires little setup. Windows Media Encoder redundancy is supported, but archiving is not.

Note: Multiple bit rate (MBR) encoding is supported only with pull broadcasting with Windows Media Encoder version 7. It is not supported with push or pull broadcasting using Windows Media Encoder version 9 or later.

Setting up a Windows Media Pull Broadcast

In a Windows Media pull broadcast, Helix Server pulls a stream from Windows Media Encoder version 7 or later over an HTTP connection,

delivering the stream to Windows Media Players over the MMS or HTTP protocol.

➤ To set up a Windows Media pull broadcast:

1. Click **Broadcasting> Windows Media Encoding**.
2. The **Mount Point** box lists the default Windows Media mount point of `/wmtencoder/`, which will appear in broadcast URLs. RealNetworks recommends that you use this value. You can define only one Windows Media broadcast mount point on each Helix Server, but you can deliver multiple push and pull broadcasts simultaneously through this single mount point.
3. Skip over the settings for **Push Encoding Port** and **Push Encoder Authentication**, which are used only with push encoding.
4. In the **Windows Media Sources** list, click the "+" icon and edit the default text that appears in the **Source Description** box. Your entry should describe the Windows Media Encoder sending the stream. It is for your reference only, and is not included in URLs.
5. In the **Host** field, enter the host name, IPv4 address, or IPv6 address of the Windows Media Encoder.
6. In the **Port** field, enter the HTTP port of the Windows Media Encoder.
7. In the **Stream Name** field, enter a name for the Windows Media stream. The stream name, which appears in links to the broadcast, typically uses the same format as prerecorded clips, including a short base name, along with the standard file extension, whether `.asf`, `.wmv`, or `.wma`. Optionally, you can precede the stream name with a path:

   ```
   news/live.wmv
   ```

   > **Note:** The path does not correspond to a physical path. Instead, you can use this when splitting the stream to direct it to some receivers but not others. For more information, see "Multiple Splitting Definitions" on page 190.

8. Repeat the preceding steps to set up another Windows Media broadcast, or to define a redundant encoder for the same broadcast. To use encoder redundancy, enter a stream delimiter after the extension, as in `live.wmv.1`. The backup stream should have the same stream name, but use `.2` as its delimiter.

For More Information: See "Using Broadcast Redundancy" on page 135.

9. Helix Server is ready to broadcast when you click **Apply**.

Note: If you change the broadcast mount point, you need to restart Helix Server before you can broadcast a live Windows Media stream.

Running the Windows Media Pull Broadcast

In a pull broadcast, Helix Server pulls the stream from the encoder on the first request by Windows Media Player. This means that you can set up Helix Server and start the encoder in advance, publishing the URL to the live stream only when you want the broadcast to begin.

When the first Windows Media Player requests the broadcast, Helix Server acquires the stream, causing a longer-than-normal delay. After this initial request, however, the broadcast is queued and subsequent users experience only the normal stream acquisition latency. If all media players quit the broadcast, Helix Server drops the live stream, reacquiring it on the next media player request.

As long as media players request the URL, Helix Server attempts to pull the requested stream from the encoder. To disable a broadcast whether or not Windows Media Encoder has stopped encoding the stream, delete the source name in the **Windows Media Encoding** page, or clear the **Stream Name** field and click **Apply**.

For More Information: Consult your Windows Media Encoder documentation for information about setting up the live encoding process. To set up broadcast URLs, see "Linking to Unicasts" on page 151.

Setting up a Windows Media Push Broadcast

In a Windows Media push broadcast, Helix Server receives a stream from Windows Media Encoder version 9 or later on a predefined port. It then delivers the stream to Windows Media Players over the MMS or HTTP protocol.

➤ To set up a Windows Media push broadcast:

1. Click **Broadcasting> Windows Media Encoding**.

2. The **Mount Point** box lists the default Windows Media mount point of `/wmtencoder/`, which will appear in broadcast URLs. RealNetworks recommends that you use this value. You can define only one Windows Media broadcast mount point on each Helix Server, but you can deliver multiple push and pull broadcasts simultaneously through this single mount point.
3. For **Push Encoding Port**, choose a port on Helix Server where Windows Media Encoder delivers its live stream over HTTP. Multiple encoders can connect through this port. The default port is 8080.
4. In the pull-down menu for **Push Encoder Authentication**, choose the authentication realm to use to verify connections by Windows Media Encoder. The default realm is `SecureWMEncoder`, which verifies encoder user names and passwords using `Digest` authentication. If you do not want to authenticate connections, choose `No Authentication`.

   Warning! If you turn off authentication, any Windows Media Encoder that can access the push encoding port will be able to broadcast through Helix Server.

   Note: Helix Server does not support NTLM authentication for encoder connections. You must use `Digest` authentication, or no authentication.

5. If you need to add a user name and password pair to validate the encoder connection, click the **Create user names and passwords** link.

   For More Information: The section "Managing Users and Passwords" on page 278 describes how to add user names and passwords to a realm.

6. Skip over the remaining fields, which are used only when pulling a broadcast from Windows Media Encoder.
7. Helix Server is ready to receive the broadcast stream when you click **Apply**.

   Note: If you change the broadcast mount point, you need to restart Helix Server before you can broadcast a live Windows Media stream.

Running a Windows Media Push Broadcast

In a push broadcast, Windows Media Encoder delivers the live stream to Helix Server as soon as the encoding begins. For this reason, it is better not to start the encoding process until shortly before you intend to broadcast to media players. Otherwise, you waste bandwidth on Helix Server by queueing a live stream that is not yet available to users.

To deliver the stream, Windows Media Encoder must be configured to contact Helix Server on the push encoding port. The single-rate, live stream must also have a name and indicate the Windows Media live stream mount point, as shown in the following example:

```
/wmtencoder/live.wmv
```

If you are using multiple encoders to deliver redundant streams, include a unique, numbered stream delimiter for each stream as described in "Using Broadcast Redundancy" on page 135:

```
/wmtencoder/live.wmv.1
```

Optionally, you can insert a virtual path to direct the stream to some receivers but not others, which the section "Multiple Splitting Definitions" on page 190 explains:

```
/wmtencoder/news/live.wmv
```

> **Note:** If you are using multiple encoders to deliver different broadcast streams to Helix Server (such as separate news and sports streams), each encoder must specify the same mount point but use a different stream name. Virtual paths may be the same or different depending on how you intend to split the streams.
>
> **For More Information:** Consult your Windows Media Encoder documentation for information about setting up the live encoding process. To set up broadcast URLs, see "Linking to Unicasts" on page 151.

Broadcasting QuickTime, MPEG, and RTP-Based Media

Helix Server can broadcast to the QuickTime Player streams that originate from the Sorenson Broadcaster, Apple's Darwin Server, or Playlist Broadcaster. QuickTime broadcasting is based on the RTSP control protocol and the RTP packet format. Helix Server's support for RTSP and RTP enable it to broadcast

other RTP-based media formats as well, including MP3 and MPEG-4. Helix Server is preconfigured to support QuickTime and RTP-based media. You typically do not need to modify Helix Server to broadcast these formats.

> **For More Information:** For more on the RTP format itself, refer to "Packet Formats" on page 251.

> **Note:** Helix Server accepts only MPEG streams encoded a single bit rate. It rejects an encoder connection if the stream is encoded at multiple streaming speeds.

Using SDP Files with Helix Server

The core of RTP-based streaming is the *Session Description Protocol* (SDP) file. The encoder produces this file, which provides general information about the encoded stream. The file is then delivered, usually by FTP, to a specific Helix Server directory. Typically, Helix Server pulls the stream from the encoder, based on information in the SDP file, when the first media player requests the broadcast. However, Helix Server can also scan its SDP directory at regular intervals, cueing the broadcast when an SDP file arrives.

It's important to understand that media players do not receive the SDP file, even though the URL to the broadcast appears to link to the SDP file. The request URL uses the SDP file name to identify the broadcast. However, the request comes through a preconfigured Helix Server mount point. This causes Helix Server to send the media player information about how to connect to the broadcast stream on Helix Server, rather than the original stream sent by the encoder.

Changing RTP Broadcast Procedures

To deliver an RTP-encoded stream, you typically do not need to change any Helix Server settings. You may want Helix Server to monitor for the arrival of the SDP file, though, so that it can cue the broadcast and speed the initial broadcast delivery. The following procedure explains how to modify broadcast settings.

➤ To modify a QuickTime or RTP-based broadcast:

1. Click **Broadcasting>QT & RTP Encoding**.
2. The **Mount Point** box defines the mount point used in the broadcast URL. The `/rtpencoder/` mount point is predefined. You can use this mount point

to broadcast QuickTime and any RTP-based media. For background on mount points, see "Mount Points" on page 86.

3. The **Base Mount Point** box defines a mount point or directory on Helix Server where the media encoder places the SDP file. The predefined `/rtpencodersdp/` entry corresponds to a subdirectory under the main `Content` directory.

   > **Note:** If you specify a different subdirectory or mount point, ensure that the directory exists before you restart Helix Server.

4. For **Connection Timeout**, define the time in seconds that Helix Server waits for the encoder to respond with a stream when the first media player requests the broadcast. If the timeout value expires before the encoder responds, Helix Server terminates the broadcast. The default value is 10 seconds. You can set the value higher if you expect a higher initial latency.

5. The **End of Session Timeout** box defines the time in seconds that Helix Server waits for the encoder to respond if it has stopped sending data but has not indicated that the broadcast has stopped. If the timeout value expires before the encoder responds, Helix Server terminates the broadcast. The default value is 10 seconds. RealNetworks does not recommend setting this value lower.

6. By default, the **Enable SDP Directory Scan** pull-down is set to `No`, which causes Helix Server to pull the stream from the encoder when the first client requests the broadcast. If you change this to `Yes`, Helix Server scans the SDP file directory at regular intervals, connecting to the encoder and cueing the broadcast when it finds a new SDP file. Hence, Helix Server prepares the stream on the arrival of the SDP file, helping to speed the delivery of the broadcast.

   > **Tip:** Because cueing the broadcast consumes resources even if no clients request the broadcast, RealNetworks recommends that you enable this feature only if you anticipate that the encoder will deliver the SDP file shortly before the first client requests the stream.

7. If you set Helix Server to scan for SDP files, enter in the **SDP Directory Scan Interval** box the frequency in seconds that Helix Server scans the SDP directory. The default is 5 seconds.

8. Click **Apply** to save the changes. Making any changes to this page requires a Helix Server restart.

Starting an RTP-Based Broadcast

As it prepares to broadcast, the QuickTime or RTP-based encoder creates an SDP file that identifies the encoded stream and selects the Helix Server port that receives the broadcast data. The broadcaster then delivers the SDP file to Helix Server's SDP directory. This directory can contain multiple SDP files for any number of broadcasts.

The predefined SDP directory for QuickTime and RTP-based broadcasts is located at the following path in a default installation of Helix Server on Windows:

```
C:\Program Files\Real\Helix Server\Content\rtpencodersdp
```

On UNIX, the directory is under the `Content` directory of the main installation directory, as in the following example:

```
/usr/local/Real/HelixServer/Content/rtpencodersdp
```

Keep in mind that the broadcast request URL does not list the actual directory that contains the SDP file. Instead, it uses the appropriate mount point, which is `/rtpencoder/` by default. For examples of this, see "QuickTime and RTP-Based Player Examples" on page 153.

> **Tip:** You can create subdirectories for different SDP files if you want to split different broadcasts in different ways, as described in "Multiple Splitting Definitions" on page 190. The subdirectory name must then precede the SDP file name in the request URL.

> **Warning!** Helix Server can receive only one broadcast connection on a port. If an RTP-based encoder attempts to deliver a broadcast stream to a port already in use, Helix Server rejects the stream and logs the action in its error log. You therefore need to communicate to content providers the correct ports to use, and ensure that firewall restrictions allow enough open ports to support all broadcasting needs.

Stopping an RTP-Based Broadcast

You must stop an RTP-based broadcast by terminating the encoder stream. Deleting the broadcast's SDP file from the SDP directory does not stop a broadcast in progress. To restart a broadcast, terminate the stream on the encoder, restart the stream, generate a new SDP file, and deliver that file to the SDP directory on Helix Server. When a broadcast ends, you can delete the SDP file from the SDP directory.

Linking to Unicasts

Links to live broadcasts look like links to on-demand clips, but use the broadcast mount points to direct the clients to live streams. Chapter 5 explains link formats in general. For information about the mount points to add to URLs to implement user name and password validation in broadcasts, see Chapter 13.

Linking from a Web Page

You can use the Ramgen or ASXgen utility described in "Using a Client Launch Utility" on page 90 to launch the appropriate media player through a Web page link. The Web page URL uses the following format, in which the HTTP port is not required if Helix Server uses port 80 for HTTP requests:

`http://`*`address:HTTPPort/launch_utility/mount_point/path/stream_name`*

RealMedia Link Examples

Using default values, a Web page URL to a broadcast originating from RealProducer in account-based mode looks like this:

```
http://helixserver.example.com/ramgen/broadcast/news/live.rm
```

The `/broadcast/` mount point is the default. A path such as `news/` is optional. You include it only if RealProducer specifies a path along with the stream name. You can use the path to select an archiving rule, for instance, or require user name and password authentication.

A broadcast originating from an earlier version of RealProducer uses the `/encoder/` mount point instead of the `/broadcast/` mount point. A Web page link looks like this:

```
http://helixserver.example.com/ramgen/encoder/news/live.rm
```

If you are using broadcast redundancy with any version of RealProducer, use the /redundant/ mount point instead of the /broadcast/ or /encoder/ mount point. Even though the streams have delimiters such as .1 and .2, you do not use the delimiters in the URL:

```
http://helixserver.example.com/ramgen/redundant/live.rm
```

Windows Media Link Examples

Using default values, a Web page URL to a Windows Media unicast looks like this:

```
http://helixserver.example.com/asxgen/wmtencoder/live.wmv.asx
```

A broadcast that uses encoder redundancy looks like this:

```
http://helixserver.example.com/asxgen/redundant/live.wmv.asx
```

> **Tip:** See "ASXgen for Windows Media Player" on page 91 for information about the extra .asx extension used in Windows Media URLs.

Linking through a Metafile

You can also use media client metafiles, which are described in "Using Metafiles" on page 88, to request the broadcast. Links entered in a metafile, or typed directly into a media client, use the following format, in which the port value is not necessary if Helix Server uses the default RTSP port (554) or MMS port (1755):

protocol://address:Port/mount_point/path/stream_name|SDPFileName

RealMedia Link Examples

In a Ram file or SMIL file, a URL to a broadcast originating from RealProducer in account-based mode looks like this:

```
rtsp://helixserver.example.com/broadcast/news/live.rm
```

Here, the news/ path is optional, and specified along with the stream name by RealProducer. A Ram or SMIL file URL to a broadcast originating from an earlier version of RealProducer looks like this:

```
rtsp://helixserver.example.com/encoder/news/live.rm
```

If you are using broadcast redundancy, use the /redundant/ encoder mount point instead of the /broadcast/ or /encoder/ mount point. Even though the

streams have delimiters such as .1 and .2, you do not use the delimiters in the URL:

```
rtsp://helixserver.example.com/redundant/live.rm
```

Windows Media Link Examples

Windows Media broadcast links in an ASX file use the following format:

```
mms://helixserver.example.com/wmtencoder/live.wmv
```

The file should also contain an alternate, HTTP link with an explicit HTTP port number for Windows Media Player 11 and later:

```
http://helixserver.example.com:8080/wmtencoder/live.wmv
```

An MMS link to a broadcast that uses encoder redundancy looks like this:

```
mms://helixserver.example.com/redundant/live.wmv
```

The HTTP version looks like the following:

```
http://helixserver.example.com:8080/redundant/live.wmv
```

QuickTime and RTP-Based Player Examples

URLs to RTP-based broadcasts, such as QuickTime or MPEG, specify the SDP file used for the broadcast. QuickTime broadcasts indicated in a reference movie file also specify the SDP file. Remember, though, that the media player does not request the SDP file from the directory where the media encoder places it. Instead, the player requests the file through the appropriate broadcast mount point, as shown here:

```
rtsp://helixserver.example.com/rtpencoder/broadcast.sdp
```

If encoder redundancy is used, the link uses the /redundant/ mount point instead:

```
rtsp://helixserver.example.com/redundant/broadcast.sdp
```

Playing a Standby Message

If a live broadcast has not started, or is interrupted and has not returned when a viewer tries to reconnect, you can send a message that indicates general information about the broadcast, such as when it is scheduled to play, and what viewers can do if the stream is interrupted. You do this by making a file that contains the message you want to display, and placing it in a subdirectory with the same name as the live mount point.

> **Tip:** For this feature to work with RTP-based broadcasts, you must set **Enable SDP Directory Scan** to `Yes`. For more information, refer to "Changing RTP Broadcast Procedures" on page 148.

➤ To create a standby message:

1. Under Helix Server's content directory, create a subdirectory with the same name as the live mount point. Keep in mind that the live mount point depends on the type of RealNetworks encoder being used. To support all default broadcast mount points, you would create the following subdirectories in a default installation of Helix Server on Windows:

   ```
   C:\Program Files\Real\Helix Server\Content\broadcast
   C:\Program Files\Real\Helix Server\Content\encoder
   C:\Program Files\Real\Helix Server\Content\wmtencoder
   C:\Program Files\Real\Helix Server\Content\rtpencoder
   C:\Program Files\Real\Helix Server\Content\redundant
   ```

   or on a typical installation on UNIX:

   ```
   /usr/local/Real/HelixServer/Content/broadcast
   /usr/local/Real/HelixServer/Content/encoder
   /usr/local/Real/HelixServer/Content/wmtencoder
   /usr/local/Real/HelixServer/Content/rtpencoder
   /usr/local/Real/HelixServer/Content/redundant
   ```

2. Create a clip in the same format as your broadcast, such as RealAudio, RealVideo, Windows Media, or QuickTime, that contains the message to stream if the broadcast is not available.

   > **Tip:** You may want to include information about when the visitor should check back, keeping in mind the different time zones in which viewers may reside.

3. Place the appropriate standby clip in each of the subdirectories. Then, if a live stream fails to arrive, Helix Server searches for the actual directory and clip that matches the broadcast mount point and stream name in the URL. In other words, it streams the prerecorded clips that use the stream names and that reside in the content subdirectories that mimic the broadcast mount points.

CHAPTER

8

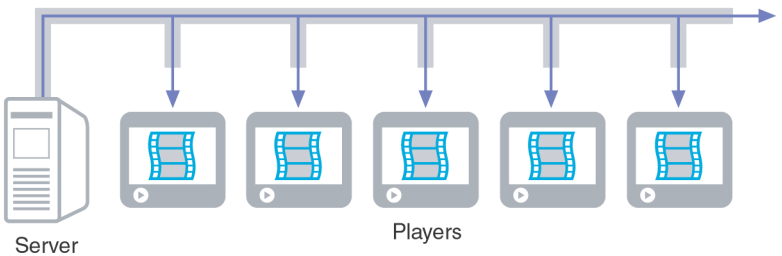
MULTICASTS

Multicasting is an alternative to unicasting that reduces the number of broadcast streams in use. Although it can increase the audience for a live event by reducing the broadcasting bandwidth, it requires a specially configured network, and is more suited for intranets than Internet delivery. You can multicast RealMedia, QuickTime, Windows Media, MPEG, and a number of RTP-based formats.

Understanding Multicasts

As Chapter 7 explained, unicasting delivers a unique broadcast stream to each media player. In contrast, multicasting sends a single live stream to multiple players. The multicast can be a live event, or a prerecorded clip broadcast by SLTA, which Chapter 10 explains. The players connect to the stream, rather than to the Helix Server, as shown in the following illustration.

Multicasting

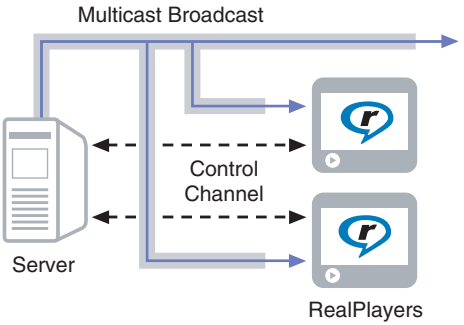


Back-Channel Multicasting

Back-channel multicasting works with all RealNetworks media players that support the RTSP protocol. In this method of multicasting, each media player maintains a control channel to Helix Server. A media player uses its channel to send commands such as **Stop**, and to deliver statistics about the quality of service to the archive log. The channel lets Helix Server receive a user name

and password if authentication is used. It also enables the Server Monitor described in Chapter 17 to track how many players are viewing the multicast.

Back-Channel Multicasting



Note: Because each player uses a control channel, back-channel multicasting is limited to the number of client connections licensed to your Helix Server.

Unicast Failovers

If a RealNetworks media player is not multicast-enabled, or cannot connect to the back-channel multicast, it fails over to a unicast automatically. This ensures that all players can receive the broadcast. Because each unicast stream consumes extra bandwidth and Helix Server overhead, you can choose to disable the failover feature and provide just the multicast. In this case, a player not able to participate in a multicast receives an error message when attempting to connect to the broadcast.

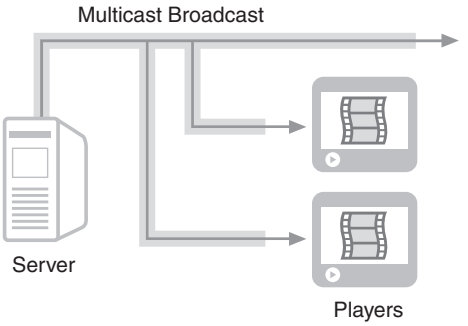
Tip: You can use back-channel multicasting with the failover feature for all broadcasts to RealNetworks media players. By default, players attempt a back-channel multicast connection first, switching to unicast if the failover feature is enabled, and the multicast is not available. Hence, enabling an automatic multicast for all broadcasts can help conserve bandwidth.

Scalable Multicasting

Using a scalable multicast, you can broadcast to an unlimited number of media players because the transmission is one-way. Unlike back-channel multicasting, scalable multicasting does not use a control channel. Thus, it uses less bandwidth, administrative overhead, and system resources on Helix

Server. Scalable multicasting works with RealPlayer G2 and later. You can also multicast to any RTP-based media player that complies with scalable multicasting standards, including Apple's QuickTime Player.

Scalable Multicasting



Session Description Files

Viewers connect to scalable multicasts by clicking a link to a Session Description Protocol (SDP) file, which Helix Server automatically generates. All data is multicast on the network once, and media players do not connect to Helix Server during the multicast. Because of this, tools such as Server Monitor do not track client connections and activity during the broadcast.

Authentication and Statistics

Because media players do not connect to Helix Server directly during the multicast, you cannot enforce user name and password validation in a scalable multicast. Optionally, you can have players connect to Helix Server or a Web server to deliver quality of service statistics when the broadcast ends (or they disconnect).

Unicast Failovers

Like back-channel multicasting, scalable multicasting has a failover feature that lets you direct RealNetworks media players to a unicast if they cannot receive the multicast. You can provide a unicast on the same Helix Server that hosts the multicast, or choose a different Helix Server. As well, you can direct players to a Web page that indicates that the multicast is not available, and provides information about broadcast alternatives.

Windows Media Multicasts

Helix Server can deliver scalable multicasts in the Windows Media format to Windows Media Players. Each multicast is delivered by Windows Media Encoder, and made available to players on a single multicast channel. Players click a Web page link to an NSC file to join the multicast. You can designate a unicast URL for players that cannot join the multicast. Unlike with RealMedia back-channel multicasts, Helix Server cannot make all Windows Media unicasts automatically available as multicasts. As well, archiving of Windows Media multicasts is not supported.

Network Configuration for Multicasts

To use multicasting, Helix Server, media players, routers, switches, and all other networking devices between them must be multicast-enabled. For this reason, multicasting is primary used on intranets. However, it is possible to deliver multicasts over the Internet where intermediary network devices have been multicast-enabled. Before using multicasting, verify the following with your network administrator:

- Routers and all equipment in your network are multicast-enabled.
- The machine running Helix Server is correctly configured for multicast support.

> **Tip:** RealNetworks and Microsoft media players are configured for multicast by default, although viewers can turn off multicast support in their player preferences.

Multicast Addresses

A multicast requires the use of a continuous range of multicast addresses on your network. Valid ranges are between `224.0.0.0` and `239.255.255.255`. Check with your network administrator about which multicast addresses are available on your network. On the public Internet, certain ranges in the multicast address space (from `224.0.0.0` to `224.0.0.255`) are reserved, and cannot be used.

The number of addresses required for a multicast depends on the type of multicast you use, as well as the number of streams you deliver. For scalable multicasting, you also need to reserve a number of Helix Server ports for the broadcast. The sections on setting up each type of multicast explain the number of addresses and ports you need.

Note: Helix Server does not support multicasting to IPv6 addresses.

Warning! If you use multiple types of multicasts, such as both back-channel and scalable multicasts, the address ranges you pick cannot overlap.

Packet Time to Live

All multicast broadcasts include a "time to live" feature. As a multicast data packet passes through a multicast-enabled router, its time to live decreases by 1. When the value reaches 0, the router discards the data packet. When you set up a multicast, you specify a time to live of `0` to `255`. The larger the value, the greater the distance a packet can travel. The default value of `16` typically keeps multicast packets within an internal network. The following table summarizes possible values.

Time to Live (TTL) Values

TTL Value	Packet Range
`0`	local host
`1`	local network (subnet)
`16`	intranet
`32`	site
`64`	region
`128`	continent
`255`	world

Multicasts with Multiple Network Interface Cards

If your Helix Server machine has multiple network interface cards (NICs), and you want to ensure that Helix Server always uses a particular NIC for multicasts, use your operating system to set a default address. On Windows, set IP bindings as described in "Binding to an IP Address" on page 66. On UNIX, use the **route** command to associate the multicast route with the appropriate NIC.

Multicasting Used with Other Features

The following sections summarize how multicasting works with other Helix Server features.

Splitting and Multicasting

Transmitters can multicast a stream to receivers, as explained in Chapter 9. This does not require the setup described in this chapter, which concerns only server-to-player multicasts. If the receiver multicasts a stream to media players, however, you must configure the receiver for multicasting as described in this chapter.

Live Archiving and Multicasting

As with all live broadcasts, you can configure Helix Server to archive files for live multicasts in the RealMedia or MP3 format. Helix Server does not archive MPEG-4 or Windows Media multicasts.

Simulated Live Broadcasts with Multicasting

You can multicast a simulated live stream from on-demand clips using SLTA, which Chapter 10 describes.

Helix Proxy and Multicasting

Depending on how the network is configured and the streams are listed in Helix Server, clients whose requests are forwarded by a Helix Proxy may receive different results.

Helix Proxy cannot join a multicast. Instead, it will try to receive the multicast using pull splitting. If pull splitting is enabled on the source Helix Server, Helix Proxy will use that broadcast, instead of connecting to the multicast. The client will receive the broadcast in unicast mode.

If there is a multicast-enabled network between Helix Proxy and the client, Helix Proxy can be configured to re-send its pull split stream by multicast instead.

> **For More Information:** Refer to *Helix Proxy Administration Guide* for information on configuring Helix Proxy.

Firewalls and Multicasting

Multicasts usually take place within an intranet, where broadcasts are not traveling outside a firewall. If a multicast passes through a firewall, the firewall must be specially configured to allow multicast traffic.

Access Control, Authentication, and Multicasting

As with all delivery methods, Helix Server verifies that the client requesting a broadcast is allowed to receive it. If you include the authentication mount

point (/secure/) in the link, Helix Server verifies the viewer's identity. You cannot authenticate scalable multicasts, however.

Reporting and Multicasting

Streams served through back-channel multicasts appear in the basic access log just like unicast material. The basic access log shows which method was used to transmit the stream. Scalable multicasts can be identified in the basic access log by their mount point in the GET statement. If Helix Server is configured for requesting client statistics (see "Gathering Client Statistics" on page 166), the log file will also contain statistics for each client.

Multicast Resources

The Helix Server implementation of back-channel and scalable multicasting is based on open industry standards. You may find the following resources useful.

General Multicasting Information

- **Addresses available for multicast use**—"Assigned Numbers," RFC 1700, available at **http://www.ietf.org/rfc/rfc1700.txt**.

Scalable Multicasting Information

- **RTP**—"RTP: A Transport Protocol for Real-Time Applications," RFC 1889, available at **http://www.ietf.org/rfc/rfc1889.txt**.
- **RTP**—"RTP Profile for Audio and Video Conferences with Minimal Control," RFC 1890, available at **http://www.ietf.org/rfc/rfc1890.txt**.
- **SDP (Session Description Protocol)**—"SDP: Session Description Protocol," RFC 2327, available at **http://www.ietf.org/rfc/rfc2327.txt**.
- **SAP (Session Announcement Protocol)—**"Session Announcement Protocol: Version 2," available at **http://www.ietf.org/rfc/rfc2974.txt**.

Defining Back-Channel Multicasting

Back-channel multicasting is enabled by default, though you need to specify multicast addresses for it to work. You can use this method to multicast any media format that RealNetworks media players can play. Once you've set up multicasting, players attempt to connect in multicast mode first, using a

unicast if the cannot connect to the multicast. This helps to save bandwidth by providing the most efficient possible connection for each player.

Calculating Address Requirements

Determining the number of addresses you need for a back-channel multicast is straightforward. You need just one address per bit rate, regardless of the number of streams. So although a single-rate video technically delivers two streams, one for the audio track and one for the visuals, both tracks can use the same address.

SureStream Broadcasts

For SureStream, you need to reserve one address for each bit rate encoded into the stream. If a SureStream stream is encoded for three audience bandwidth targets, for example, you need three addresses. The duress streams encoded for a particular bandwidth target are not used, and do not require additional addresses.

> **Note:** Media players cannot shift between SureStream streams during the multicast.

Automatic Multicasts

If you intend to leave back-channel multicasting on for all broadcasts, you need enough multicast addresses to accommodate your typical broadcast. It's a good idea to implement a policy, such as using only two bandwidth targets when broadcasting RealVideo. If you have not reserved enough addresses for a particular multicast, the event is unicast automatically, as long as you have not disabled the failover feature.

Configuring Back-Channel Multicasting

The following procedure describes how to set up Helix Server for back-channel multicasting. Minimally, you need to define your multicast address range. Other features are optional.

➤ To set up back-channel multicasting:

1. In Helix Administrator, click **Broadcast Distribution> Back-Channel Multicasting**.
2. The **Enable Multicast** list turns on this feature. Ensure that the default value of `Yes` is selected. If you set this to `No`, multicasting is disabled, and all media players use unicasting for all broadcasts.
3. Set the **Enable SAP** list to `Yes` to announce the multicast session, as described in "Publicizing Multicasts" on page 175.
4. In the **RTSP Port** box, list the port number on media clients where Helix Server directs RTSP multicast streams. The default is `3554`.
5. Specify the range of IPv4 addresses to which you want to multicast streams by filling in the **IP Address Range** box. Helix Server uses the first available addresses in this range. See "Calculating Address Requirements" on page 162 for information about the number of addresses you'll need.
6. Indicate how far multicast packets can travel over a network by typing a value in the **Time to Live** box. For more information, see "Packet Time to Live" on page 159.
7. To allow missing packets to be resent to clients that request them, select `True` from the **Resend** list. Resending packets adds network overhead, but delivers higher-quality multicasts.
8. If you want to deliver the broadcast through multicasting alone, choose `Yes` from the **Multicast Delivery Only** list. Media players that cannot use multicasting will not be able to connect to the broadcast. Use this feature when broadcasting only to multicast-enabled media players, or if you are multicasting a high-bandwidth presentation and do not want to provide a unicast option.

   Warning! Selecting this option prevents you from unicasting any broadcast to RealNetworks media players. If you want to reuse unicasting after your multicast, turn this option off when the multicast ends.

9. The **Access Rules** section lets you restrict the range of media players that can connect to the multicast. A predefined rule allows all multicast-enabled players with access to the broadcast URL to connect to the

multicast. You can delete this rule, modify it, and set up other rules as necessary:

a. To add a new rule, click the "+" icon and, optionally, change the default name in the **Edit Client Access Rule Description** box. Because the access rules simply list addresses of media players that have access, the order of the rules in the **Access Rules** box does not matter.

b. For the highlighted rule, enter an IPv4 address or domain name for acceptable players in the **Client IP Address or Hostname**. The value `Any` is predefined for the first rule, meaning that all IPv4 addresses are accepted. To restrict the range, delete the rule, or change its value and set a netmask.

c. In **Client Netmask**, specify the range of client IPv4 addresses around the one you entered in the preceding step by selecting a bit mask. If **Client IP Address** is set to `Any`, though, leave **Client Netmask** set to `None`. See Appendix B for details about assigning a range of IP addresses using a bit mask.

10. Click **Apply**.

Tip: General access control rules, which are described in Chapter 12, are enforced before multicast access rules. A media player excluded by general access control will not connect to any multicast, regardless of the multicast access rules.

Starting a Back-Channel Multicast

After you configure a back-channel multicast, you start a unicast as described in Chapter 7. Media players that can connect to the multicast do so. Players that cannot connect to the multicast use a unicast, as long as the failover feature is not disabled. Links to multicasts are identical to links for unicasts. This enables a single link to serve both multicast and unicast clients. For information on linking formats, see "Linking to Unicasts" on page 151.

Setting Up Scalable Multicasting

This section describes how to set up a live channel to broadcast to any number of media players that support the RTP packet format and the scalable multicasting standards. This includes RealPlayer and Apple QuickTime Player.

You can multicast any audio or video format that plays in the media players used by your audience.

> **For More Information:** The section "Multicast Resources" on page 161 directs you to information about multicasting standards.

Determining the Number of Addresses and Ports

You must reserve a contiguous block of IPv4 addresses and a continuous range of Helix Server ports for a scalable multicast. The number of addresses and ports you need is based on the number of streams you broadcast. The following sections describe how to calculate the numbers of addresses and ports you'll need.

> **Note:** The multicast addresses you choose cannot overlap with addresses assigned to back-channel multicasts.

Single-Rate Audio and Video

A single-rate audio broadcast uses one stream, requiring one address. A single-rate video contains at least two streams, one for the audio track and one for the visual track. It may also have an event stream used to open URLs, for instance. The scalable multicasting specification requires an address for each stream. However, Helix Server has a default **Reuse Address** feature that lets you use just one multicast IP address for each single-rate video, whether or not it includes an event stream. RealNetworks recommends that you use this feature unless you have a specific reason not to do so.

> **Note:** The **Reuse Address** feature works only with RealNetworks media players.

> **Tip:** Using multiple addresses is useful if media players use a low-bandwidth connection, and are able to play just the audio track of a video stream.

SureStream RealAudio and RealVideo

For SureStream, you need to reserve one address for each encoded bit rate. If a SureStream RealAudio broadcast is encoded for three bandwidth targets, for example, you need three addresses. If you broadcast RealVideo and are not using the **Reuse Address** feature, you need two addresses for each rate. If a SureStream RealVideo broadcast is encoded for three audience bandwidth

targets, for example, you need three addresses if the **Reuse Address** feature is used, six addresses if it isn't. The duress streams encoded for a particular bandwidth target are not used, and do not require additional addresses.

> **Note:** Media players cannot shift between SureStream streams during a multicast.

Port Ranges

Scalable multicasts require a continuous, even-numbered range of ports. The number of ports required depends on the media type, and the number of bit rates encoded in the stream. You need to reserve two ports for each data stream:

- For video, reserve four ports for each encoded bit rate (two ports for each audio track, and two for each visual track) regardless of whether you use the **Reuse Address** feature. SureStream RealVideo with three encoded bit rates, for instance, needs 12 data ports.
- For audio, reserve two ports for each encoded bit rate. For SureStream RealAudio with three encoded bit rates, for example, you need 6 ports for data.
- An audio or video stream may also have an events stream, which requires an additional two ports. With RealAudio or RealVideo, for instance, an event stream can automatically open URLs in the viewer's Web browser.

> **Tip:** You can determine a clip or unicast stream's encoded bit rate (or rates) by playing the clip or live stream in RealPlayer, and giving the **View>Clip>Clip Source** command.

Gathering Client Statistics

RealNetworks media players can transmit statistics about the amount and quality of data they received during a scalable multicast. As with unicasts, client statistics are sent at the end of a presentation (or when the player disconnects), and are stored in the basic access log described in Chapter 15. Once you have gathered statistics, you can draw conclusions about the quantity of clients and quality of service.

Because scalable multicasts can serve many thousands of players, your Helix Server may not be able to handle all of the statistics connections at the end of the session, either because of load or client licensing limitations. In this case,

you can limit the amount of information sent to Helix Server (though not the number of connections), or offload the task to a Web server.

Limiting Statistics

Under **Logging & Monitoring>Access & Error Logging**, you can control the type of data sent at the end of a scalable multicast by modifying the **Client Stats** setting. You can instruct clients to send minimal data by setting **Logging Style** to `0` and by clearing any **Client Stats** settings. The section "Logging Style" on page 326 shows the data recorded with this logging style. Although it records minimal data, Helix Server still accepts connections from all players, up to its licensed maximum.

> **Note:** If you change the access logging style, the setting applies to all content served through any method.

Offloading Statistics to a Web Server

When players connect to a scalable multicast, Helix Server can instruct them to send their connection statistics to a Web server at the end of the session. Web servers are often configured for load balancing, and may be better equipped than Helix Server to handle large numbers of simultaneous connections that have short durations.

To use a Web server, you will need to write a CGI script that receives the statistics and to writes them to a log file. The statistics sent to the Web server are a subset of those normally recorded in the Helix Server basic access log. They are transmitted through HTTP `POST`, and they use the following format, in which the # symbol is a separator:

`[Stat1:`*`statistics_1`*`][Stat2:`*`statistics_2`*`]#`*`sent_time`*

> **For More Information:** See "Client Statistics" on page 340 for details about client statistics 1 and 2.

Setting Up a Live Channel

The following procedure explains how to set up a live channel for delivering a scalable multicast to RealNetworks media players. You might set up just one live channel for all scalable multicasts. Alternatively, you can create different live channels to multicast broadcast streams in different ways. To run two multicasts at the same time, for example, you define two separate channels.

➤ To create a live channel:

1. In Helix Administrator, click **Broadcast Distribution>Scalable Multicasting**.
2. The **Mount Point** box lists the mount point used for all scalable multicasts. The default mount point is `/scalable/`. RealNetworks recommends that you use this default value, but you may change it if you wish.
3. To create a new live channel, click the "+" icon and enter a descriptive name for this multicast session in the **Edit Channel Description** box.
4. Turn on scalable multicasting for this channel by selecting `Yes` from the **Enable Channel** list.
5. Set the **Enable SAP** list to `Yes` to announce the multicast session, as described in "Publicizing Multicasts" on page 175.
6. In the **Path** box, identify the stream name the live broadcast uses. To accept all broadcast streams, keep the default value of an asterisk (`*`). Otherwise, specify a stream name, such as `live.rm`, and include any virtual path used by the encoder, as in `news/live.rm`. You do not need to include the encoder's unicast mount point, such as `/broadcast/` or `/encoder/`.
7. You next specify the ports and addresses used by the scalable multicast. Refer to "Determining the Number of Addresses and Ports" on page 165 for more information on these topics.
   a. In the **Port Range** boxes, type the port range where clients listen for streams.
   b. In the **IP Address Range** boxes, type the range of addresses to use. Helix Server uses the first available address in this range. To use a single address instead of a range, type the same address in each box.

      Tip: You can use the same range of ports and addresses for multiple channels. If you plan to multicast on multiple channels at the same time, however, ensure that your ranges are broad enough to cover all of the simultaneous multicasts. Although two channels can share the same range of addresses, for example, they cannot both broadcast on the same address simultaneously.
8. Indicate how far multicast packets can travel on your network in the **Time to Live** box. For information on these values, see "Packet Time to Live" on page 159.

9. Type a value for **Timeout**. This represents the number of seconds a client waits for multicast packets before it stops, or uses the value in **Alternate URL**.

10. From the **Reuse Address** list, select `False` if you want to use a separate address for each stream in a video. Select `True` if you want to use one address for both streams. Refer to "Determining the Number of Addresses and Ports" on page 165 for more information.

    Note: The **Reuse Address** feature works only with RealNetworks media players.

11. You next decide whether to shift media players that cannot receive the multicast to a unicast. This feature works only with RealNetworks media players.

    a. From the **Shift to Unicast** list, select `No` if you do not want to shift clients to unicasting. You may want to choose this when broadcasting a high bit rate presentation in which a lot of unicast streams would consume too much bandwidth. If you choose `No`, you can ignore the next box.

    b. To make the backup unicast available on the same Helix Server, leave the **Alternate Unicast URL** box blank. If you want to shift unicasts to a different Helix Server, supply that server's address and the path to the broadcast. Here is an example:

    ```
    rtsp://helixserver.example2.com/broadcast/vivaldi.rm
    ```

    If you do not want to redirect players to an alternate stream, you can point them to a Web page that posts a message, such as, "This presentation is available only to multicast-enabled RealPlayers." In the **Alternate Unicast URL** box, enter the fully qualified URL of your Web page, such as the following:

    ```
    http://www.example.com/no_multicast.html
    ```

12. The next set of options controls whether client statistics are logged on Helix Server, on a Web server, or not at all. For background on this feature, see "Gathering Client Statistics" on page 166. Only RealNetworks clients report statistics, so you can set this to `No` if multicasting to other RTP-based media players.

    a. In the **Send Client Statistics to Web Server** box, select `Yes` to have clients send their connection statistics to a Web server at the end of the

multicast. Select `No` to send connection statistics to Helix Server. If you select `No`, you can ignore the following settings.

b. In the **Web Server Address or IP Address** box, type the address of the Web server that collects the statistics.

c. In the **Web Server Port** box, type the HTTP port number of the Web server.

d. In the **Web Server CGI Path** box, type the path of the CGI script that consolidates the client statistics into a log file on the Web server. For example, enter:

```
cgi-bin/client-stats/logstat
```

13. Click **Apply**.

Delivering the Scalable Multicast

Once you've configured the multicast, you launch your broadcast as described in Chapter 7. If you leave scalable multicasting turned on, and configure your multicast to cover all broadcasts by entering "*" as the path, the multicast will be available for all unicasts intended for RealNetworks media players. Unlike back-channel multicasts, however, scalable multicasts use a different URL format than unicasts. This means that the scalable multicast is available only if you provide the proper multicast link.

> **Tip:** If you've enabled the **Shift to Unicast** feature, you can publish just the multicast URL for all broadcasts. Players that can connect through multicasting will do so. Players that cannot use multicasting will use unicasting.

Multicast SDP Files

When Helix Server receives a scalable multicast request, it automatically creates an SDP (Session Description Protocol) file, which is a standard format that contains information such as the multicast addresses and ports, as well as the broadcast stream's title, author, and copyright information.

The media player receives the SDP file when the viewer clicks the multicast link in a Web page, and connects to the multicast using the file's information. A viewer can also download the SDP file, typically by right-clicking on the Web page link, then connect to the multicast later by opening the file directly in the media player.

Linking to a Scalable Multicast

Scalable multicast links differ from unicast links. They use the same format whether they appear in a Web page or in a metafile, and they always specify the HTTP protocol. The requested stream always ends with the .sdp extension. The following illustrates the link format:

http://*address:port/mount_point/path/file.rm*.sdp

Here is an example in which Helix Server uses its default port 80 for HTTP, so the port number is not listed:

http://helixserver.example.com/scalable/news/live.rm.sdp

- The default mount point is /scalable/, but you can change this as described in "Setting Up a Live Channel" on page 167. The /ramgen/ parameter is not required, even though HTTP is used.
- A virtual path name such as news/ is optional. You include it only if the encoder specifies this path when it delivers the stream.
- The requested file includes the stream name specified for the broadcast by the encoder, such as live.rm. You then append an extra .sdp extension. Even though MPEG unicasts link to an SDP file, as in live_mp4.sdp, the multicast requires an extra .sdp extension in the request URL, as in live_mp4.sdp.sdp.

> **Tip:** The SDP file contains exact channel settings. If you plan to repeat a multicast, and you want users to connect through a saved SDP file, use the same encoder settings, addresses, and ports as in the preceding multicast.

> **Note:** For players to receive the broadcast and log statistics, /scalable must be in the HTTP delivery list, which is described in "Allowing HTTP Delivery" on page 68. Enabling scalable multicasts adds this mount point to the list automatically.

Multicasting Windows Media

As with all Windows Media broadcasts, a Windows Media multicast is delivered by Windows Media Encoder. You need only one IP address and port for each multicast. This style of multicasting builds on the basic unicasting definitions you set up according to instructions in "Broadcasting Windows Media" on page 143.

Defining a Multicast Channel

To set up Windows Media multicasts, you first define one or more multicast channels. A multicast can go out on a channel as long as the channel is enabled.

➤ To create a Windows Media multicast channel:

1. In Helix Administrator, click **Broadcast Distribution>Windows Media Multicasting**.
2. The **NSCFilePath** box lists the path to the NSC file, which Windows Media Players use to connect to multicasts. The default path is the `nscfile` directory under the `Content` directory, but you can change this to another directory that holds on-demand content. Helix Server automatically generates the NSC file in the specified directory, using the channel name as the file name and `.nsc` as the extension.

   Note: If you use a different directory for the NSC file, add that path to the HTTP delivery list, as described in "Allowing HTTP Delivery" on page 68.

3. To create a new live channel, click the "+" icon in the **Channels** section and enter a descriptive name for this multicast session in the **Channel Name** box. Because the channel name becomes the NSC file name, RealNetworks recommends that you do not include spaces or special characters in the name.
4. Enable multicasting for this channel by selecting `Yes` from the **Enable Broadcast** list. After the multicast, you can disable the channel by resetting this list to the default value `No`.

   Tip: In a pull broadcast, the Windows Media multicast is pulled on request. Because viewers can save the NSC file locally, it's a good idea to disable a channel when you do not intend to use it.

5. In the **Multicast Address** box, type the class D multicast address to use. You need only one address for each multicast. Each channel must use a different address.
6. In the **Port Range** boxes, type the port where clients should listen for streams on this channel. Each channel must use a different port.

7. Indicate how far multicast packets can travel on your network in the **Time to Live** box. For information on these values, see "Packet Time to Live" on page 159.
8. In the **Stream Description** box, you can type any text that describes this multicast stream. This is for your reference, and is not used in the multicast.
9. To make a backup unicast available for media players that cannot join the multicast, supply the address and path to the unicast in the **Alternate Unicast URL** box. Although the unicast can be on the same Helix Server, you need to define the address explicitly for the unicast failover to work. Here is an example:

   ```
   mms://helixserver.example.com/wmtencoder/chopin.wma
   ```

10. For **Client Statistics URL**, supply an HTTP address or host name for the Microsoft Internet Information Server (IIS) that gathers client statistics at the end of the multicast. If you leave this field blank, media players do not send back any statistics. Players cannot send statistics to Helix Server.

    > **For More Information:** Refer to the Windows Media Services documentation on reporting multicast statistics for instructions about setting up IIS to gather statistics.

11. Proceed to the next section if you want to define the live sources that use this multicast channel. You can click **Apply** to save the channel definition, however, if you want to set up its channel sources later.

Setting Up a Live Source

Each channel can multicast a live stream from any number of Windows Media Encoders. However, it can multicast a stream from just one of its live sources at a time. Below a channel definition, you set up the live sources that each channel can use. Each channel maintains a separate list of sources.

➤ To set up a live source for a channel:

1. In Helix Administrator, click **Broadcast Distribution>Windows Media Multicasting**.
2. In the **Channels** box at the top of the page, highlight the channel for which you want to create a live source.

3. Scroll to the bottom of the page and click the "+" icon in the **Channel Sources** section to set up a new live source.
4. Enter a descriptive name for this source in the **Channel Source Name** box. This name is for your reference only, and is not used in the multicast.
5. For **Stream Format File**, enter the full path and file name of the ASF file (extension `.asf`) that Windows Media Encoder generates and is copied to Helix Server when the encoding begins. This file is used only by Helix Server, and is not delivered to media players.

   Tip: Because ASF is a streaming format, RealNetworks recommends that you do not place the format file in a directory used for on-demand streaming. This helps you to avoid mistaking the format file for a streaming clip.

6. For **Live Source**, specify the mount point and stream name that identifies the live stream pulled from Windows Media Encoder. These elements are defined in the Windows Media unicast page, described in "Broadcasting Windows Media" on page 143. The default mount point is `/wmtencoder/`, so a **Live Source** entry might look like this:

   ```
   /wmtencoder/live.wmv
   ```

   Warning! You must define a specific broadcast stream for each live source. Windows Media multicasting does not support the use of a wildcard character (such as "*") to pull all broadcasts that use the `/wmtencoder/` mount point.

7. Click **Apply** to save the live source definition.

Running the Windows Media Multicast

Once you've enabled the multicast and defined the live source, you are ready to begin the multicast. Start Windows Media Encoder running, and deliver the ASF stream format file to the predefined location on Helix Server. In a pull broadcast, the broadcast starts on the first media player request.

Linking to the Multicast

Windows Media Players join the broadcast by requesting a download of the NSC file, which then instructs them on how to join the multicast. You can launch the broadcast through an HTTP link in a Web page, as shown in the

following example. The /asxgen/ mount point normally required to launch Windows Media Player is not required:

```
http://helixserver.example.com/nscfile/live_channel.nsc
```

> **Tip:** If you're using the unicast failover option, you can publish just the NSC file URL. Players that can connect through multicasting will do so. Players that cannot use multicasting will switch to the unicast URL.

Stopping a Windows Media Multicast

As a channel multicasts the stream, the channel name appears in the **Transmitting Channels** box at the top of the multicasting set-up page. To stop the broadcast, highlight the channel name, check the **Stop Transmitting** box, and click **Apply**. This changes the **Enable Channel** list to No to prevent the stream from being pulled again. To restart the channel, select Yes in the **Enable Channel** list and click **Apply**.

Publicizing Multicasts

Optionally, you can publicize back-channel and scalable multicasts to anyone running a program that listens for the Session Announcement Protocol (SAP). These applications, such as SDR and ICAST Guide, display a list of all multicasts currently playing. Helix Server creates the SAP file automatically. Programs that listen for SAP announcements show the title, author, and copyright information encoded into the files you multicast.

> **Note:** The Windows Media multicast format does not support Session Announcement Protocol.

➤ To set up Helix Server to create SAP files:

1. In Helix Administrator, click **Broadcast Distribution>Session Announcement**.
2. In the **Host IP Address** box, type the IPv4 address of this Helix Server. The SAP announcement will include this address.
3. From the **Enable SAP Service** pull-down list, select Yes. This instructs Helix Server to create and send SAP files. The default value is No.
4. From the **Listen to SAP** list, ensure that the default value of Yes is selected. This option enables Helix Server to collect in-use multicast addresses.

Helix Server consults this list when selecting a multicast address from the user-supplied address range, thus ensuring that it selects unique addresses that are not in use elsewhere on your network.

5. Click **Apply**.

CHAPTER

9

TRANSMITTERS AND RECEIVERS

Transmitters and receivers are two components of *splitting*, which enables you to deliver broadcast streams in any format to multiple Helix Servers. The servers then unicast or multicast the stream to media players. This chapter describes different splitting arrangements, then explains how to set up Helix Server as a transmitter, a receiver, or both.

Understanding Splitting

At its most basic level, splitting is a technology that enables one component to deliver a live or simulated live broadcast stream to another component. This allows an encoder to transmit a stream to one or more servers, for example, and a server to distribute a stream to one more additional servers. You can thereby deliver single broadcasts to many more users than you could with a single Helix Server. As a result, splitting fosters higher-quality broadcasts by distributing broadcast streams closer to viewers.

> **Note:** Splitting builds on the foundation of unicasting described in Chapter 7. Before you configure a splitting arrangement, be sure that you understand the concepts and procedures described in that chapter.

> **Tip:** The section "SLTA Quick Start Tutorials" on page 218 introduces you to transmitters and receivers by explaining how to simulate a live broadcast on your Helix Server.

Definitions

Because splitting technology has many uses and components, it is important to understand the following general terms.

Encoder
: An encoder is software that generates a live or simulated live stream. For RealMedia, an encoder can be RealProducer or the simulated live transfer agent (SLTA). Additional encoders that you can use include Windows Media Encoder and Sorenson Broadcaster.

Transmitter
: A transmitter is a Helix Server on which a stream originates. The transmitter sends the stream to a receiver, and can also broadcast the stream directly to media players. RealProducer and SLTA can also function as transmitters. They send a stream only to a Helix Server receiver, however, and not to individual media players.

Receiver
: A receiver is a Helix Server that receives a stream from a transmitter and broadcasts it to media players.

Relay
: A relay is a Helix Server that functions as both a receiver and a transmitter. It receives a stream from another source, and transmits that stream to another receiver. It may also broadcast the stream directly to media players.

Push Splitting
: In push splitting, a transmitter initiates the splitting session by delivering the broadcast stream to one or more receivers.

Pull Splitting
: In pull splitting, a receiver initiates the splitting session by contacting a transmitter and requesting the stream.

Encoder-to-Server Splitting

You can use splitting technology to deliver a live or simulated live broadcast stream from an encoder to Helix Server. Although this arrangement involves splitting, it does not necessarily involve multiple Helix Servers. Helix Server can acquire a stream in many ways, not all of which use splitting. The following

table explains the various methods of encoder-to-server delivery, indicating whether each method is based on splitting technology.

Splitting Used in Server Stream Acquisition

Stream Delivery Method	Split?	Notes
Live stream using RealProducer push or pull mode.	**yes**	The push and pull modes in RealProducer both involve splitting, in which RealProducer is the transmitter and Helix Server is the receiver.
Simulated live stream using SLTA.	**yes**	In its advanced mode, SLTA, which Chapter 10 covers, functions as a transmitter, and Helix Server acts as a receiver.
Live stream using RealProducer in account-based mode.	no	Although account-based mode is a subset of push mode, it does not use splitting. Instead, it uses the stream delivery technology of earlier versions of RealProducer. To broadcast in this mode, follow the instructions in "Setting Up Account-Based Broadcasting" on page 140. You do not need to set up Helix Server as a receiver.
Live stream using RealProducer G2 through 8.5.	no	Earlier versions of RealProducer do not use splitting technology. The section "Encoding with an Older Version of RealProducer" on page 142 explains how to use them to deliver a stream to Helix Server. You do not need to set up Helix Server as a receiver.
Live or simulated live stream using any other media encoder.	no	Only RealNetworks encoders use splitting technology. Encoders for other media formats, such as Windows Media and QuickTime, deliver streams to Helix Server by other means. See Chapter 7 for more information. You do not need to set up Helix Server as a receiver.

No matter how Helix Server acquires a broadcast stream, it can use splitting technology to transmit that stream to other Helix Servers. Keep in mind, therefore, that encoder-to-server and server-to-server splitting are different functions, and one does not require the other. This enables you to split Windows Media or QuickTime broadcasts, for example, even though the encoders are not based on RealNetworks splitting technology.

Server-to-Server Splitting

In server-to-server splitting, a Helix Server transmits a live or simulated live stream to another Helix Server. The transmitter may have acquired the stream

through a splitting set-up with a RealNetworks encoder, or through a conventional broadcast method. This allows you to distribute the same stream to multiple Helix Servers across your network. You can use splitting technology to tie together a cluster of Helix Servers on an intranet, as well as to connect different servers around the globe over the Internet.

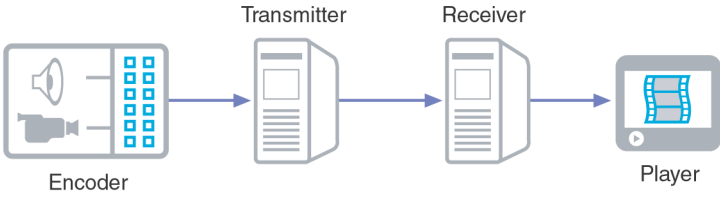
Splitting offers many features that help you to tie together large networks of Helix Servers, conserve bandwidth, and transmit different streams to different receivers. The following are the major features of server-to-server splitting, which subsequent sections explain in detail:

- **push and pull splitting**—A splitting configuration can use push splitting, pull splitting, or both.
- **unicasting and multicasting**—Transmitters can unicast or multicast to receivers. You can also combine methods to unicast a stream across the Internet, then split it by multicasting behind a firewall.
- **single transmitters or chained relays**—One transmitter can split a stream to multiple receivers. Or, each Helix Server can relay the stream to the next Helix Server. You can combine these methods to tie together a large network of servers in any configuration.
- **SureStream-aware splitting**—When you broadcast RealMedia, SureStream-aware splitting saves bandwidth by sending only the encoded streams that have been requested.
- **multiple splitting definitions**—You don't have to split each broadcast the same way. Each Helix Server can have multiple transmitter and receiver definitions. This lets you push certain broadcasts, for example, and pull others. Streams can originate anywhere on your network.
- **redundancy**—Splitting fully supports encoder redundancy for any media type. In addition, it lets you add server and stream redundancy.

Push Splitting

The following illustration shows the conventional, or "push" form of server-to-server splitting. Here, the transmitter initiates the connection to the receiver. When a media player requests the broadcast, the receiver is ready to deliver the stream. In this form of splitting, Web page links typically point to the receiver. A transmitter can also deliver streams to media players, however.

Push Splitting



Pull Encoding with Push Splitting

In push splitting, the encoder typically pushes the broadcast stream to the transmitter, initiating the broadcast. However, the transmitter can pull the stream from the encoder, too. This lets you use push splitting with pull encoding methods. To do this, you might create a private link to the broadcast on the transmitter. When you're ready to push the stream to the receiver, you click this link, which causes the transmitter to pull the stream from the encoder, and push it to the receiver. Viewers can then connect to the broadcast on the receiver.

Push Splitting Configuration

You do the following to configure push splitting:

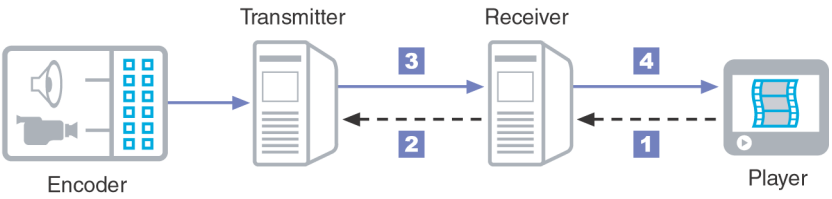
- Set up the originating Helix Server as a push transmitter as described in "Configuring a Pull Splitting Transmitter" on page 204. Ignore the pull-splitting configuration fields.
- Define one or more Helix Servers as receivers as described in "Setting Up a Receiver" on page 205. Ignore the pull-splitting configuration fields.
- Set up your encoder to deliver the broadcast stream as appropriate for that encoder. With RealProducer, you can use push or pull delivery, though push is more appropriate for this arrangement as it allows you to start the entire broadcast directly from the encoder.
- Create links to the broadcast as described in "Linking to Split Content" on page 208.

Pull Splitting

The following illustration shows pull splitting, in which the transmitter does not deliver the stream to the receiver until the first media player makes a request (step 1). There's a slight delay as the receiver requests (step 2), receives (step 3), and delivers (step 4) the stream. After that, the stream is live on the

receiver, and subsequent player requests do not involve the session setup delay of step 2.

Pull Splitting



For More Information: For details about delays caused by pull splitting, see "Stream Acquisition Latency for Pull Splitting" on page 196.

Pull Splitting Bandwidth Efficiency

Although pull splitting results in greater latency than push splitting on the first broadcast request, it can save on bandwidth because the stream is not transmitted to the receiver if no media player requests the stream. As well, if all media players disconnect from the receiver before the broadcast ends, the data stream between transmitter and receiver is dropped. Hence, pull splitting never consumes bandwidth between transmitter and receiver if no one is viewing the broadcast on that receiver.

Tip: You can combine push and pull splitting for optimal results. Suppose that you are delivering a broadcast across many different time zones. You could push the stream to receivers that reside in daytime zones. Where it's late at night and there are fewer potential viewers, you could have receivers pull the stream only on viewer request.

Push Encoding with Pull Splitting

If you set up pull splitting between transmitters and receivers, you can still use RealProducer or SLTA to push the stream to the transmitter. This cues the broadcast so that the transmitter can respond faster to pull splitting requests from receivers.

Pull Splitting Configuration

You do the following to configure pull splitting:

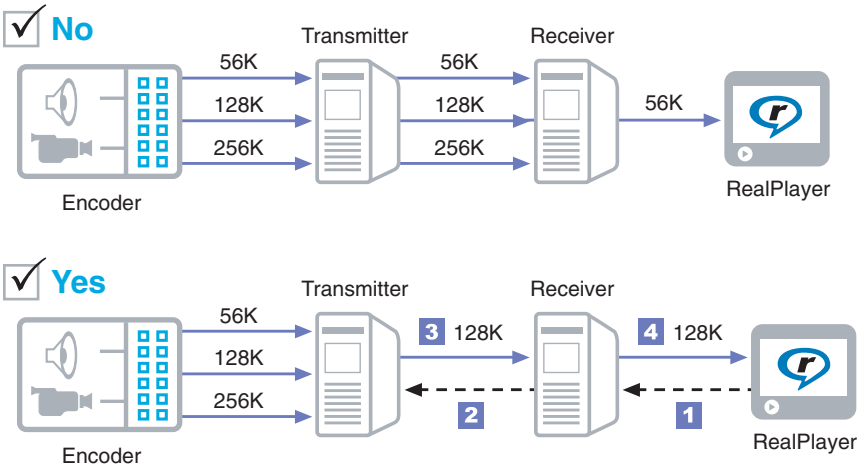
- Set up the originating Helix Server as a transmitter as described in "Configuring a Pull Splitting Transmitter" on page 204.
- Define one or more Helix Servers as receivers as described in "Setting Up a Receiver" on page 205. Enable the receiver for pull-splitting.
- Set up your encoder to deliver the broadcast stream as appropriate for that encoder. With RealProducer, you can use push or pull delivery.
- Create links to the broadcast as described in "Linking to Split Content" on page 208.

SureStream-Aware Splitting

SureStream-aware splitting is similar to pull splitting when you broadcast RealMedia streams encoded at different bit rates. Without SureStream-aware splitting, the transmitter sends all streams encoded in the broadcast to the receiver. In the following illustration, the top portion illustrates a broadcast that is not SureStream-aware. The broadcast itself may be either push or pull. When the receiver acquires the broadcast stream, it gets all of the SureStream bit rates, even if only the 56 Kbps stream has been requested by media players.

In contrast, SureStream-aware splitting sends each stream only on request. The bottom half of the following illustration shows a player requesting the 128 Kbps stream. Even though the encoder sends the transmitter all the streams, only the 128 Kbps stream goes to the receiver. If another media player later requests the 56 Kbps stream, the transmitter forwards that stream, too. If a certain stream is never requested, it is never sent to the receiver. As well, the transmitter drops a stream if media players stop using it during the broadcast.

SureStream-Aware Splitting Enabled from Transmitter to Receiver



SureStream-aware splitting works with both push splitting and pull splitting arrangements. Even if your entire network is set up for push splitting, receivers are sent individual SureStream streams only on request. Like pull splitting, SureStream-aware splitting helps to conserve bandwidth during broadcasts, but is not available when transmitters and receivers communicate through multicasting.

> **Tip:** To archive a SureStream-aware broadcast, set up your archive on the Helix Server connected to the encoder. This is the only server guaranteed to receive all SureStream streams. For more on archiving, see "Archiving Broadcasts" on page 131.

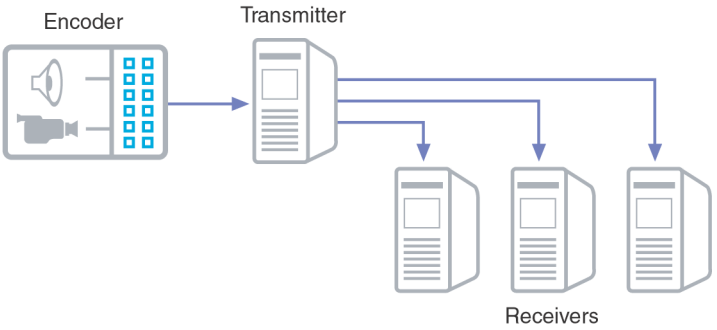
Complex Splitting Arrangements

You can combine the basic splitting methods (push and pull) with the two transport methods (unicast and multicast) in many ways. Splitting also supports encoder redundancy, and can add failover protection for streams and transmitters. The following sections describe different splitting setups, their benefits, and drawbacks. Keep in mind that you can create different splitting arrangements throughout different parts of the network to achieve your goals, whether that's to maximize bandwidth efficiency, or to reduce broadcast latency.

One-to-Many Splitting

A common splitting arrangement uses a single transmitter to broadcast to multiple receivers. If you do this through unicasting, each receiver gets a unique stream, so bandwidth consumption increases with each receiver. Multicasting uses less bandwidth, and is a better solution if all components are on a multicast-enabled network. The following illustrations shows unicasting through push splitting, though you can also use pull splitting, in which each receiver connects to the transmitter only when it needs the stream. Server-to-server multicasting is not available with pull splitting, however.

One-to-Many Splitting

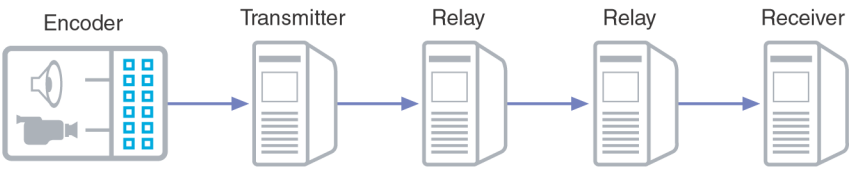


Because RealProducer and SLTA use splitting technology, they can function as transmitters connecting to multiple Helix Server receivers. A Helix Server is therefore not required to be the transmitter. For a live broadcast, though, you may not want to use RealProducer to transmit the stream to many receivers. Real-time encoding of a live event uses a lot of processor power, and it's better not to burden the encoder with stream management overhead as well. This is not an issue in a simulated live broadcast using SLTA, however, because the streams are already encoded.

One-to-One Chaining

Another option is to use one-to-one chaining, in which each receiver transmits to another receiver. Receivers in the middle of the chain thereby function as relays. This option is viable if a group of servers is spread across a wide area, and uses unicasting over the Internet to communicate. A transmitter in San Francisco might push a stream to a Tokyo receiver, which pushes it to a Sydney receiver, and so on.

One-to-One Chaining

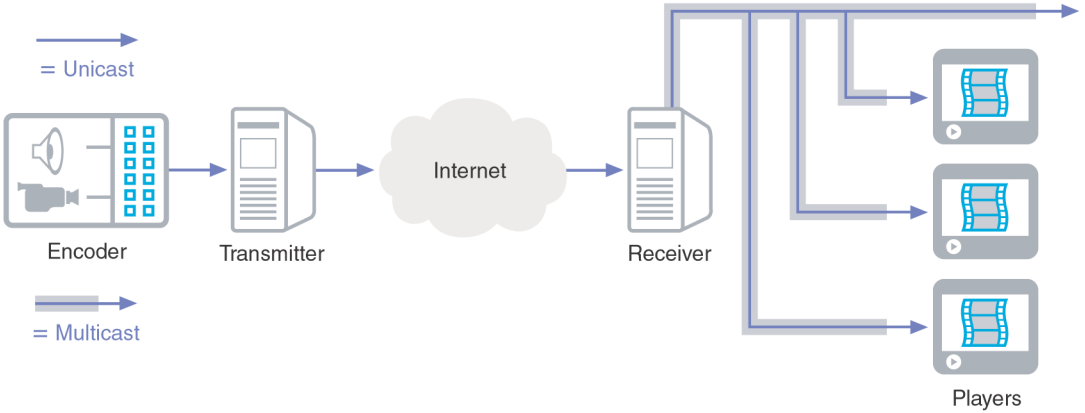


Although you can use pull splitting with a relay chain, push splitting suits this setup better. With pull splitting, there may be a long latency period if the first broadcast request comes from far down the chain. In this case, the request has to make its way back the chain, causing each receiver in the chain to pull the stream from the preceding transmitter. As well, pull splitting links for a relay chain quickly become very long, increasing your chances of writing incorrect URLs to the broadcast. Using URL aliases, though, eases this burden.

Unicast Delivery, Multicast Distribution

Splitting works with multicast delivery, as well as unicasting. You can unicast streams to receivers across the Internet, an intranet, a wide area network, or a local area network, for example. Then, within an intranet, you can multicast the stream from the receivers to media players. The following illustration shows this type of delivery.

Unicast Delivery, Multicast Distribution

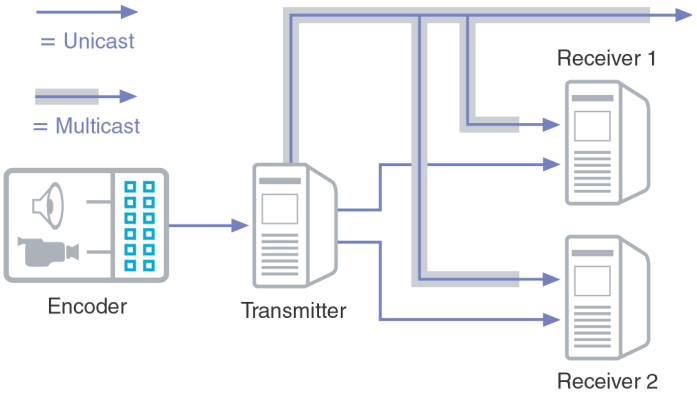


Dual Unicast and Multicast Transport Methods

If your server network is multicast-enabled, you can simultaneously unicast and multicast a broadcast stream to receivers. Duplicate packets arriving by

different transport methods increase the network overhead, but do not cause problems for receivers. When a receiver reassembles the broadcast stream, it uses the packets that arrive first, regardless of their transport methods. If a unicast packet is late or missing, for example, the receiver may get the right packet through multicast. The following figure illustrates this dual delivery.

Broadcasting by Multiple Methods

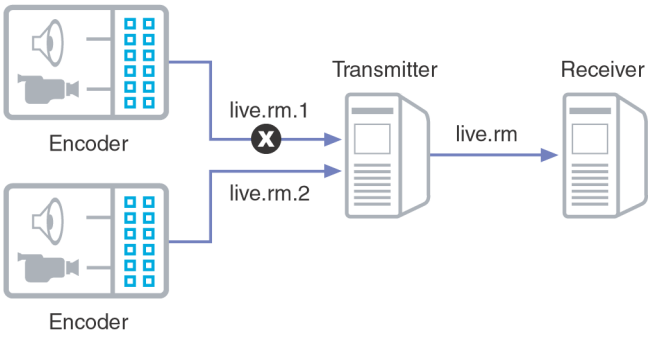


Tip: Server-to-server multicasting requires a multicast-enabled network just like server-to-player multicasting. However, you do not configure server-to-server multicasts as described in Chapter 8. You simply select multicast as your transport method when configuring transmitters and receivers.

Redundant Stream Splitting

As explained in "Using Broadcast Redundancy" on page 135, Helix Server supports encoder redundancy for all media formats. This redundancy carries over automatically with splitting, requiring no special configuration for transmitters and receivers. As shown in the following illustration, separate encoders deliver the same stream (delimited with an integer, as in `live.rm.1` and `live.rm.2`) to the transmitter. If the primary stream fails, the transmitter uses the backup stream. It sends the receiver one stream, which may come from either the primary or the backup encoder.

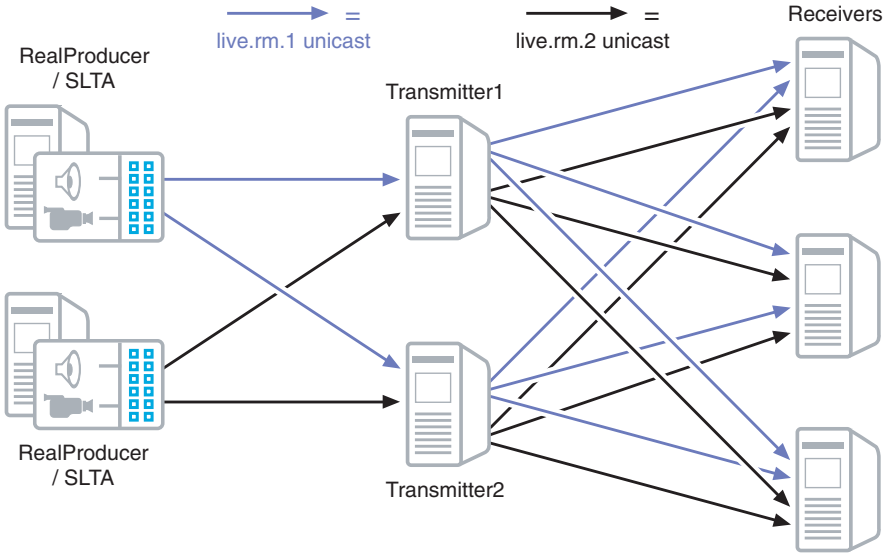
Simple Redundant Source for Splitting



Transmitter Redundancy

The setup in the preceding illustration provides a single level of encoder redundancy. You can increase redundancy within a splitting arrangement by adding transmitter redundancy, as shown in the following illustration. Here, both the primary and backup encoders send streams to two Helix Servers that each split the stream to three receivers. (RealProducer and SLTA can push the same stream to multiple Helix Servers.) Under normal conditions, each receiver gets a version of `live.rm` from each transmitter.

Redundant Sources and Redundant Transmitters

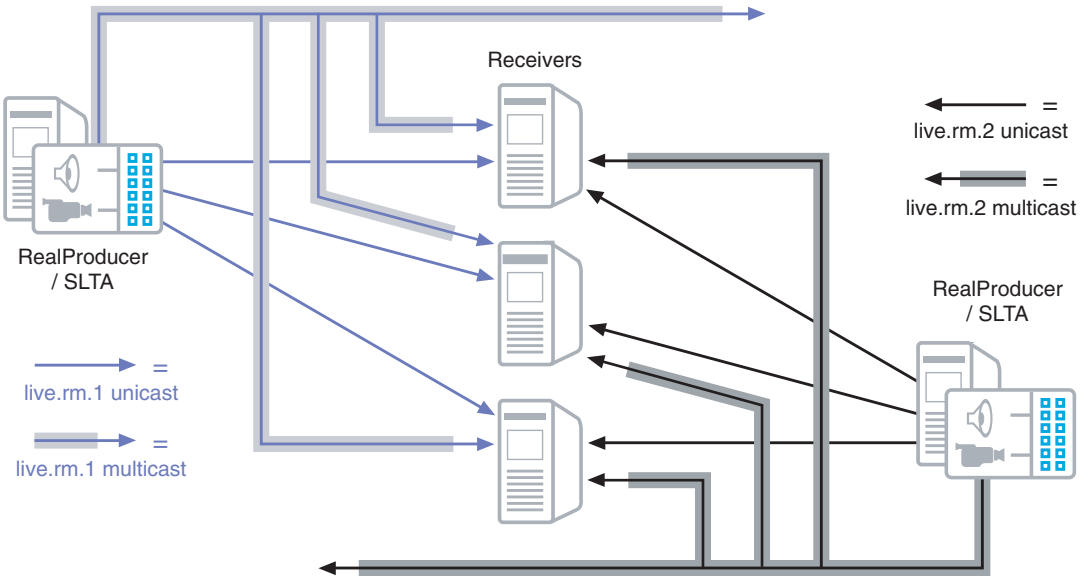


If the primary encoder in the preceding illustration fails, both transmitters switch to the stream from the backup encoder. Again, the receivers get two streams, one from each transmitter. Note that in this configuration, each receiver still receives a stream even if one encoder *and* one transmitter fail. This provides both encoder and transmitter redundancy. Three out of the four encoder and transmitter components would have to fail for the entire broadcast to fail.

Transport Redundancy

The following illustration shows multiple encoders and delivery methods used to provide encoder and transport redundancy. The primary and backup encoders both unicast a separate stream to each receiver, as well as multicast the broadcast stream to all receivers. Each receiver gets four streams. It uses the primary `live.rm.1` stream as long as those packets arrive by either unicast or multicast. If the primary encoder fails, or its transport methods are blocked, each receiver switches to the `live.rm.2` backup sent over unicast or multicast.

Redundant Encoders Using Unicasting and Multicasting



Note that the receivers in the preceding illustration could also function as transmitters that split streams to other receivers, as described in "Transmitter Redundancy" on page 188. This would provide three layers of redundancy at the encoder, transmitter, and transport levels. Although such a complex

arrangement and high degree of redundancy is generally not necessary, RealNetworks components provide support for all of these layers, which you can put together as needed.

Multiple Splitting Definitions

You may set up your transmitters and receivers to split every broadcast the same way. In this case, you always use a certain Helix Server as your primary transmitter, and your other Helix Servers as receivers or relays. This is not necessary, however, and a single network of Helix Servers can support numerous splitting arrangements, enabling you to transmit from any Helix Server. As well, you can split broadcasts from a single transmitter in many different ways.

When you set up splitting, you can create multiple transmitter and receiver definitions on each Helix Server. When you set up a push transmitter, for example, you define how it connects to each receiver. For a single Helix Server receiver on one physical machine, you can create multiple receiver definitions. RealMedia broadcasts can use one definition, for example, while MPEG broadcasts use another. This lets you multicast one format, for instance, while unicasting another.

You can also use multiple receiver definitions to split broadcasts in different ways according to stream source names, which appear in links to broadcasts. A source name has three parts:

1. **mount point**—As explained in Chapter 7, every broadcast uses a mount point. RealMedia broadcasts pushed by RealProducer use the `/broadcast/` mount point, for example, while MPEG broadcasts use `/rtpencoder/`.
2. **path**—The optional path name does not necessarily correspond to an actual path. It's generally a name inserted between the mount point and the stream name to provide splitting functionality. How you define the path differs for each media format:
    - For RealMedia broadcasts, you define the path name through RealProducer.
    - For Windows Media pull broadcasts, you define the path when configuring the encoder pull session on Helix Server, as described in "Setting up a Windows Media Pull Broadcast" on page 143. When using push encoding, you define the path name through Windows

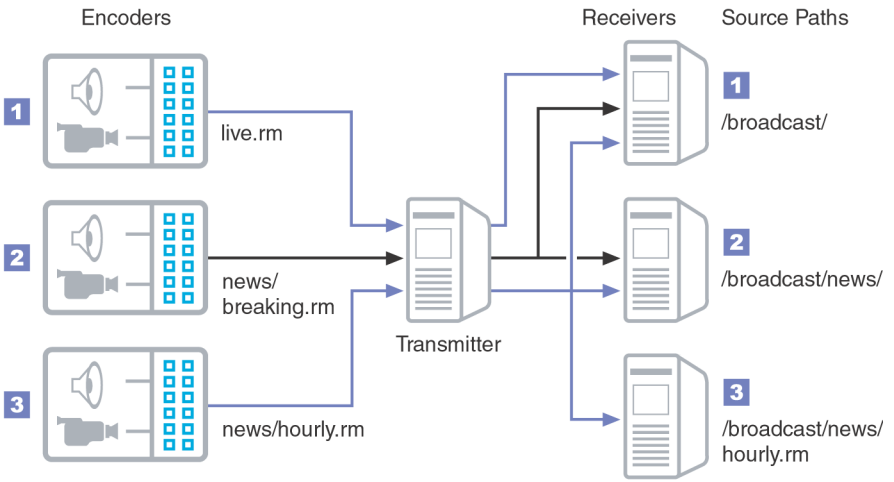
Media Encoder, as noted in "Running a Windows Media Push Broadcast" on page 147.

- For QuickTime and RTP-based broadcasts, the path name reflects the subdirectory where the encoder placed the SDP file. For more information, see "Starting an RTP-Based Broadcast" on page 150.

3. **stream name**—In all broadcasts, the stream name appears last in the URL and looks like an on-demand clip name, ending with the media format's standard file extension.

The following illustration shows how you can use mount points, paths, and stream names within receiver definitions to split different broadcasts in different ways. Three encoders connect to the same transmitter and deliver three separate streams. The first stream, `live.rm`, uses no path. The second and third streams, `news/breaking.rm` and `news/hourly.rm`, use the same path name, but different stream names.

Stream Direction through Mount Points, Paths, and File Names



Each receiver in the preceding illustration is defined with a different broadcast source path. (The definitions are actually created on the transmitter, which is responsible for directing the streams.) The first receiver accepts all broadcasts that use the broadcast mount point, `/broadcast/`. In this example, it receives all three broadcast streams because specifying just the mount point is the widest source criterion.

The second receiver gets all broadcasts that use the `/broadcast/news/` mount point and path. It therefore receives streams 2 and 3, but not stream 1, which

does not use the news/ path. The third receiver uses the most specific criteria. It receives only the broadcast streams that use the /broadcast/news/hourly.rm mount point, path, and stream name.

Splitting Considerations

This section provides general information you'll find useful when setting up any splitting arrangement. Reducing end-to-end latency may be necessary for certain live broadcasts. When you use pull splitting, you need to consider the initial latency involved in acquiring a stream. When unicasting multiple streams, bandwidth is an issue because each receiver gets a separate broadcast stream.

End-to-End Latency Reduction

During a live broadcast, a viewer experiences a latency of approximately 10 seconds between the time events occur and when they display in RealPlayer. Although this amount of latency is acceptable in most circumstances, it may be too long in certain cases, such as video-conferencing or a Web broadcast synchronized to a television event. In these cases, you can lower the end-to-end broadcast latency for RealAudio or RealVideo over RTSP to as low as two seconds. But if data packets are lost on the network, the broadcast is more likely to stutter.

> **Note:** *Latency* in this context refers to the difference in time between when live events occur and when they appear in RealPlayer. There is also a separate startup latency that occurs when a viewer clicks a link to a broadcast. Helix Server automatically reduces startup latency whether or not you use end-to-end latency reduction.

> **For More Information:** The section on end-to-end latency reduction in the broadcast planning chapter of *RealProducer User's Guide* provides additional information about this topic.

Latency Modes

You cannot precisely control the amount of broadcast latency because network conditions are variable and unpredictable. However, RealProducer, Helix Server, and RealPlayer provide features that allow you to reduce broadcast latency significantly under stable network conditions. These

products support three separate latency modes, summarized in the following table.

Summary of End-to-End Latency Modes

Latency Mode	Expected End-to-End Latency	Preroll for CBR Video	Preroll for VBR Video	Preroll for Audio-Only Clips	Error Resiliency
0–normal	10 or more seconds	4 seconds	4 seconds	23 milliseconds to 2.3 seconds depending on the codec	high
1–reduced	4 or more seconds	2 seconds	2 seconds	500 milliseconds	moderate
2–minimal	2 or more seconds	1 second	0.5 seconds	23 to 480 milliseconds depending on the codec	low

Receiver Buffering with Low-Latency Streams

When Helix Server receives a broadcast stream from RealProducer, it buffers the stream for a certain time before delivering the stream to RealPlayer. It automatically adjusts buffering times based on the latency mode set in the broadcast stream by RealProducer. This ensures that the server buffers data less when reduced end-to-end latency is desired. The following table lists the default receiver buffering used for each latency mode.

Receiver Buffering Based on Latency Mode

Latency Mode	Receiver Buffering
0 – normal latency	1 second
1 – reduced latency	0.05 seconds
2 – minimal latency	none

Forward Error Correction with Low Latency

The amount of forward error correction (FEC) used by RealProducer also affects how much receiver buffering should be used. For low latency, you may want to reduce or turn off error correction on all transmitters (when push splitting is used) or receivers (when pull splitting is used) in the broadcast chain. Although adding error correcting packets to a low-latency stream does

not cause problems with latency reduction, the packets may arrive too late to be of use.

> **For More Information:** See "Defining a Push Transmitter" on page 200 and "Enabling Pull Splitting Requests" on page 207. Error correction in RealProducer is described in the broadcast chapter of *RealProducer User's Guide*.

Product Version Requirements for Low Latency Broadcasts

Latency reduction is supported only with RealMedia broadcast that use RTSP. A low-latency broadcast requires the use of version 11 products or higher throughout the broadcast chain:

- Latency reduction requires RealProducer 11 or later to encode the latency mode flag into the live stream. Earlier versions of RealProducer cannot produce a low-latency stream.
- Low-latency broadcasts are recommended only when most viewers are using RealPlayer 11 or later. Based on the latency mode flag, these players request fewer or no resends for lost packets. Earlier versions of RealPlayer may request resends of dropped packets, introducing unnecessary traffic into the network.
- The packet structure for low-latency broadcasts is not compatible with server versions earlier than Helix Server version 11. If a splitting chain includes earlier servers, you may need to turn off the low latency feature at certain transmitters in the chain.
- Helix Proxy does not currently support latency reduction in live broadcasts. If Helix Proxy splits a live, low-latency stream to multiple RealPlayers, it buffers the stream, introducing broadcast latency. As explained in "Restricting Proxies from Caching or Splitting Content" on page 123, you can instruct Helix Proxies not to split the low-latency stream. This preserves the latency reduction but increases Helix Server bandwidth use because Helix Proxy passes a separate server stream to each of its RealPlayer clients.

Disabling Low Latency Streams

Because server versions earlier than Helix Server version 11 do not support the low-latency packet structure, you should turn off low-latency features when pushing the broadcast to an older server. Manually disabling low latency is not necessary for pull broadcasting, however. If an older server version requests a

low-latency stream, the pull-enabled transmitter detects the receiver server version and disables the latency feature automatically.

Even if all servers in the splitting chain are version 11 or higher, you may want to turn off low latency if you are not sure that the majority of your intended audience uses RealPlayer 11 or later. Although earlier players will play the low-latency stream, they may request data resends that increase the network overhead.

As explained in the following sections, you can turn off low-latency to all downstream receivers, or to selected receivers. When you turn off the low-latency flag, all downstream receivers use their normal data buffering, which the section "Receiver Buffering with Low-Latency Streams" on page 193 describes. As well, RealPlayers do not use their low-latency features. The broadcast therefore functions like a normal-latency broadcast for all downstream recipients.

Disabling Low Latency Globally

The Helix Server configuration file includes a global variable that turns off the low latency feature for all incoming streams. This causes Helix Server and all downstream receivers to use their normal receiver buffering values, which the section "Receiver Buffering with Low-Latency Streams" on page 193 describes. To disable latency this way, edit the configuration file and set the `AllowLowLatency` variable to `0`, as shown in the following example:

```
<Var AllowLowLatency="0"/>
```

> **For More Information:** For instructions on modifying the configuration file, refer to Appendix A, as well as *Helix Server Configuration and Registry Reference*.

Disabling Low Latency for a Specific Receiver

To turn off the low-latency feature for a specific receiver in a push broadcast, set the **Low Latency** pull-down for that receiver to `Disabled` in the transmitter definition. Although the transmitter uses its low-latency buffering value, the specified receiver and any subsequent downstream receivers use standard buffering, as described in "Receiver Buffering with Low-Latency Streams" on page 193. As well, RealPlayers receiving a stream from the affected receivers do not use their low-latency features. For all downstream recipients, therefore, the broadcast exhibits normal latency.

Note: Turning off latency reduction globally through the `AllowLowLatency` variable disables the low-latency feature regardless of the setting for a specific transmitter.

For More Information: "Setting Up a Transmitter" on page 200 explains how to configure a transmitter through the Helix Administrator graphical application.

Stream Acquisition Latency for Pull Splitting

Receiver latency occurs in pull splitting when the first media player requests the broadcast. On the first request, the receiver pulls the stream, but must wait for session announcement information to arrive from the transmitter. To determine how long a receiver takes to acquire a live split stream, you need to consider the following:

- The degree of normal network latency that exists between the transmitter and receiver.
- The configurable interval used to send session descriptions, which is defined on the transmitter in push splitting, and on the receiver in pull splitting.

A receiver does not start to process incoming data until it receives the session description from the transmitter. If the transmitter's session metadata rate is set to 30 seconds, for example, the receiver needs to wait up to 30 seconds to get the session description and begin to process broadcast packets. This time delay does not include any network latency that may exist between the transmitter and the receiver.

As soon as the live distribution stream arrives at the receiver, the receiver constructs a local buffer before streaming data to the connected media players. To help minimize latency, a receiver that uses pull splitting builds only a 1-second buffer before sending the stream to the media player that initiated the pull splitting session.

Note: With SureStream-aware splitting, session description information is sent to receivers even when no players are connected. Because the receiver does not have to wait for the next session description, latency is not an issue even though SureStream-aware splitting functions like pull splitting.

Bandwidth Consumption

Calculating the bandwidth used by a broadcast depends on whether the broadcast is single-rate stream, or a multi-rate format like SureStream. Single-rate broadcasts consume bandwidth according to the following criteria:

- The encoded bit rate of the live stream.
- The percentage of overhead (approximately 10 percent) for session announcement information and TCP/IP headers.
- The percentage of overhead for the configurable error correction rate.

For example, a single-rate 100-Kbps stream with an error correction rate set to 10 percent (10 Kbps) will need approximately 10 Kbps of overhead, for a total bandwidth of approximately 120 Kbps.

The bandwidth consumed by a SureStream broadcast equals the combined bit rates of all the audiences, plus the overhead percentages described previously. If SureStream-aware splitting has been enabled, bandwidth use is determined by the actual number of bit rates requested by media players, however. Your calculation will give you a maximum, but not all streams may be requested during the broadcast.

Error Correction and Receiver Buffering

When you use push or pull splitting, setting a low forward error correction rate (FEC) may cause error correction packets to arrive after the receiver has transmitted the contents of its data buffer to its connected clients. When this happens, the error correction packets are useless. This problem is most likely to occur with low-bandwidth RealVideo streams when your splitting arrangement uses an error correction rate between 1 and 20.

Recommended Receiver Buffering Times for RealVideo

To ensure that error correction packets reach the receiver before it streams its buffer to clients, you can increase the time that the receiver buffers data. The following table provides some minimum receiver buffering times when splitting RealVideo streams. For example, if you split a 34 Kbps RealVideo stream using an error correction rate of 5, set a receiver buffering time of at

least six seconds. The default receiver buffering time for push or pull splitting is one second.

Minimum Receiver Buffering Times for RealVideo

Speed	FEC=20	FEC=15	FEC=10	FEC=5	FEC=1
34 Kbps	2 seconds	2 seconds	3 seconds	6 seconds	30 seconds
80 Kbps	1 second	1 second	2 seconds	3 seconds	13 seconds
150 Kbps	1 second	1 second	2 second	3 seconds	11 seconds
225 Kbps	1 second	1 second	1 second	2 seconds	8 seconds
350 Kbps	1 second	1 second	1 second	2 seconds	7 seconds
450 Kbps	1 second	1 second	1 second	1 second	5 seconds

Note the following about these recommended buffering times:

- The preceding values are for RealVideo only. Other video formats may have different packet sizes, and hence may require different receiver buffering times.
- Because the default buffering value for push and pull splitting is one second, it is not necessary to modify your configuration file unless you need to set a higher value.
- Audio packets are much smaller than video packets, so when you stream audio-only clips, you generally do not need to increase the buffering time on the receiver when using a low error correction rate.
- The buffering times given in the preceding table are minimum values. You can always set a higher buffering time. However, higher buffering times may delay initial playback unnecessarily when a media player requests a stream.
- For push splitting, you set the error correction rate on the transmitter, as described in "Defining a Push Transmitter" on page 200. For pull splitting, you set the rate on the receiver as described in "Enabling Pull Splitting Requests" on page 207. You set the receiver buffering time manually through the receiver configuration file.

Setting Receiver Buffering Variables

To make the receiver buffer the split stream longer than one second, edit the receiver's configuration file with any text editor. You need to add one or two variables depending on whether you use push splitting, pull splitting, or both.

The variables are not predefined, and you add them within the Receivers list, which falls under the BroadcastReceivers list.

For push splitting, add <Var PushBufferingTime="*milliseconds*"> to the list. For pull splitting, add <Var PullBufferingTime="*milliseconds*">. Each variable's value is an integer that specifies the number of milliseconds of buffering. It is easiest to add one or both variables below the error correction variable. For example:

```
<Var FECLevel="10"/>
<Var PushBufferingTime="2000">
<Var PullBufferingTime="3000">
```

> **Note:** Although you set the buffering times for both push and pull splitting within the receiver list, the FECLevel variable in this list sets the forward error correction only for pull splitting. For push splitting, the error correction rate is set through the FECLevel variable in the transmitter list on the Helix Server used as the transmitter.
>
> **For More Information:** See Appendix A for information about the configuration file. For information about configuration variables, see *Helix Server Configuration and Registry Reference*.

Other Features Used with Splitting

This following table describes the ways in which splitting works with other features.

Splitting Used with Other Features

Feature	Notes
SLTA	SLTA transmits prerecorded clips as simulated live broadcasts. Using it is a good way to test your receiver configurations before you broadcast a live event
Live Archiving	Both transmitters and receivers can archive live RealMedia streams. With SureStream-aware splitting, archive on the transmitter connected to the encoder to ensure that all streams are archived.
Multicasting	A multicast transmission between receivers and media players, as described in "Unicast Delivery, Multicast Distribution" on page 186, makes the most effective use of bandwidth.

(Table Page 1 of 2)

Splitting Used with Other Features (continued)

Feature	Notes
Helix Proxy	Because live events are not files, Helix Proxy cannot cache broadcasts. Instead, Helix Proxy replicates live streams among media players through pull splitting.
Firewalls	In their default configuration, the transmitter and the receiver use UDP for fast, efficient distribution. Some firewalls may block UDP traffic, however. For more information, see "Communicating With Receivers" on page 256.
Access Control	If you use the access control feature to block access to your transmitter, receivers can still receive push broadcasts, but transmitters cannot honor a receiver's resend requests. Pull splitting receivers will not receive any content at all.
Authentication	When splitting a broadcast that requires viewer validation, place copies of all databases that store authentication information on receivers to distribute the authentication load.

(Table Page 2 of 2)

Setting Up a Transmitter

The following sections explain how to configure Helix Server as a push or a pull transmitter. Although the same Helix Server can function as a push or a pull transmitter in different cases, you define each transmitter instance separately. That is, the push transmitter settings do not require you to set anything in the pull settings area, and vice versa.

If your Helix Server is receiving a broadcast stream from RealProducer or SLTA, you set it up as a receiver, not a transmitter. Here, RealProducer or SLTA is the transmitter, which you set up as described in *RealProducer User's Guide* or Chapter 10, respectively. In these cases, you need to define Helix Server as a transmitter only if it forwards the broadcast stream to other Helix Server receivers.

Defining a Push Transmitter

You create a separate push transmitter definition for each receiver that your Helix Server transmits to. To broadcast to multiple receivers at the same time, you define multiple transmitter instances that all use the same broadcast source path. Alternatively, you can set up receivers to receive only certain broadcasts. This lets you create a large list of receivers that are sent broadcast streams only when those streams include certain path criteria.

➤ To define a push transmitter:

1. In Helix Administrator, click **Broadcast Distribution>Transmitter**.
2. In the **Source Name** box, enter a name that identifies this transmitter. Because this name appears in links to the split broadcast, use letters and numbers, but not spaces. Examples in this chapter use `Japan` as the transmitter name.
3. In the **Broadcast Receivers** area, click the "+" icon, and change the name in the **Edit Receiver Name** box to the name or description for the Helix Server that will receive the broadcast stream. This name is for your reference only, and is not used in the broadcast.
4. For **Source Path**, specify the mount point and, optionally, a path and a stream name used for broadcasts from this transmitter. The default is `/broadcast/`, and you can use just this mount point to send the receiver all broadcast streams delivered to this transmitter by RealProducer or SLTA.

   If you use an earlier version of RealProducer, change the broadcast mount point to `/encoder/`. For Windows Media broadcasts, the default is `/wmtencoder/`. And for QuickTime or general RTP-based broadcasts, the mount point is `/rtpencoder/`. If you are using redundant encoders with any media format, use the mount point `/redundant/`.

   > **For More Information:** For more on using a path and stream name, see "Multiple Splitting Definitions" on page 190.

5. In the **Receiver IP Address or Hostname** box, enter the host name, IPv4 address, or IPv6 address of the Helix Server that receives this broadcast stream. If you plan to multicast the stream from this transmitter, you must specify an appropriate class D, IPv4 multicast address.
6. For **Local IP Address**, specify the network address used for transmitting this broadcast. The receiver requires this address for resend requests.

   > **Tip:** If you have a multi-homed machine that uses several IP addresses, be sure to use an IP address bound to Helix Server, as described in "Binding to an IP Address" on page 66.

7. From the **Transport** list, select the transport method for transmitter-to-receiver communication. You must set the same transport on both the transmitter and the receiver. The options, listed in order of recommended use, are the following:

- udp/unicast

  This is the most common option. It unicasts the stream from the transmitter to the receiver over UDP.

- udp/multicast

  If your network hardware is multicast enabled, this option is the most bandwidth-efficient. Selecting this option requires the receiver to use a class D multicast address.

- tcp

  Use this option only if broadcasting with UDP is not possible. TCP broadcasts are less efficient because they require more network overhead.

8. Several fields and lists let you configure optional transmitter features:

   a. To enable SureStream-aware splitting, ensure that `Enabled` is selected in the **SureStream Aware Splitting** box. Otherwise, select `Disabled`. The setting has no effect if you are not splitting a SureStream broadcast. For more information, see "SureStream-Aware Splitting" on page 183.

   b. The default setting of `Enabled` for the **Low Latency** pull-down allows Helix Server to split a low-latency stream to the designated receiver. Setting this pull-down to `Disabled` returns the stream to its normal latency behavior. You should do this if any downstream receivers predate Helix Server version 11, or if the intended audience includes a large percentage of RealPlayers earlier than version 11.

      Note: This field has no effect on streams other than RealMedia broadcasts that use the low-latency feature.

      For More Information: For an explanation of the conditions under which you should disable low latency transmissions, refer to "Disabling Low Latency Streams" on page 194.

   c. If you selected the **udp/multicast** transport option, enter a number in the **Multicast Time to Live (TTL)** box. For a list of possible TTL values, refer to "Packet Time to Live" on page 159. The default is 16.

   d. For **Error Correction Rate**, you can set the percentage of corrective packets sent. The default value for Internet splitting is 20, which adds a 20% bandwidth overhead to the stream for error correction. A higher number results in more reliable delivery of data, but consumes

proportionally more network bandwidth. If you're splitting within a local area network, you can set this to a lower value or to `0` (zero) because you are less likely to need error correction.

> **Note:** If you lower the error correction rate, you may need to raise the receiver buffering time, as described in "Error Correction and Receiver Buffering" on page 197.

e. The **Metadata Transmit Rate** field sets the frequency in seconds at which special packets are sent from the transmitter to the receiver. These packets describe the stream being split, and are required for the receiver to process the incoming stream. Use a number from 1 to 60. A lower number may result in lower startup latency on a receiver, but consumes more network bandwidth. See "Stream Acquisition Latency for Pull Splitting" on page 196 for more information.

f. To improve quality of service, receivers can re-request packets that arrived in poor quality, or not at all. Select `Yes` in **Honor Resend Requests** to enable packet resending. Because this option requires a back channel from the receiver to the transmitter, it increases bandwidth use slightly to enable the resend traffic.

g. Set **Relay Live Broadcasts** to `Yes` if this transmitter is a relay that forwards streams from one Helix Server to another Helix Server, as described in "One-to-One Chaining" on page 185. If the originating encoder delivers the stream to this Helix Server, set the value to `No`.

> **Note:** A `Yes` value for **Relay Live Broadcasts** means that this server's address does not appear in the media request URL on a receiver further down a splitting chain. The server will still forward a stream if this value is `No`. But in this case, the server's address must appear in the request URLs on receivers.

9. The **Port Range** boxes set the ports on receivers to which the stream is sent. If you change the default ports, RealNetworks recommends using a range of 20 ports. If you cannot use a range of 20 ports, reserve at least 1 port per Megabit of broadcast stream bandwidth.

10. To secure stream transmissions, select `Basic` from the **Security Type** list, and type a password in the **Password** box. The transmitter passes this word to the receiver to verify its identity. To turn off the password requirement,

choose None from the **Security Type** list. The password needs to be defined on the receiver as well.

11. Click **Apply**.

Configuring a Pull Splitting Transmitter

The following procedure explains how to set up Helix Server as a pull splitting source. Pull splitting definitions are entirely independent of push splitting information. With pull splitting, you do not define specific receivers as you do in push splitting. Instead, you set basic parameters that identify the broadcast stream. Then, any receiver can pull the stream. In other words, the pull splitting transmitter definition does not contain any information about potential receivers.

➤ To configure the transmitter for pull splitting:

1. In Helix Administrator, click **Broadcast Distribution>Transmitter**.
2. In the **Pull Splitting Sources** area, click the "+" icon and edit the name that appears in the **Edit Source Name** box. This name is for your reference only, and does not appear in URLs.
3. In the **Source Path** box, enter the path through which live streams are sent, typically /broadcast/, /encoder/, /redundant/, /wmtencoder/, or /rtpencoder/, as explained in Chapter 7. The default value of /broadcast/ means that this transmitter definition allows receivers to pull any broadcast sent to the transmitter from RealProducer in push or pull mode.

   Tip: Optionally, you can specify a path and stream name to define only specific broadcasts that can be pulled. For an example of using source paths, refer to "Multiple Splitting Definitions" on page 190.

4. For **Local IP Address**, specify the network address the transmitter uses. This is the address the receiver contacts to pull the stream. All pulled streams are unicast over UDP when possible, or TCP if UDP is not available.
5. In the **Listen Port** box, type a port number on which this transmitter listens for requests. These port numbers are used in request URLs. If you leave this field blank, Helix Server uses the value 2030.

> **Tip:** All pull splitting definitions can use the same listen port, as long as no simultaneous broadcasts use the same stream name. If you want to broadcast the same stream name at the same time through different pull splitting definitions, give each definition a unique listen port number.

6. If you want to enable SureStream-aware splitting, select **Enabled** in the **SureStream Aware Splitting** box. For more information, see "SureStream-Aware Splitting" on page 183.
7. Type a password in the **Password** box. The receiver will use this word to verify its identity when it requests a stream. To turn off this feature, choose `None` from the **Security** list.

   > **Warning!** Password validation is strongly recommended for all pull-splitting broadcasts. If you turn off validation, any server can pull the stream from your transmitter.

8. Click **Apply**.

Setting Up a Receiver

The following sections describe how to define basic receiver information, and how to enable that receiver to make pull splitting requests. To configure a receiver, you need to know the following settings on the transmitter:

- broadcast source (mount point, path, and stream name)
- transmission port range
- transmitter address
- security type and password
- transport method (unicast, multicast, or TCP)
- listen port (pull splitting only)

Defining Basic Receiver Information

For a pull splitting or push splitting receiver, you set up basic information about the receiver as described in the following procedure. Just as a transmitter definition lists properties of receivers, the receiver defines properties of transmitters.

➤ To configure a receiver:

1. In Helix Administrator, click **Broadcast Distribution>Receiver**.
2. The **Mount Point** box defines the mount point used in URLs to the broadcast on the receiver. It identifies the source as a live broadcast rather than a clip. The default is `/broadcast/`.

   Tip: A broadcast may go out from a transmitter on a different mount point, such as `/wmtencoder/` for Windows Media. Each receiver can broadcast the split stream from its `/broadcast/` mount point regardless of the media format, however.

3. In the **Broadcast Transmitters** area, click the "+" icon and enter a descriptive name for the transmitter in the **Edit Transmitter Name** box.
4. For **Transmitter Address/Netmask**, type the host name, IPv4 address, or IPv6 address of the transmitter. To receive broadcast streams from a range of transmitters, specify a bit mask after the specified IP address, separating the two entries with a forward slash ("/").

   For More Information: See Appendix B for information about using a bit mask.

5. In the **Port Range** boxes, set the ports on which the stream is received. The port values should match the settings on the transmitter.
6. In the **Transport** box, select the same option that's in use on the transmitter. If you select `udp/multicast`, you must also add to the **Multicast Address** field a class D multicast address. This address must match the value in the transmitter's **Receiver IP Address or Hostname** box.
7. To re-request packets that did not arrive at the receiver, select `Yes` for **Resend Requests**. This option provides greater quality, but requires more network bandwidth. If you enable resend requests, the transmitter needs to have **Honor Resend Requests** set to `Yes`.
8. For **Security Type** and **Password**, use the same values set on the transmitter.
9. If you are not enabling pull splitting on the receiver, click **Apply**. Otherwise, configure pull splitting as described below.

Enabling Pull Splitting Requests

To enable pull splitting, you set the receiver properties explained in "Defining Basic Receiver Information" on page 205. Then, add the pull splitting information described in the following procedure.

➤ To enable the receiver for pull splitting:

1. On the **Broadcast Distribution>Receiver** page, select the receiver you want to configure for pull splitting, and scroll down to bottom of the page.
2. From the **Enable Pull Splitting Requests** list, select `Yes`.
3. In the **Pull Splitting Virtual Path** box, type a value in the form of a mount point, such as `/split/`. You will use this virtual path in URLs to indicate that the stream is pulled.
4. For **Error Correction Rate**, set the percentage of corrective packets sent by the transmitter (the FEC rate set on a transmitter is used only with push splitting). The default value is 0, which generates no corrective packets. A higher number for FEC results in more reliable delivery of data, but consumes proportionally more network bandwidth. A value of 20, for instance, adds a 20% overhead to the stream for error correction.

   Tip: If you're splitting within a local area network, you can keep this value at `0` (zero) because you are less likely to need error correction than when splitting across the Internet.

   Note: If you use an error correction rate between 1 and 20, you may need to raise the receiver buffering time, as described in "Error Correction and Receiver Buffering" on page 197.

5. The **Metadata Transmit Rate** box sets the rate at which the transmitter sends packets describing the session (the metadata rate set on a transmitter is used only with push splitting). A higher number results in better quality and higher startup latency on the receiver, but consumes less network bandwidth. Use a number from 1 to 60. The default value is 30, which sends the metadata once every 30 seconds, adding up to 30 seconds of latency to the initial pull request.
6. To split over a connection in which stream packets are likely to be lost, you can set the **Pull Splitting Backchannel Transport** field to `TCP`. This setting results in a highly reliable connection that requires slightly more network overhead.

7. Click **Apply**.

Linking to Split Content

A link to a split stream points to the receiver, and indicates the transmitter where the stream began. Links for push splitting and pull splitting streams are different. For both types of links, you can use URL aliases, which are especially useful for the longer pull splitting links. The following sections explain link formats in general, and provide example links for RealMedia broadcasts. For broadcasts in other formats, you need to use transmitter mount points and stream names as appropriate for that format.

> **Tip:** For examples of links to basic broadcasts in different media formats, see "Linking to Unicasts" on page 151. The links to split streams build on those basic formats.

Writing Push Splitting Links

Push splitting links point to the Helix Server receiver, but include the source name of the transmitter to identify the broadcast. These URLs are relatively simple, and build on the basic unicast URL format.

Linking from a Web Page

A Web page link to a pushed RealMedia broadcast in which a Helix Server receiver uses the default port 80 for HTTP might look like this:

```
http://receiver.example.com/ramgen/broadcast/Japan/broadcast/news/live.rm
```

- The URL first specifies the IP address or host name of the receiver, as in `receiver.example.com`. Include the HTTP port number if port 80 is not used.
- The `/ramgen/` mount point launches RealPlayer, as explained in "Using a Client Launch Utility" on page 90. For a Windows Media broadcast, you use `/asxgen/`.
- The standard broadcast mount point for RealMedia on a receiver is `/broadcast/`, but each receiver may define a different mount point, as described in "Defining Basic Receiver Information" on page 205.
- Following the broadcast mount point, you specify the transmitter name, such as `/Japan/`. The **Source Name** box of the transmitter definition sets this name, as explained in "Defining a Push Transmitter" on page 200.

- The URL next specifies the transmitter's broadcast mount point and, optionally, its virtual path. These are defined in the transmitter's **Source Path** box. The mount point may also include a virtual path, as in `/broadcast/news/`.
- The requested file, shown above as `live.rm`, is the stream name specified by the encoder. With Windows Media pull broadcasts, the stream name is defined on the transmitter, as explained in "Setting up a Windows Media Pull Broadcast" on page 143. In a Windows Media push broadcast, Windows Media Encoder defines the stream name, as noted in "Running a Windows Media Push Broadcast" on page 147. For QuickTime and RTP-based broadcasts, the stream name is the SDP file name.

Linking through a Relay

It typically does not matter if one or more relays exist between the receiver and the transmitter, as illustrated in "One-to-One Chaining" on page 185. The link refers only to the receiver and the originating transmitter, as long as each relay has its **Relay Live Broadcasts** field set to `Yes`.

If one or more relays in the chain has **Relay Live Broadcasts** set to `No`, however, you need to include those servers' transmitter names in the URL. Here is an example of a Web page URL that lists two relays along with the transmitter and receiver address:

```
http://receiver.example.com/ramgen/broadcast/LosAngeles/broadcast/
Sydney/broadcast/Japan/broadcast/news/live.rm
```

In this example, the `Japan` transmitter is the originating transmitter, just as in the preceding examples. The stream travels from this transmitter to the `Sydney` transmitter, then to the `LosAngeles` transmitter, before arriving at the receiver. Note that the URL components reflect the exact order in which the stream travels through the chain.

Linking to a Transmitter or Relay

An originating transmitter, as well as any relay in the chain, can also deliver the broadcast to media players. URLs to an originating transmitter are standard unicast URLs, as described in "Linking to Unicasts" on page 151. The Web-page URL to the broadcast on the `Japan` transmitter would therefore look like this:

```
http://transmitter.example.com/ramgen/broadcast/news/live.rm
```

To link to the broadcast on a relay, you treat the relay as if it were the receiver at the end of the splitting chain. It does not matter if the relay has its **Relay Live Broadcasts** value set to `Yes` or `No`. For example, the Web page link to the broadcast on the `LosAngeles` relay, which links through the `Sydney` relay to the `Japan` transmitter looks like this:

```
http://relay2.example.com/ramgen/broadcast/Sydney/broadcast/
Japan/broadcast/news/live.rm
```

Linking from a Ram or SMIL File

You can also launch a broadcast through a Ram file or SMIL file, as described in "Using Metafiles" on page 88. The URL format is similar to the preceding examples, but specifies RTSP or MMS, and omits the `/ramgen/` or `/asxgen/` parameter. In the following example of a URL to a receiver, the port number is omitted, which means that port 554 is used for RTSP:

```
rtsp://receiver.example.com/broadcast/Japan/broadcast/news/live.rm
```

Here is a sample URL to the unicast on the `Japan` transmitter:

```
rtsp://transmitter.example.com/broadcast/news/live.rm
```

Creating Pull Splitting Links

Pull splitting links are more complex than push splitting links. The link URL tells the receiver where to pull the stream by giving the transmitter's address and listen port. If you have a relay chain as described in "One-to-One Chaining" on page 185, a link to a broadcast on a specific receiver must indicate all of the preceding links in the chain in the proper order. URL aliases can shorten the published URLs, as well as mask the transmitter addresses.

> **Note:** When pull-splitting Windows Media streams, you must use URL aliases because Windows Media Player does not recognize the URL format used for pull splitting.

Linking from a Web Page

A Web page link to a pulled RealMedia broadcast in which a Helix Server receiver uses the default port 80 for HTTP might look like the following example, in which the URL is shown on two lines to clarify the link portions. The first line gives the receiver information, while the second line supplies the transmitter parameters:

```
http://receiver.example.com/ramgen/broadcast/split/
transmitter.example.com:2030/broadcast/news/live.rm
```

- The URL first specifies the IP address or host name of the receiver, as in `receiver.example.com`. Include the HTTP port number if port 80 is not used.
- The `/ramgen/` mount point launches RealPlayer, as explained in "Using a Client Launch Utility" on page 90. For a Windows Media broadcast, you use `/asxgen/`.
- The standard broadcast mount point for RealMedia on a receiver is `/broadcast/`, but each receiver may define a different mount point, as described in "Defining Basic Receiver Information" on page 205.
- Following the broadcast mount point, you specify a path that indicates a pull splitting broadcast, as in `/split/`. The **Pull Splitting Virtual Path** box of the receiver's pull splitting section sets this name, as explained in "Enabling Pull Splitting Requests" on page 207.
- The link's transmitter portion lists the transmitter's IP address or host name, along with its listen port, as in `transmitter.example.com:2030`. The section "Configuring a Pull Splitting Transmitter" on page 204 describes these values.
- The URL next specifies the transmitter's broadcast mount point and, optionally, its virtual path. These are defined in the **Source Path** box of the pull splitting section in the transmitter definition. The standard value for a RealMedia broadcast is `/broadcast/`, but the mount point is different for various media formats.

  > **Note:** This mount point is not included in the URL if the transmitter is an encoder such as RealProducer rather than a Helix Server transmitter.

- The requested file, shown above as `live.rm`, is the stream name specified by the encoder. The name may also include a virtual path, as in `news/live.rm`. With Windows Media broadcasts, the stream name is defined on the transmitter, as explained in "Broadcasting Windows Media" on page 143. For QuickTime and RTP-based broadcasts, the stream name is the name of the SDP file.

  > **For More Information:** You can also link to the transmitter as described in "Linking to a Transmitter or Relay" on page 209.

Linking through a Relay

If the stream travels through a relay before reaching the receiver, the URL must include the relay address. When a media player requests the broadcast from the receiver, the receiver contacts the relay to pull the stream, which, in turn, contacts the transmitter to pull the stream. The relay information comes between the transmitter and receiver information. As with the transmitter, the URL gives the relay's address, listen port, and broadcast mount point:

```
http://receiver.example.com/ramgen/broadcast/split/
relay.example.com:2030/broadcast/
transmitter.example.com:2030/broadcast/news/live.rm
```

Linking from a Ram or SMIL File

You can also launch a pull splitting broadcast through a Ram file or SMIL file, as described in "Using Metafiles" on page 88. The URL format is similar to the preceding examples, but specifies RTSP or MMS, and omits the `/ramgen/` or `/asxgen/` parameter. In the following example of a URL to a receiver, the port number is omitted, which means that port 554 is used for RTSP:

```
rtsp://receiver.example.com/broadcast/split/
transmitter.example.com:2030/broadcast/news/live.rm
```

Using URL Aliases

The section "Setting Up Aliases" on page 97 explains how to create an alias for URL elements. This is a handy way to shorten URLs and mask transmitter addresses in pull splitting URLs. Aliases are optional except when you use pull splitting in a Windows Media broadcast. Windows Media Player does not recognize transmitter addresses and listen ports in pull splitting URLs.

You set up the appropriate alias on each Helix Server that receives broadcast requests from media players. To illustrate the use of an alias, consider the pull splitting example shown in the preceding section:

```
http://receiver.example.com/ramgen/broadcast/split/
relay.example.com:2030/broadcast/
transmitter.example.com:2030/broadcast/news/live.rm
```

On the receiver that handles this broadcast request, you could define the following alias to mask the relay and transmitter addresses:

URL component:	`relay.example.com:2030/broadcast/` `transmitter.example.com:2030/broadcast/news/`
Alias:	`pull/`

Your published URL would then look like this:

```
http://receiver.example.com/ramgen/broadcast/split/pull/live.rm
```

Note: You can use just one alias in each URL.

CHAPTER

10

SIMULATED LIVE BROADCASTS

The Simulated Live Transfer Agent (SLTA) is a Helix Server utility that allows you to stream a prerecorded clip or a broadcast archive as if it were a live event. Using SLTA, you can deliver encore presentations, or simulate radio or TV programming using any number of clips. This chapter covers the basic and advanced modes of SLTA, explaining how to set up SLTA, create playlists, and run your simulated live broadcast.

Understanding Simulated Live Broadcasts

SLTA is a command line tool installed with Helix Server. Running on Windows or UNIX, it sends a media stream to Helix Server, which then broadcasts the stream to media players. SLTA supports all broadcasting features:

- **multicasting**—SLTA can deliver the same stream to multiple Helix Servers on a multicast-enabled network.
- **push delivery**—SLTA can initiate the broadcast by contacting one or more Helix Servers and delivering the stream. If you are unicasting rather than multicasting to each Helix Server, the number of connections is limited primarily by the outgoing bandwidth on the machine that hosts SLTA.

  For More Information: See "Bandwidth Consumption" on page 197 for more about bandwidth use in splitting arrangements.

- **pull delivery**—Helix Server can initiate the broadcast by contacting SLTA and acquiring the stream when the first media player requests the broadcast. SLTA supports connections from multiple pull-enabled receivers. The number of connections is limited primarily by the outgoing bandwidth on the machine that hosts SLTA.

- **broadcast redundancy**—You can run separate SLTA applications on different machines to create primary and backup streams.
- **SureStream-aware splitting**—SLTA can conserve bandwidth by delivering each SureStream stream only when it is requested.

> **For More Information:** For the basics of splitting, see "Understanding Splitting" on page 177.

Broadcast Formats

SLTA can simulate an audio or video broadcast in RealMedia, QuickTime, supported MPEG (including MP3), AU, WAV, or Windows Media format. To simulate a broadcast using QuickTime clips, you can also use Playlist Broadcaster to deliver prerecorded media to Helix Server, which then broadcasts it as described in "Broadcasting QuickTime, MPEG, and RTP-Based Media" on page 147.

Basic and Advanced Modes

SLTA has two modes: basic and advanced. Basic mode simulates account-based broadcasting in RealProducer. It allows you to push the broadcast stream to a single Helix Server. It automatically sets to predefined values certain stream delivery variables that are configurable in advanced mode. For example, basic mode always uses a 10% forward error correction rate and a 30-second data acquisition interval.

In advanced mode, SLTA functions like a transmitter in a splitting setup, sending a stream to one or more Helix Servers configured as receivers, as described in "Setting Up a Receiver" on page 205. The receivers then broadcast the stream to media players. The following table summarizes the features available in basic and advanced modes.

SLTA Basic and Advanced Mode Features and Requirements

Feature or Requirement	Basic	Advanced	Reference
RealMedia, Windows Media, QuickTime, or MPEG	**yes**	**yes**	page 216
Helix Server configured as a receiver	no	**yes**	page 218
SLTA configuration file necessary	no	**yes**	page 222
pull delivery	no	**yes**	page 228
push delivery	**yes**	**yes**	page 224

(Table Page 1 of 2)

SLTA Basic and Advanced Mode Features and Requirements (continued)

Feature or Requirement	Basic	Advanced	Reference
push delivery to multiple servers	no	**yes**	page 224
multicast delivery to multiple servers	no	**yes**	page 225
SureStream-aware delivery	no	**yes**	page 225
configurable forward error correction	no	**yes**	page 225
ignore server resend requests	no	**yes**	page 225
configurable metadata transmit rate	no	**yes**	page 225
path name preceding stream name	no	**yes**	page 226
SLTA-buffered transport	no	**yes**	page 227
broadcast redundancy	**yes**	**yes**	page 236
UDP transport	**yes**	**yes**	page 238
TCP transport	**yes**	**yes**	page 238
clip playlists	**yes**	**yes**	page 229
shuffle play	**yes**	**yes**	page 238
clip title, author, and copyright overrides	**yes**	**yes**	page 238
wallclock synchronization	**yes**	**yes**	page 238

(Table Page 2 of 2)

Helix Server Setup

Using SLTA requires a small amount of setup on Helix Server.

Basic Mode Configuration

SLTA's basic mode is authenticated, so you need to set up a user name and password on Helix Server:

- Although you can use the name and password that logs you into Helix Administrator, RealNetworks recommends that you define a new name and password as described in "Encoder Validation" on page 270.
- You select the encoder realm that contains the SLTA user name and password under **Broadcasting>RealNetworks Encoding**, as described in "Setting Up Account-Based Broadcasting" on page 140.

> **Tip:** In basic mode, SLTA uses the port range defined on the RealMedia broadcasting page (**Broadcasting>RealNetworks Encoding**). The default settings are generally sufficient.

Advanced Mode Configuration

When you run SLTA in advanced mode, you configure Helix Server as a receiver, as described in "Setting Up a Receiver" on page 205. RealNetworks recommends that you do this before you define your SLTA configuration file.

Command Line Operation

When you're ready to broadcast, you run SLTA from the command line, specifying the configuration file, the clip or playlist, and any additional options. For each broadcast, you define a single stream name, such as `encore.rm`, which is used in place of the actual clip or playlist name. When viewers click the link to this stream, they join the broadcast in progress.

> **Note:** To run SLTA, you must be able to open a command prompt on your operating system, and navigate to a specific directory. This chapter does not explain how to perform these functions.

SLTA Quick Start Tutorials

The following tutorials are optional. They introduce you to the procedures for simulating a broadcast, using the smallest set of variables and commands required to configure and run SLTA. Once you understand the overall operation of SLTA, you'll more easily pick up the many additional configuration features and command line options that you can use.

> **Note:** In the following tutorials, you run SLTA on your Helix Server machine. In a production environment, however, RealNetworks recommends that you run SLTA and Helix Server on separate machines.

Quick Start for SLTA Basic Mode

In this tutorial you run SLTA in basic mode.

➤ To simulate a basic mode broadcast with a prerecorded clip:

1. Anywhere on your Helix Server computer, create a directory named `Simulate`, and copy the following files to it:
   a. The `slta.exe` (Windows) or `slta` (UNIX) file from the Helix Server `Bin` directory.

b. The slta.bat (Windows) or slta.sh (UNIX) file from the Helix Server Bin directory.

c. The realvideo10.rm clip from the Helix Server Content directory. You can use another RealMedia clip if you wish.

The location of Helix Server in a default installation on Windows is C:\Program Files\Real\Helix Server.

2. Open a command prompt and navigate to your Simulate directory. Enter one of the following commands to transmit the realvideo10.rm clip (or another clip if you wish) under the stream name live.rm. On Windows:

```
slta.bat 127.0.0.1 80 name password live.rm realvideo10.rm
```

On UNIX:

```
slta.sh 127.0.0.1 80 name password live.rm realvideo10.rm
```

- You need to substitute the actual Helix Server address for 127.0.0.1 if your files are not on the same machine as Helix Server, or you're not using the local host address.
- The 80 entry refers to the standard HTTP port, which may be different on your Helix Server. See "Defining Communications Ports" on page 63 for port information.
- For this exercise, use the name and password you use to log into Helix Administrator.

3. Start RealPlayer, give the **File>Open** command, and enter the following URL:

```
rtsp://127.0.0.1/broadcast/live.rm
```

You need to substitute the actual Helix Server address if your RealPlayer is not on the same machine as Helix Server, or you're not using the local host address. You'll also need to include the RTSP port number if Helix Server does not use port 554.

4. When you finish testing, stop SLTA from the command line by pressing **Ctrl+c** on Windows, or giving the kill command with the process ID on UNIX. After a few seconds, the broadcast stops playing in RealPlayer.

Quick Start for SLTA Advanced Mode

Follow the steps in this tutorial to run SLTA in advanced mode, creating both a transmitter and a receiver on your Helix Server computer.

➤ To simulate an advanced mode broadcast with a prerecorded clip:

1. Anywhere on your Helix Server computer, create a directory named `Simulate`, and copy the following files to it:
   a. The `slta.exe` (Windows) or `slta` (UNIX) file from the Helix Server `Bin` directory.
   b. The `slta.bat` (Windows) or `slta.sh` (UNIX) file from the Helix Server `Bin` directory.
   c. The `realvideo10.rm` clip from the Helix Server `Content` directory. You can use another RealMedia clip if you wish.

   The location of Helix Server in a default installation on Windows is `C:\Program Files\Real\Helix Server`.

2. Through Helix Administrator, configure Helix Server as a receiver as described in "Setting Up a Receiver" on page 205. Enter the following values, accepting the default values for any setting not listed.

Helix Server Receiver Values for SLTA Quick Start

Setting	Value	Notes
Transmitter Name	`Simulation`	A transmitter can have any name, but the name should not include spaces.
Transmitter Address	`127.0.0.1`	Using this value requires that your Helix Server binds to the local host address, which it does by default. If you've changed this as described in "Binding to an IP Address" on page 66, use the actual network address or host name.
Transmitter Netmask	`32 bits`	This is required when using local host. It's not required if using a full address.
Security Type	`None`	This option requires no password. In an actual broadcast, RealNetworks recommends that you always use `Basic` security.

3. In a text editor, enter the following XML-based syntax, saving the file as `transmit.cfg` in your `Simulate` directory. You'll need to change the `Address` variable if you're not using the local host address. Also, make sure that the `PortRange` variable matches the setting on the receiver. Values must be quoted, and variable tags must end with a slash (/).

```
<List Name="BroadcastDistribution">
   <Var SourceName="Simulation"/>
   <List Name="Destinations">
      <List Name="TestReceiver">
         <Var PathPrefix="*"/>
         <Var PortRange="30001-30020"/>
         <Var Address="127.0.0.1"/>
         <Var Protocol="udp/unicast"/>
         <List Name="Security">
            <Var Type="None"/>
         </List>
      </List>
   </List>
</List>
```

> **Note:** This configuration sets up SLTA as a push transmitter. The variables for pull splitting, which are not shown in the preceding example, are different.

4. Open a command prompt and navigate to your `Simulate` directory. Enter one of the following commands to transmit the `realvideo10.rm` clip (or another clip if you wish) under the stream name `live.rm`. On Windows:

   ```
   slta.bat -c transmit.cfg live.rm realvideo10.rm
   ```

   On UNIX:

   ```
   slta.sh -c transmit.cfg live.rm realvideo10.rm
   ```

5. Start RealPlayer, give the **File>Open** command, and enter the following URL:

   ```
   rtsp://127.0.0.1/broadcast/Simulation/live.rm
   ```

   You need to substitute the actual Helix Server address if your RealPlayer is not on the same machine as Helix Server, or you're not using the local host address. You'll also need to include the RTSP port number if Helix Server does not use port 554.

6. When you finish testing, stop SLTA from the command line by pressing **Ctrl+c** on Windows, or giving the `kill` command with the process ID on UNIX. After a few seconds, the broadcast stops playing in RealPlayer.

Configuring SLTA for Advanced Mode

You do not need to configure SLTA to run in basic mode. In advanced mode, SLTA relies on an XML-based configuration file for instructions on transmitting to each Helix Server receiver. When you run SLTA, you indicate which configuration file to use. This allows you to define multiple configuration files for different transmission scenarios. The following sections explain how to set up your configuration file for advanced mode broadcasting.

> **Tip:** It's a good idea to set up Helix Server as a receiver before you write your configuration file. To do this, follow the instructions in "Setting Up a Receiver" on page 205.

Using the Configuration Template

The main Helix Server installation directory contains a template for the SLTA configuration file, `slta.cfg`. You can edit this template with any text editor, and save new configuration files as plain text under any name. The `.cfg` extension is recommended, but not necessary. The following example shows the contents of the `slta.cfg` template:

```
<List Name="BroadcastDistribution">
   <Var SourceName="ExampleSourceName"/>
   <List Name="Destinations">
      <List Name="ExampleName">
         <Var PathPrefix="*"/>
         <Var PortRange="30001-30020"/>
         <Var AcquisitionDataInterval="30"/>
         <Var FECLevel="0"/>
         <Var SureStreamAware="0"/>
         <Var BufferlessTransport="1"/>
         <Var LocalAddress="0.0.0.0"/>
         <Var Address="127.0.0.1"/>
         <Var TTL="16"/>
         <Var ResendSupported="0"/>
         <Var Protocol="udp/unicast"/>
         <List Name="Security">
            <Var Type="Basic"/>
            <Var Password="ExamplePassword"/>
         </List>
      </List>
    </List>
```

```
   <List Name="Pull Settings">
      <List Name="PullSource1">
         <List Name="Security">
            <Var Type="Basic"/>
            <Var Password="ExamplePassword"/>
         </List>
         <Var SureStreamAware="0"/>
         <Var ListenPort="2030"/>
         <Var PathPrefix="/"/>
         <Var LocalAddress="0.0.0.0"/>
      </List>
   </List>
</List>
```

> **Tip:** If you are not familiar with XML-formatted lists and variables, read "Editing the Configuration File" on page 394 for an overview.

Setting Basic Transmitter Properties

The configuration template has an outer list named `BroadcastDistribution`, which you should not change. This main list contains a variable for the transmitter name, and separate sections for push splitting and pull splitting variables. Typically, you need to define only one of these sections depending on the type of splitting you use:

```
<List Name="BroadcastDistribution">
   <Var SourceName="ExampleSourceName"/>
   <List Name="Destinations">
      <List Name="ExampleName">
         ...all push splitting variables here...
      </List>
   </List>
   <List Name="Pull Settings">
      ...all pull splitting variables here...
   </List>
</List>
```

> **Tip:** When using push splitting, you can delete the pull splitting variables, and vice versa. This is not necessary, though, and if you do so, be careful not to delete an element you need, such as a closing `</List>` tag. Missing elements will cause an error when you run SLTA.

Naming the Transmitter

Whether you use push splitting or pull splitting, you define a name for the SLTA transmitter in the SourceName variable at the top of the file:

```
<Var SourceName="ExampleSourceName"/>
```

If you plan to run multiple instances of SLTA on the same machine simultaneously, each configuration file should define a different source name. This name appears in Web page links for push transmissions, so it should not include spaces. Here is an example:

```
<Var SourceName="BroadcastEncore"/>
```

> **Tip:** It helps you to keep track of transmitters and receivers if you use same name you defined for SLTA in the **Broadcast Transmitters** box on the Helix Server receiver.

Defining Push Splitting

In push splitting, SLTA initiates the connection, contacting one or more Helix Server receivers, and delivering the simulated live stream. The push splitting section starts with the Destinations list, which encompasses a single receiver definition predefined as ExampleName. Change the name to any value that describes the Helix Server receiver. Here's an example:

```
<List Name="Destinations">
   <List Name="Sydney Receiver">
      ...all push splitting variables here...
   </List>
</List>
```

> **Note:** This name is for your convenience, and is not used in the broadcast. The name cannot contain a period character (“.”), however.

Specifying Multiple Receivers

If you intend to deliver the same simulated live broadcast to more than one Helix Server, duplicate the receiver list within the Destinations list, and give the second receiver a new name, as shown in the following example. Each receiver list then defines the same set of variables, but with different values as appropriate for each receiver. You can set up as many receivers as necessary:

```
<List Name="Destinations">
   <List Name="Sydney Receiver">
      ...all push splitting variables for the first receiver...
   </List>
   <List Name="Tokyo Receiver">
      ...all push splitting variables for the second receiver...
   </List>
</List>
```

SLTA can unicast a separate stream to each receiver, or multicast a single stream on a multicast-enabled network. As explained in Chapter 9, you can also use a Helix Server receiver to transmit (or "split") the SLTA stream to any number of other receivers. This may be a better solution if your SLTA transmitter has limited outgoing bandwidth and you cannot multicast from SLTA to all of the receivers.

> **Tip:** Using virtual paths, you can define a single configuration file for multiple receivers, but deliver a stream only to certain receivers when you run SLTA. For more information, see "Directing Streams through Paths" on page 226.

Setting Push Configuration Values

The following table describes the SLTA push splitting variables you define for each receiver you use. Several transmitter variables correspond to receiver variables that must have a similar value. Minimally, you need to ensure that the `Address`, `PortRange`, `Protocol`, and `Password` variables match the settings on the receiver, as described in "Setting Up a Receiver" on page 205. The section "Setting Up a Transmitter" on page 200 describes many of the transmitter features in greater detail.

SLTA Push Splitting Variables

Variable	Value	Function	Reference
`PathPrefix`	`*|`*`path`*	Streams all broadcasts with "*", or specifies a virtual path name.	page 226
`PortRange`	*port-port*	Indicates the range of ports set up on the receiver for data transmission.	page 203
`AcquisitionData Interval`	*seconds*	Sets the frequency that the transmitter sends metadata information. Valid range is 0 to 60.	page 203

(Table Page 1 of 2)

SLTA Push Splitting Variables (continued)

Variable	Value	Function	Reference
FECLevel	*integer*	Sets a percentage of corrective packets sent. The valid range is 0 to 100.	page 202
SureStreamAware	0|1	Indicates whether to use SureStream-aware transmission. The value 0 is "no," 1 is "yes."	page 202
BufferlessTransport	0|1	Uses the clip's timestamp information. The value 0 is "no," 1 is "yes." (Always set this value to 1.)	page 227
LocalAddress	*address*	Specifies the address or host name of the SLTA transmitter.	page 201
Address	*address*	Indicates the address or host name of the receiver.	page 201
TTL	integer	Sets the time to live for multicasting.	page 202
ResendSupported	0|1	Ignores (0) or honors (1) the receiver's resend requests.	page 203
Protocol	*protocol*	Sets the protocol used in transmitting streams. Choices are udp/unicast, udp/multicast, and tcp.	page 201
Type	Basic|None	Sets the type of security used.	page 203
Password	*password*	Provides a password for Basic security.	page 203

(Table Page 2 of 2)

Tip: It's OK to ignore variables for features that you do not use. For example, TTL is used only with multicasts, so its setting does not affect unicasts.

Directing Streams through Paths

To broadcast all streams to all defined receivers, leave PathPrefix set to its default value of "*" for each receiver. As described in "Multiple Splitting Definitions" on page 190, however, you can use virtual path names and stream names to direct different broadcasts to different receivers. Suppose that you define three different receivers, and give each a different PathPrefix value as shown in the following example:

```
<List Name="Destinations">
   <List Name="Sydney Receiver">
      <Var PathPrefix="*"/>
      ....
   </List>
```

```
    <List Name="Tokyo Receiver">
        <Var PathPrefix="news/"/>
        ....
    </List>
    <List Name="Bombay Receiver">
        <Var PathPrefix="news/hourly.rm"/>
        ....
    </List>
</List>
```

The Sydney receiver, which uses the default path "*", gets all streams broadcast by SLTA using this configuration file. The Tokyo receiver gets only the streams that specify the news/ virtual path along with the stream name, which can be anything, when you start the broadcast. The Bombay receiver gets only the broadcast streams named news/hourly.rm.

The advantage of using virtual paths is that you can use one configuration file to deliver different streams to different receivers at different times. Instead of creating a separate configuration file for each receiver, you can define all receivers in one configuration file, then use the virtual paths to determine which receivers get which streams when you run SLTA.

Alternatively, you can create multiple configuration files to direct broadcasts to different receivers. Then, instead of specifying a path when starting the broadcast, you select the appropriate configuration file. Either method can create the same results, and you should choose the method that you find more convenient.

> **For More Information:** The section "Starting SLTA" on page 233 explains how to indicate the path when you start the broadcast.

> **Note:** As noted in the table "SLTA Pull Splitting Variables" on page 229, pull splitting definitions can include a virtual path prefix, too. There is generally no need to set this path, however. In pull splitting, SLTA responds to requests. It does not actively direct streams as it does in push splitting. You can therefore leave the default path prefix of "/" set for pull splitting.

Using Bufferless Transport

For most clip types, you should leave BufferlessTransport set to its default value of "1". In this mode, SLTA transmits the clip according to the clip's internal timestamp information, which is appropriate for most streaming audio and

video formats. If you set this variable to "0", SLTA buffers clip data itself and builds its own broadcast scheduler, which increases broadcast latency and processor overhead. You can enable bufferless transport only for clips that stream at a constant bit rate.

Defining Pull Splitting

In pull splitting, Helix Server initiates the connection, contacting SLTA to acquire the stream when the first media player requests the simulated live broadcast. The pull splitting section starts with the `Pull Settings` list, which encompasses a single definition predefined as `PullSource1`. Change this definition name, which is not used in broadcasts, to anything that describes your SLTA transmitter (do not use a period character is the name):

```
<List Name="Pull Settings">
   <List Name="NewsEncoreTransmitter">
      <List Name="Security">
         <Var Type="Basic"/>
         <Var Password="ExamplePassword"/>
      </List>
      <Var SureStreamAware="0"/>
      <Var ListenPort="2030"/>
      <Var PathPrefix="/"/>
      <Var LocalAddress="0.0.0.0"/>
   </List>
</List>
```

Any number of receivers can pull the same stream from a single SLTA transmitter. You therefore do not need to replicate the pull splitting variables for each receiver in use, as you do with push splitting. You need to define multiple configuration files only if you intend to deliver different streams to one or more receivers at the same time. You then run multiple instances of SLTA, using a different configuration file, stream name, and playlist for each instance.

Setting Pull Configuration Values

The following table describes the SLTA pull splitting configuration variables you define. Most configuration is done on the receiver, which must be enabled for pull splitting, as described in "Enabling Pull Splitting Requests" on page

207. The section "Setting Up a Transmitter" on page 200 describes transmitter features in greater detail.

SLTA Pull Splitting Variables

Variable	Value	Function	Reference
PathPrefix	/|*path*	Streams all broadcasts with "/", or specifies a virtual path name.	page 227
ListenPort	*port*	Set the port where SLTA listens for stream requests.	page 204
SureStreamAware	0|1	Indicates whether to use SureStream-aware transmission. The value 0 is "no," 1 is "yes."	page 205
LocalAddress	*address*	Specifies the address or host name of the SLTA transmitter.	page 204
Type	Basic|None	Sets the type of security used.	page 205
Password	*password*	Provides a password for Basic security.	page 205

Creating a Playlist

You do not need to write a playlist to broadcast a single clip. If you have a series of clips to broadcast, though, you list them in sequence in the playlist. A playlist can contain any number of clips. When you run SLTA, you can play the clips in the order they're listed, or in random order (shuffle play). Refer to "Using SLTA Options" on page 236 for information about command line options that affect playlists.

Writing a Basic Playlist

In a text file (file extension .txt), list on a separate line each file that you want SLTA to stream. If the files do not reside in the same directory as the playlist, include either their full paths, or paths relative to the location of the playlist. In the following example, clips reside in the same directory as the playlist:

```
CompanyLogo.rm
Welcome.rm
President.rm
Treasurer.rm
Conclusions.rm
```

Warning! All files in the playlist must be in the same media format, and must be encoded at the same bit rate. If you use SureStream, all clips in the playlist must be SureStream clips encoded for the same set of rates. You cannot mix SureStream clips with single-rate RealAudio or RealVideo clips.

Tip: You can determine a clip's encoded bit rate or rates by opening it in RealPlayer, and giving the **View>Clip>Clip Source** command.

Adding Title, Author, and Copyright Information

Prerecorded clips often have title, author, and copyright information encoded into them. Through the playlist, you can override this information on a clip-by-clip basis. Or, you can set the same title, author, and copyright values for all clips. RealPlayer users display this information through the clip info command (**Ctrl+i**). Information about a certain clip displays only as that clip is broadcast.

Tip: Although a playlist isn't required to broadcast a single clip, you can write a playlist containing a single clip just to use the title, author, and copyright overrides.

Setting Information for All Clips

To include the same title, author, and copyright information for every clip, add `title`, `author`, and `copyright` tags to the beginning of the playlist. This information overrides any encoded information in the clip. End each tag with a colon, as shown in the following example:

```
title: Annual Report
author: Chris Lee, Executive Assistant
copyright: Copyright 2006
...clip1...
...clip2...
```

Tip: Presentation information, which you can specify in addition to clip information, is not supported directly through SLTA. You can add overall presentation information by delivering the simulated live broadcast through SMIL, however. For more information, see *Introduction to Streaming Media*.

Using Individual Clip Parameters

If you append individual title, author, and copyright parameters to a clip in the playlist, the specified information overrides the playlist values described above, as well as values encoded in the clip. You can use any combination of title, author, and copyright parameters for each clip. Use double quotation marks around values. Separate the first parameter from the clip file name with a question mark (?). Precede all subsequent parameters with an ampersand (&), as shown in this example:

```
Welcome.rm?title="Annual Report"&author="Chris Lee"&copyright="(c) 2006"
```

Mixing Information Styles

You can mix the different ways of delivering clip information. When you do this, keep in mind that clip information is used in the following order:

1. Information specified for each clip in the playlist.
2. Information specified for all clips at the start of the playlist.
3. Information encoded in each clip.

In the following playlist example, suppose that each clip has a title, author, and copyright value encoded directly in the clip:

```
copyright: (c) 2001-2006
CompanyLogo.rm?title="Welcoming Remarks"
Welcome.rm?title="Welcoming Remarks"
President.rm?title="President's Address"
Treasurer.rm?title="Treasurer's Summary"
Conclusions.rm?title="The Future is Bright"
```

Within this playlist, the title parameter for each clip overrides the titles encoded in the clips. The playlist's copyright parameter sets the same copyright value for all clips in the list, also overriding the clips' encoded values. No author values are specified, though, so each clip uses its encoded author text.

Running SLTA

This section describes the SLTA command line syntax, environment variables, and options. To run SLTA in advanced mode, you must first configure your Helix Server receiver, set up your SLTA configuration file, and, optionally, write your playlist.

Running SLTA on a Different Machine

If you run SLTA infrequently, you can use your Helix Server machine to run the SLTA executable. In a production environment in which you run SLTA continually, however, RealNetworks recommends that you install SLTA on a machine other than the Helix Server machine. This distributes the processing load between SLTA and Helix Server.

Note: To install SLTA on a different machine, contact your RealNetworks representative to obtain Helix Live Transmitter. This product installs SLTA on a selected machine or multiple machines. Note, too, that you can run SLTA and Helix Server on different operating systems.

Setting Environment Variables

Running SLTA on your Helix Server machine requires that you set two environment variables. If you plan to use SLTA on your Helix Server machine frequently, set the following environment variables permanently. This lets you run the SLTA executable file directly:

`SLTA_PLUGIN_PATH`	Full path to the directory that holds Helix Server plug-ins. This is typically the `Plugins` subdirectory of the main Helix Server installation directory.
`SLTA_SUPPORT_PATH`	Full path to the directory that holds Helix Server libraries (DLLs). This is typically the `Lib` subdirectory of the main Helix Server installation directory.

Tip: If you plan to use SLTA only occasionally, you can run the provided batch (Windows) or shell (UNIX) file described in "Starting SLTA" on page 233 to set the variables for your current session.

Setting Variables on Windows

To set SLTA environment variables permanently on a default installation of Helix Server on Windows, add the following values to your environment:

```
SLTA_PLUGIN_PATH      C:\Program Files\Real\Helix Server\Plugins
SLTA_SUPPORT_PATH     C:\Program Files\Real\Helix Server\Lib
```

If you installed Helix Server in a location other than the default, modify the `C:\Program Files\Real\Helix Server` portion of the paths above to the actual base path where Helix Server resides.

How you add variables to your environment depends on the operating system you're using. On Windows 2000, Windows XP, and Windows Server 2003, for example, you add variables with the **Start>Settings>Control Panel>System> Advanced> Environment Variables** command.

Setting Variables on UNIX

On UNIX platforms, add the environment variables to the `.profile` file or the shell personalization files (`.bashrc` or `.cshrc`) of the shell under which SLTA runs. For example:

```
SLTA_PLUGIN_PATH=/usr/local/Real/HelixServer/Plugins
export SLTA_PLUGIN_PATH

SLTA_SUPPORT_PATH=/usr/local/Real/HelixServer/Lib
export SLTA_SUPPORT_PATH
```

Starting SLTA

On Helix Server, SLTA files are located in the `Bin` subdirectory of the main Helix Server directory. The file you run depends on your operating system, and whether you've set environment variables permanently.

SLTA Executable to Use

	Windows	UNIX
With variables set permanently.	slta.exe	slta
Without having set variables permanently.	slta.bat	slta.sh

Starting SLTA in Basic Mode

From a command prompt, use the following syntax to run SLTA in basic mode.

slta[*.ext*] *server_address HTTP_port name password stream_name.ext clip.ext*|*playlist*.txt [-options]

- Use the appropriate SLTA executable file as described in the preceding table.

- Next, specify the network address (DNS name, IPv4 address, or IPv6 address) and HTTP port of your Helix Server. See "Defining Communications Ports" on page 63 for port information.
- The name and password entries give the encoder access to Helix Server. For more information, see "Helix Server Setup" on page 217.
- For *stream_name.ext*, specify the name of the simulated live stream. This name appears in links to the broadcast, and uses the appropriate media extension, such as `.rm` for a RealMedia broadcast. When broadcasting QuickTime or MPEG, use `.mov` or an appropriate MPEG extension (such as `.mpeg` or `.mp4`) for the stream name. You do not need to use an SDP file.
- Next, you specify either the single clip you want to broadcast, or the playlist you wrote according to the instructions in "Creating a Playlist" on page 229. If the clip or playlist is not in the SLTA directory, specify the full path, or a path relative to the location of the SLTA executable file.
- Optionally, you can use command line options as described in "Using SLTA Options" on page 236. For example, SLTA loops a clip or playlist by default. With command line options, you can specify how many repetitions to play.

Basic Mode Example

If you set the SLTA environment variables permanently on Helix Server, you would use a command such as the following to broadcast a single RealMedia archive (`awards.rm`) as a simulated live event (`encore.rm`), using no additional SLTA options:

```
slta.exe helixserver.example.com 80 simulator k56weiq9 encore.rm awards.rm
```

Notes on Running SLTA in Basic Mode

- As SLTA begins the broadcast, it displays the progress indicator described in "Monitoring and Stopping SLTA" on page 239.
- SLTA contacts the receiver immediately to deliver the stream.
- You can verify connections on Helix Server using the Server Monitor described in Chapter 17.

Starting SLTA in Advanced Mode

From a command prompt, use the following syntax to run SLTA in advanced mode.

slta[*.ext*] -c *config_file*.cfg *stream_name.ext clip.ext*|*playlist*.txt [-options]

- Use the appropriate SLTA executable file as described in "Starting SLTA" on page 233.
- For *config_file*.cfg, give the full or relative path to a configuration file you created according to instructions in "Configuring SLTA for Advanced Mode" on page 222.
- For *stream_name.ext*, specify the name of the simulated live stream. This name appears in links to the broadcast, and uses the appropriate media extension, such as .rm for a RealMedia broadcast. When broadcasting QuickTime or MPEG, use .mov or an appropriate MPEG extension (such as .mpeg or .mp4) for the stream name. You do not need to use an SDP file.

  If you defined virtual paths for receivers, as described in "Directing Streams through Paths" on page 226, you can include a virtual path in front of the stream name, as in news/live.rm. This path, which does not correspond to any directory on the machine, uses a Web-style forward slash even on Windows.
- Next, you specify either the single clip you want to broadcast, or the playlist you wrote according to the instructions in "Creating a Playlist" on page 229. If the clip or playlist is not in the SLTA directory, specify the full path, or a path relative to the location of the SLTA executable file.
- Optionally, you can use command line options as described in "Using SLTA Options" on page 236. For example, SLTA loops a clip or playlist by default. With command line options, you can specify how many repetitions to play.

Advanced Mode Example

If you set the SLTA environment variables permanently on Helix Server, you would use a command such as the following to broadcast a single RealMedia archive (awards.rm) as a simulated live event (encore.rm), using no additional SLTA options:

```
slta.exe -c transmit.cfg encore.rm awards.rm
```

Notes on Running SLTA in Advanced Mode

- As SLTA begins the broadcast, it displays the progress indicator described in "Monitoring and Stopping SLTA" on page 239.
- If you set up push splitting, SLTA contacts the receiver immediately to deliver the stream.

- If you're using pull splitting, SLTA does not deliver the stream until the receiver requests it.
- You can verify connections on the Helix Server receiver using the Server Monitor described in Chapter 17.

Running Redundant SLTA Encoders

In both basic and advanced modes, SLTA supports redundant encoders as described in "Using Broadcast Redundancy" on page 135. This allows the Helix Server receiver to switch to a backup stream if the primary encoder stream fails. Although you can run two instances of SLTA on the same machine, this provides process redundancy only. For true redundancy, run an instance of SLTA on two separate machines that use different power supplies and network connections.

To set up SLTA redundancy, install SLTA from the Helix Live Transmitter package on your backup machine. Copy over the configuration file, the playlist (if used), and the broadcasted clip or clips. Because the content is prerecorded, you want to start SLTA on both machines at the same time, or as close as possible. You may want to set up a script appropriate for your operating system that launches both SLTA processes on the two machines simultaneously.

The only difference between the two SLTA instances is the stream name used in the command that starts SLTA. Each stream name needs a delimiter that identifies it as the primary or the backup. The following example builds on an example in a preceding section by appending `.1` to the stream name to identify it as the primary stream:

```
slta.exe -c transmit.cfg encore.rm.1 awards.rm
```

With SLTA on your backup machine, you add `.2` to the stream name:

```
slta.exe -c transmit.cfg encore.rm.2 awards.rm
```

Using SLTA Options

SLTA provides optional command-line arguments that affect the simulated live broadcast. The options appear last on the command line in any order, each option preceded by a hyphen. Here is an example:

```
slta.exe -c transmit.cfg encore.rm playlist.txt -e -n10 -r
```

The following table summarizes the options. Some options are useful only with playlists. Unless noted otherwise, all options are available in basic or advanced mode.

SLTA Command-Line Options

Option	Function	Reference
c	Use the specified configuration file (advanced mode only)	page 234
e	Disable playlist title, author, and copyright information.	page 238
f	Force playlist title, author and copyright updates when broadcasting RealMedia clips.	page 238
n*N*	Play *N* files, then stop, where *N* is an integer that specifies the number of files to play. Using just -n sets up a continuous loop, which is the default behavior.	page 237
r	Transmit playlist files in a random order (shuffle play).	page 237
t	Force TCP transport instead of UDP (basic mode only).	page 238
w	Enable wallclock synchronization with other streams.	page 238

Modifying the Playback Order

By default, SLTA transmits a clip or playlist in a continuous loop. If you specify just one file, for example, SLTA continuously transmits that file until you stop the broadcast. You can use command line -n*N* and -r options to modify the default playback.

Specifying the Number of Clips to Play

The -n*N* switch specifies the total number of files to play. To play a clip just once, for instance, you specify -n1. To play it twice, use -n2. For a single playback of a playlist that contains 3 clips, for example, you specify -n3. A value such as -n5 means to play the single clip five times, or to play the first five clips on the playlist (not to play each clip in the playlist five times). If you use -n5 for a playlist that has just three clips, SLTA plays the three clips in order, then returns to the start of the list to play the first two clips again, for a total of five clips played.

Changing a Playlist During a Broadcast

When SLTA loops a playlist, either continuously (the default behavior) or a predetermined number of times because of the -n*N* option, it reads the playlist each time it starts a loop. This allows you to change the playlist during the broadcast. Simply edit the existing playlist, or replace the playlist with another

of the same name. SLTA uses the modified playlist after it has played all of the clips in the current playlist.

> **Tip:** This feature lets you stream long broadcasts without writing a single, long playlist. Your first playlist might last an hour. During that hour of the broadcast, you can change the playlist to specify the clips that play during the second hour, and so on.

Using Shuffle Play

When you use a playlist, you can create shuffle play by including the `-r` option. If you use just this option without `-n`*N*, the playlist cycles continuously, playing clips in a random order during each cycle. You can use both the `-r` and `-n`*N* switches to cycle randomly through the playlist a certain number of times. For example, `-r` and `-n10` cause SLTA to transmit a playlist of five clips twice, with clips playing in a random order each time.

Disabling Title, Author, and Copyright Overrides

If you use the `-e` option when broadcasting any media format, SLTA does not use the title, author, and copyright information included in the playlist. The media player still displays any title, author, and copyright information encoded in the clips, however.

The `-f` option preserves playlist overrides, which are described in "Adding Title, Author, and Copyright Information" on page 230. You generally do not need to include this option because it is the default for RealMedia broadcasts. Additionally, using this option when broadcasting content in a format other than RealMedia may cause the broadcast to fail.

Using TCP Transport

In basic mode, SLTA opens a UDP connection to Helix Server unless you use the `-t` option to force a TCP connection. Although TCP provides a more reliable connection in a lossy environment, it increases the network overhead. In advanced mode, the `-t` option is not available because you specify TCP or UDP transport in the SLTA configuration file.

Synchronizing to a Wallclock

When you include the `-w` option, SLTA sets the first timestamp of the packet stream to the time defined by the clock on the SLTA computer. You can then use the SMIL wallclock timing feature to synchronize events in a SMIL

presentation to specific times in the broadcast stream. For example, you might pop up an advertisement at 2:35 P.M. as recorded by the SLTA computer clock.

> **For More Information:** For information about the SMIL wallclock feature, see the advanced timing chapter of *RealNetworks Production Guide*.

Monitoring and Stopping SLTA

As SLTA runs, it displays the name of the file it is currently transmitting. A line of asterisks indicates the percentage of the file that has been sent. For example, a row of asterisks that lines up below the number `5` indicates that the file is approximately 50 percent complete. Here is an example:

```
Transmitting Welcome.rm...
0----1----2----3----4----5----6----7----8----9----10
**********************
```

SLTA stops automatically and displays the text `Done` when it has finished transmitting all of the files according to the playlist (if used) and the specified options. If it cannot transmit a file in a playlist because it cannot find the file, or the file is encoded in a format different from the preceding files, it skips that file, prints an error message, and transmits the next file.

Ordinarily, you will not need to stop SLTA manually except to halt the broadcast before it completes, or to terminate a continuously looping broadcast. To stop SLTA, press **Ctrl+c** at the command line from which you started SLTA on Windows. On UNIX, use the `kill` command with the process ID of the SLTA process.

Linking to the Simulated Live Broadcast

The following sections explain simulated live broadcast link formats in general, and provide example links for RealMedia broadcasts. For broadcasts in other formats, you need to use transmitter mount points and stream names as appropriate for that format.

Writing Basic Mode Links

Links to SLTA broadcasts in basic mode use the unicast URL formats described in “Linking to Unicasts” on page 151.

Linking from a Web Page

A Web page link to a RealMedia broadcast in which Helix Server uses the default port 80 for HTTP might look like this:

```
http://helixserver.example.com/ramgen/broadcast/archive.rm
```

- The /ramgen/ mount point launches RealPlayer, as explained in "Using a Client Launch Utility" on page 90. For a Windows Media broadcast, you use /asxgen/.
- The normal broadcast mount point is /broadcast/, which you use for all media formats delivered by SLTA. If you are using redundant SLTA transmitters, use the /redundant/ mount point instead. This default mount point can be changed as described in "Modifying Encoder Redundancy Settings" on page 136.
- The requested file, shown above as archive.rm, is the stream name you specify when starting SLTA. You do not use the actual name of the broadcasted clip or playlist.

Linking from a Ram or SMIL File

You can also launch a simulated, live RealMedia broadcast through a Ram file or SMIL file, as described in "Using Metafiles" on page 88. The URL format is similar to the preceding example, but it specifies RTSP or MMS, and omits the /ramgen/ or /asxgen/ parameter. In the following example, the port number is omitted, which means that port 554 is used for RTSP:

```
rtsp://helixserver.example.com/broadcast/archive.rm
```

Creating Push Splitting Links

Links for advanced mode broadcasts use the same format as split broadcasts, which are described in "Linking to Split Content" on page 208. Push splitting links point to the Helix Server receiver, but include the name of the SLTA transmitter to identify the broadcast.

Linking from a Web Page

A Web page link to a RealMedia broadcast in which Helix Server uses the default port 80 for HTTP might look like this:

```
http://helixserver.example.com/ramgen/broadcast/Simulated/news/archive.rm
```

- The /ramgen/ mount point launches RealPlayer, as explained in "Using a Client Launch Utility" on page 90. For a Windows Media broadcast, you use /asxgen/.
- The default broadcast mount point is /broadcast/, but the Helix Server receiver may define a different mount point. If you are using redundant SLTA transmitters, use the /redundant/ mount point instead. This default mount point can also be changed, as described in "Modifying Encoder Redundancy Settings" on page 136.
- Following the broadcast mount point, you specify the transmitter name, such as /Simulated/. The name is set in the SourceName variable of the configuration file, as explained in "Setting Basic Transmitter Properties" on page 223.
- A path name such as news/ is optional. You include it only if you've specified a value other than "*" for the PathPrefix variable in the configuration file. For more information, see "Directing Streams through Paths" on page 226.
- The requested file, shown above as archive.rm, is the stream name you specify when starting SLTA. You do not use the actual name of the broadcasted clip or playlist.

Linking from a Ram or SMIL File

You can also launch a simulated, live RealMedia broadcast through a Ram file or SMIL file, as described in "Using Metafiles" on page 88. The URL format is similar to the preceding example, but it specifies RTSP or MMS, and omits the /ramgen/ or /asxgen/ parameter. In the following example, the port number is omitted, which means that port 554 is used for RTSP:

```
rtsp://helixserver.example.com/broadcast/Simulated/news/archive.rm
```

Writing Pull Splitting Links

As with push splitting, a pull splitting link requests the stream from the receiver, and indicates the transmitter where the broadcast originates. The link does not use the transmitter name given in the SLTA configuration file, however. Instead, it provides the address and listen port of the transmitter so that the receiver can locate the transmitter and pull the broadcast stream.

Linking from a Web Page

A Web page link to a RealMedia broadcast in which the Helix Server receiver uses the default port 80 for HTTP might look like this:

```
http://helixserver.example.com/ramgen/broadcast/pull/
simulator.example.com:2030/news/archive.rm
```

- The /ramgen/ mount point launches RealPlayer, as explained in "Using a Client Launch Utility" on page 90. For a Windows Media broadcast, you use /asxgen/.
- The default broadcast mount point is /broadcast/, but the Helix Server receiver may define a different mount point. If you are using redundant SLTA transmitters, use the /redundant/ mount point instead. This default mount point can also be changed, as described in "Modifying Encoder Redundancy Settings" on page 136.
- Following the broadcast mount point, you specify the pull splitting mount point defined on the Helix Server receiver, such as /pull/. The name is set in the receiver definition, as explained in "Enabling Pull Splitting Requests" on page 207.
- After the pull splitting mount point, you give the transmitter's address and listen port. The example above is /simulator.example.com:2030/. The SLTA configuration file defines the listen port, as described in "Setting Pull Configuration Values" on page 228.

  Note: When broadcasting Windows Media, you need to mask the address and port through an alias, as described in "Using URL Aliases" on page 212.

- A path name such as news/ is optional. You include it only if you've specified a value other than "/" for the PathPrefix variable in the configuration file. For more information, see "Setting Pull Configuration Values" on page 228.
- The requested file, shown above as archive.rm, is the stream name you specify when starting SLTA. You do not use the actual name of the broadcasted clip or playlist.

Linking from a Ram or SMIL File

You can also pull a RealMedia broadcast through a Ram file or SMIL file, as described in "Using Metafiles" on page 88. The URL format is similar to the

preceding example, but it specifies RTSP or MMS, and omits the /ramgen/ or /asxgen/ parameter. In the following example, the receiver's RTSP port number is omitted, which means that port 554 is used for RTSP:

```
rtsp://helixserver.example.com/broadcast/pull/simulator.example.com:2030/
news/archive.rm
```

> **Note:** When broadcasting Windows Media, you need to mask the address and port through an alias, as described in "Using URL Aliases" on page 212.

PART

V

SECURITY

The server administrator must ensure security for the network where Helix Server resides, as well as consider the needs of media clients that may be behind restrictive firewalls. The following chapters help you handle these security issues.

CHAPTER

11

FIREWALLS

Firewalls may present communications problems to Helix Server. This chapter helps you to find solutions to these problems. It first provides background on firewalls and network protocols. It then recommends ways to work with firewalls to give viewers the best possible streaming media experience. Finally, it lists the communications ports that RealNetworks components use.

> **Note:** You may also want to consult the firewall information on the RealNetworks technical support Web site at **http://www.service.real.com/firewall**.

Understanding Firewalls

A firewall is a software program or device that monitors, and sometimes controls, all transmissions between an organization's internal network and the Internet. However large the network, a firewall is typically deployed on the network's edge to prevent inappropriate access to data behind the firewall. The firewall ensures that all communication in both directions conforms to an organization's security policy.

Firewall technologies are configurable. You can limit communication by direction, IP address, protocol, ports, or numerous other combinations. Firewalls positioned between your Helix Server and other computers may cause communication failures if the firewall does not allow for the types of communication Helix Server requires. These other computers may be media clients or servers set up as encoders or receivers.

If you have access to the firewall, you can configure it to enable the ports, protocols, and addresses that optimize Helix Server communication. In some cases, however, your organization's security policy may prevent optimal streaming. For example, if a client is behind a firewall that permits only one-way, outbound access to the Internet, that client would not efficiently receive

clips streamed by Helix Server over the Internet. This is because the client needs to establish a two-way connection to achieve optimal streaming results.

Protocol Layers

A protocol is a language that computers use when communicating over a network. The Transmission Control Protocol/Internet Protocol—commonly called TCP/IP—encompasses a suite of protocols upon which the Internet is built. TCP/IP protocols work on a layering principal, in which each layer is assigned a specific network task.

For communication to occur, a source computer sends a message from its highest network layer to its lowest. The lowest network layer at the source forwards the message over the network. When the message arrives at the destination computer, it must pass through the exact same layers, but in reverse order.

Each network layer uses specific protocols to perform its task. Packets passed down from upper layers are tucked inside lower layer packets. This is called *encapsulation*. By encapsulating packets, a layer can handle its responsibilities without understanding the preceding layer. Through this layering scheme, a destination layer on one computer receives exactly the same object sent by the corresponding source layer on another computer.

For example, an application such as a Web browser packages data, such as a Web page request made over HTTP, at the application layer, passing it to the lower transport layer. There, the HTTP request packets are bundled into TCP packets that are then delivered to destination Web server. When the Web server receives the source TCP packets, it strips off the TCP shells, and bumps the HTTP message up to the destination computer's application layer. This layer, in turn, delivers to the HTTP-based request to the Web server.

> **Note:** Network layering is a complex topic. This section omits discussion of additional layers required to deliver packets over a network, focusing instead on the transport and application layers, and the protocols relevant to streaming media for each.

Transport-Layer Protocols

All transport-layer protocols transfer data between hosts. The transport-layer protocol in use can greatly affect the quality of the stream received. There are two main transport protocols used on IP networks: TCP and UDP. Helix

Server utilizes both of these protocols, and the choice of protocol is generally negotiated automatically by the servers and clients involved.

Transmission Control Protocol (TCP)

Helix Server can use TCP in a number of ways. Because TCP offers a single channel for bi-directional communication, Helix Server uses it as a control channel to communicate with clients about passwords and user commands such as pause and fast-forward. The TCP protocol also guarantees delivery of packets, and has built-in congestion control that helps to provide reliable communication.

On the down side, TCP responds slowly to changing network conditions, and creates network overhead through its error checking facility. For this reason, TCP is best suited for delivering low-bandwidth material like passwords or user commands. In some cases, TCP can facilitate communication through a firewall. For example, firewalls that block UDP traffic between Helix Server and its clients may permit TCP connections.

User Datagram Protocol (UDP)

Helix Server uses UDP packets to deliver data to client software on its data channel. Client software sends UDP-based requests to Helix Server when packets on the data channel have not arrived. Because the transport does not consume as much network overhead, it can deliver packets faster than TCP.

Because video and audio data typically consume large amounts of bandwidth, it makes the most sense to use UDP to deliver streaming media. For this reason, Helix Server uses UDP as the default for transmitter-to-receiver communication. With RealProducer, you can set up the encoder to act as a transmitter. You therefore can, and should, use a UDP connection to deliver live feeds from the encoder.

Application-Layer Protocols

Helix Server uses four application-layer protocols to deliver streaming media to clients: RTSP, MMS, and HTTP. The following table summarizes their use.

Application-Layer and Transport-Layer Protocols

Player Software	Application Protocol	Transport Options
RealPlayer, QuickTime Player, MPEG players	RTSP	TCP and UDP, or TCP only
Windows Media Player	MMS	TCP and UDP, or TCP only
Windows Media Player, for streaming media and protocol roll-over	HTTP	TCP only
RealPlayer, for HTTP cloaking	HTTP	TCP only

Real-Time Streaming Protocol (RTSP)

A standards-based protocol designed for serving multimedia presentations, RTSP is very useful for large-scale broadcasting. Only RTSP can deliver SureStream files with multiple bit-rate encoding. SMIL, RealText, and RealPix also require RTSP. RTSP uses TCP for player control messages, and UDP for video and audio data. RTSP can also use TCP to deliver data, but this is not recommended. Use RTSP with RTSP-compatible players such as RealPlayer, MPEG players, and QuickTime Player.

Microsoft Media Services (MMS)

The MMS protocol is designed specifically for serving multimedia presentations. Although it is not standards-based, you can use it to broadcast live or on-demand Windows Media clips to Windows Media Player. MMS uses TCP for player control messages, and UDP for video and audio data. MMS can also use TCP to deliver data, but this is not recommended.

HyperText Transfer Protocol (HTTP)

HTTP is typically used for Web pages. With Helix Server, HTTP is used to display Helix Administrator pages and HTML-based documentation. It is also used for requesting metafiles that point client software to streaming media content. HTTP may also be used in HTTP cloaking, which is a method of delivering streaming media to clients behind firewalls that restrict streaming media protocols.

Note: Helix Server also uses HTTP to transport live Windows Media streams from Windows Media Encoder to Helix Server.

Packet Formats

All Internet data is delivered in IP packets. But just as TCP or UDP can wrap a control protocol for streaming media, IP packets can wrap data packets in formats designed to deliver streaming media data. Helix Server can use both of the following packet formats.

RealNetworks Data Transport (RDT)

When Helix Server communicates to RealPlayer over RTSP, it uses RDT as the packet format. A proprietary format, RDT allows the use of RealMedia features such as SureStream.

Real-Time Transport Protocol (RTP and RTCP)

RTP is a standards-based packet format designed as the companion to the RTSP protocol. QuickTime Player, for example, uses RTP as its packet format. Helix Server fully supports RTP, and shifts to RTP automatically when streaming to an RTP-based client. RealPlayer also supports RTP, using this format when receiving data from RTSP/RTP servers. Helix Server also supports RTCP, a control protocol used for monitoring and control of RTP sessions.

Firewalls and Helix Server Features

The following table describes how firewalls affect Helix Server features.

Firewalls and Helix Server Features

Feature	Effect of a Firewall
On-Demand Streaming	Issues that clients may have in connecting to on-demand streams are described in "Streaming to Client Software Behind Firewalls" on page 256.
Live Unicasting	Clients connect to live broadcasts in the same way they connect to on-demand streams. However, the encoder that supplies Helix Server with its live data may not be able to connect to Helix Server if a firewall exists between the encoder and Helix Server. See "Working With An Encoder, Receiver, or Proxy" on page 255.

(Table Page 1 of 2)

Firewalls and Helix Server Features

Feature	Effect of a Firewall
Splitting	Working with receivers that are located on the other side of a firewall requires special consideration. Chapter 9 covers this issue.
Multicasting	Multicasts usually take place within an intranet, where broadcasts are not travelling outside a firewall. If a multicast is occurring through a firewall, the firewall must be specially configured to allow multicast traffic.
Helix Proxy	Helix Proxy connects to your Helix Server just as any other client would.
Access Control, and Reporting	When a firewall exists between a client and Helix Server, the IP address that appears in the basic access log's *client_IP_address* field may not be the true client address, and you might not get an accurate idea of exactly which clients are viewing material streamed by your Helix Server.
Authentication	Requests by Helix Server for authentication information (either from the user or the client software) is delivered over the control channel. If a firewall prevents the control channel connection, Helix Server cannot authenticate the request and therefore will not deliver it.
ISP Hosting	If there is a firewall between users and the location where they are to store their content for hosting, they may not be able to send their clips to Helix Server.

(Table Page 2 of 2)

Placing Helix Server in a Network

One of the first decisions to make about deploying Helix Server is where it should live on your network. If you are streaming content only to clients inside your organization, typically it is best to locate your Helix Server inside the firewall. This requires no special configuration for Helix Server or the firewall. Clients on the Internet side of the firewall, however, most likely will not be able to access your Helix Server.

If you need to stream content to clients on the Internet, it's better not to locate Helix Server behind a firewall. For optimal streaming, Helix Server needs to use streaming protocols, and to process incoming and outgoing UDP connections on a variety of ports. Although you may be able to change your organization's security policy to enable optimal communication, this may hamper the effectiveness of the firewall.

The best solution may be to create a perimeter network, sometimes known as a *De-Militarized Zone* (DMZ), and move Helix Server there. In this scenario, you fortify the connection between main and perimeter networks, but allow a less stringent security policy in the perimeter. This keeps the main network secure, while maintaining optimal connections from the Internet to your Helix Server.

Working with Firewall Technologies

In some organizations, the only choice is to place Helix Server behind the firewall, and then stream media through the firewall. If this is your situation, you must configure your firewall to enable optimal communication with Helix Server and other computers. If not, communication may be substandard or even impossible.

The following sections describe the requirements for optimal communication in various situations. They will help you to decide what types of traffic to allow through your firewall. The section "Default Ports" on page 259 lists the specific ports that Helix Server uses to communicate with other computers. Keep in mind that the values listed there are defaults, many of which can be changed to suit your needs.

Limiting Incoming UDP Traffic

Helix Server enables you to limit the number of ports used for incoming UDP packets from clients such as RealPlayer. The client sends these packets to acknowledge data reception, and to request a resend of lost packets. Although a firewall can accept all incoming UDP communication, network administrators often hesitate to leave open a large number of UDP ports. Service quality degrades, though, if Helix Server cannot receive these packets. To solve this problem, Helix Server can redirect replies from all clients to a few UDP ports, thus limiting the number of open UDP ports.

> **For More Information:** For instructions on how to set up this feature, see "Changing Port Assignments" on page 64.

Working with a NAT Firewall

If your Helix Server resides behind a Network Address Translation (NAT) firewall, links to streaming media may fail because the media player attempts to communicate to the firewall rather than the server. The section "Streaming

through NAT Firewalls" on page 92 explains how to work around this problem.

Working With A Virtual IP Address

Firewalls are not the only type of technology that can affect where you place Helix Server. You might also need to place a cluster of Helix Servers behind a virtual IP address. Although this network topology is possible, its implementation may have unintended consequences for RTSP traffic behind a restrictive firewall.

How Virtual IP Addressing Works

A typical IP address resolves to a single server. A virtual IP address, on the other hand, resolves to a cluster of servers, typically through a hardware switch. Consider the case of a cluster of servers set behind a hardware switch, with all servers sharing the same content, but configured with unique, private IP addresses. In this scenario, only the hardware switch is assigned a public IP address. It receives all public communication, passing each request to a host in the cluster, based on load.

The Problem with Virtual Addressing

A problem arises from the combination of public and private IP addresses. If a firewall blocks streaming media protocols, the client communicates through *HTTP cloaking*. In most cases, this effectively bypasses firewall security, which typically allows HTTP traffic to pass. The section "HTTP Cloaking" on page 257 discusses this strategy in detail. For now, it is important only to grasp that for cloaking to work, the client must be able to make *two* HTTP connections to the *same* Helix Server.

When a client uses HTTP cloaking, Helix Server replies to the initial HTTP connection with its actual IP address, and not the virtual IP of the cluster. This allows the client to circumvent the hardware switch, and establish its second HTTP connection directly with the Helix Server handling the request. But if that server uses a private IP address, the client cannot make the second, necessary connection, and HTTP cloaking fails.

Resolving the Virtual Addressing Problem

There are two ways to resolve the virtual addressing problem:

- Configure firewalls to allow connections by streaming media protocols. This is the ideal solution, because communication between server and clients will be the most efficient.

- Use globally routable IP addresses on all hosts behind a virtual IP address. This way, clients can make HTTP-cloaked requests behind firewalls that restrict streaming media protocols. This requires your organization to register for a larger number of public IP addresses, however.

If neither of these solutions is possible, some RTSP clients outside of highly restrictive firewalls may not be able to access your content.

Working With An Encoder, Receiver, or Proxy

Once you have placed Helix Server in relation to your firewall, you need to consider the placement of other servers. These servers might be other Helix Servers set up as receivers or relays, or they may be computers running encoding or media proxy software. Generally, the same rules and limitations discussed in the preceding sections apply to placing these types of servers as well. If possible, place other encoders or receivers in the perimeter network along with Helix Server. If that's not possible, solutions depend largely on the type of server you're considering.

Communicating With Encoders

This section assumes that you've placed Helix Server behind your firewall, and have taken steps to ensure optimal communication with encoders outside your firewall. Even if your firewall is optimally configured, another organization's firewall may cause problems.

Suppose that another organization broadcasts a live feed to your Helix Server from behind their own restrictive firewall. The best solution is to contact that organization and have them move their encoder from behind the firewall. Alternatively, you could agree with that organization to open the required network paths. If neither of the preceding solutions is feasible, you can attempt an encoder connection using TCP.

HTTP Broadcasts from Windows Media Encoder

When Windows Media Encoder delivers a live Windows Media stream, the only protocol option is HTTP over TCP. With pull broadcasting, Helix Server pulls the live feed from the encoder. Contact the administrator of the Windows Media Encoder for the HTTP port number to pull from. Helix Server receives the live feed on its default HTTP port.

With push broadcasting, Windows Media Encoder delivers the HTTP stream to a predefined port on Helix Server. The section "Setting up a Windows Media Push Broadcast" on page 145 explains how to define the push encoder port.

Communicating With Receivers

By default, receivers and Helix Servers use UDP to communicate. An option is available for them to use TCP instead. If the receiver is behind a firewall, you should move the receiver to the perimeter network. If this is not possible, change the transport protocol used for transmitter-to-receiver communication to TCP as described in Chapter 9, "Transmitters and Receivers".

Communicating With Helix Proxies

Helix Proxy servers commonly work behind a firewall. In this respect, a Helix Server-to-Helix Proxy connection behaves like a Helix Server-to-client connection, except for HTTP communication. Helix Proxy first tries to connect with RTSP using UDP for data transport. If the firewall prohibits UDP connections, Helix Proxy tries RTSP, using TCP for data transport. Helix Proxy has no option for HTTP delivery. Thus, if the firewall prohibits RTSP, Helix Proxy will not be able to proxy streams on behalf of clients.

Streaming to Client Software Behind Firewalls

This section describes how client technologies communicate to Helix Server when the streaming media viewer resides behind a firewall. It provides suggestions for setting up Helix Server to accommodate client connection features.

Channel Negotiation

Helix Server uses two connections, known as *channels*, to communicate with clients:

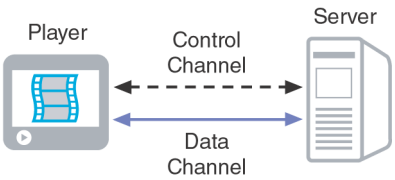
- control channel

  Helix Server uses this channel for communication with the client. Over this channel, Helix Server requests and receives passwords, and clients send instructions such as fast-forward, pause, and stop.

- data channel

  The media clips themselves are streamed over a separate *data channel*.

Helix Server Communication with RealPlayer



At the transport layer, most media players, including RealPlayer, can work around situations in which the first communication fails because the player resides behind a firewall that blocks the preferred protocol. The primary strategy involves shifting automatically to protocols and delivery methods that aren't blocked. Typically, the client shifts the control channel to the less efficient TCP, which is less likely to be blocked than UDP.

If the control connection is established, the client then negotiates the data channel. Optimally, the data channel will use the more efficient UDP transport. If the stream is live, some client software, including RealPlayer, attempts to set up a UDP multicast first. If this method fails, the client next attempts UDP unicast. And if that fails, the client uses the established control channel for data. In short, the client tries to set up the most optimal data delivery method, relying on the control connection as a last resort.

> **For More Information:** Unicasting and multicasting are described in Chapter 7 and Chapter 8, respectively.

HTTP Cloaking

Some firewalls restrict streaming media protocols like RTSP or MMS, and the client cannot establish the control connection. In these cases, the client and server disguise streaming media traffic as HTTP, a solution known as *HTTP cloaking*. Because most firewalls allow HTTP traffic, this solution circumvents the communication problem. However, HTTP is not designed for streaming media, and the user does not get the highest quality stream.

The HTTP cloaking method must also work around limitations in the HTTP protocol. For example, RTSP clients use two HTTP streams to connect to a single Helix Server. Because the client initiates both streams, the client firewall typically allows these connections as outgoing HTTP traffic. The first

connection uses the HTTP `GET` method, the standard means for a browser to request a Web page. At the receiving end, Helix Server strips off the HTTP disguise, using the encapsulated RTSP information to determine what information to send the client.

Helix Server must then wait for the second HTTP connection from the same client to proceed with streaming the media. This second connection uses the HTTP `POST` method, the standard means for a Web server to send data to a browser. Once both of these client-initiated streams are established, the client and Helix Server can pass RTSP packets in two directions, through a firewall that blocks RTSP.

Port 80 For HTTP Traffic

For HTTP cloaking to work, the client must connect to Helix Server's HTTP port. It has no problem if Helix Server uses the well-known port 80 for HTTP. But problems can arise if Helix Server uses a different HTTP port. Even if the client knows which port to use, the client will not be able to connect if its firewall restricts outgoing HTTP traffic to port 80.

For this reason, RealNetworks recommends setting the HTTP port on Helix Server to port 80. This configuration offers the widest possible support of all types of player software. If this is not possible, RealPlayer can often discover the Helix Server HTTP port, and will be able to connect to the HTTP port as long as a firewall does not block their outbound communication. Not all players can do this, however.

> **For More Information:** When you install Helix Server and a Web server on the same machine, you need to take certain precautions before assigning port 80 to Helix Server. For more information, see "Web Servers and Helix Server" on page 42.

Port Hinting

Port hinting offers a solution for Helix Servers that cannot use default port values. It allows Helix Server to send the port numbers in use to RealPlayer when you launch clips using the Ramgen utility. This feature is turned on by default. Content creators can also define ports to try on the Helix Server machine when defining Ram files.

> **For More Information:** See "Handling Communication through Nonstandard Ports" on page 65 for more information about these features.

Default Ports

The following sections explain the default ports used by Helix Server when receiving requests and sending data. This information will help you to decide which ports to open on your firewall. Note that many port values, such as the RTSP and HTTP ports, are configurable.

> **For More Information:** See also the RealNetworks firewall information at **http://service.real.com/firewall**. For more about changing default port values, refer to "Defining Communications Ports" on page 63.

Media Players

The following table lists the default ports that Helix Server uses to listen for media player requests.

Default Listen Ports for Media Player Requests

Activity	Port Number	Transport	Purpose
listen on	554	TCP	Control channel for RTSP requests. Data channel also, if TCP was requested.
listen on	80	TCP	HTTP requests, as well as RTSP and MMS cloaked through HTTP.
listen on	1755	TCP or UDP	TCP control channel for MMS requests. Data channel also, if TCP was requested. UDP resend requests by MMS.
listen on	6970-32000	UDP	Data channel for RealNetworks and RTP-based media players.
listen on	34445-34459	UDP	RDT/RTP client replies for UDP resends, etc.
listen on	1024-5000	UDP	Data channel for MMS media players.
listen on	1-65000	Multicast	MMS multicast.

When Helix Server accepts a stream request, it directs outgoing data to a port specified by the media player. The following table lists the data port ranges

used by media players. A firewall should not restrict outgoing data sent to these client ports.

Default Data Ports on Media Players

Activity	Port Number	Transport	Purpose
send to	6970-32000	UDP	Packet delivery for RealNetworks media players.
send to	1024-65535	UDP	Packet delivery for RTP-based media players.
send to	1024-5000	UDP	Packet delivery for MMS media players.
send to	1-65000	Multicast	Packet delivery for MMS multicast.

Note: If the client chooses TCP for the data channel, Helix Server uses the same port for both the control channel and the data channel.

Tip: In addition to these settings, RealPlayer inherits proxy settings (if any) from the default browser. Users can turn off this feature through the RealPlayer **Preferences** menu, however.

Splitting

If Helix Server is set up as a transmitter or receiver as explained in Chapter 9, it uses the following default ports. A receiver that fulfills requests from media players also uses the ports described in the section "Media Players" on page 259.

Default Transmitter and Receiver Ports

Activity	Port Number	Transport	Purpose
transmitter listen on	2030	TCP or UDP	Monitor pull splitting requests from downstream receivers.
receiver send to	2030	TCP or UDP	Issue pull-splitting requests to upstream transmitters.
transmitter send to	30001-30020	TCP or UDP	Push data to receivers.
receiver listen on	30001-30020	TCP or UDP	Receive push requests and data.

Helix Proxy

The following tables list the default ports to open to allow Helix Server to communicate with Helix Proxy.

Default Ports Used with Helix Proxy

Activity	Port Number	Transport	Purpose
listen on	554	TCP	Control channel for RTSP requests from Helix Proxy version 9 and later.
listen on	3030	TCP or UDP	Data and control channel for pull-splitting requests from RealSystem Proxy version 8 and earlier.
listen on	7802, 7878	TCP	Cache requests from RealSystem Proxy version 8 and earlier.
send to	6970-32000	UDP	Proxy data channel.

Live Stream Encoders

The following table lists the ports that Helix Server uses to receive a live stream from an encoder.

Default Ports for Live Stream Encoders

Activity	Port Number	Transport	Purpose
listen on	4040	TCP	Control channel for RealProducer version 6 and 6.1 connections. Data channel also, if TCP is used as the data transport.
listen on	6970-32000	UDP	Data channel for RealProducer 6 and 6.1.
listen on	50001-50050	TCP or UDP	Media packet reception from RealProducer in account-based transmitter mode.
listen on	80	TCP	Negotiate transmitter settings for RealProducer in account-based transmitter mode. HTTP connection for pull broadcasting with Windows Media Encoder. HTTP connection for push broadcasting with Windows Media Encoder.

Note: RealProducer can also use encoding methods that imitate a stream sent by a Helix Server transmitter. If you have set up RealProducer to function this way, the ports listed in the section "Splitting" on page 260 are used.

Helix Administrator and Content Caching

The following table lists the default ports that Helix Server uses to receive requests from administrative tools and content caching subscribers.

Default Ports for Administrative Tools and Content Subscribers

Activity	Port Number	Transport	Purpose
listen on	admin port (random)	TCP	Helix Administrator requests.
listen on	9090	TCP	Server Monitor data.
listen on	554	TCP	Media distribution requests.

CHAPTER

12

ACCESS CONTROL

Using the access control feature, you can block requests to Helix Server by media clients, encoding software, and other servers, based on the IPv4 or IPv6 address of the requesting machine and the Helix Server port to which the request is made. This chapter explains how to set up access control rules.

> **Note:** To implement user name and password control for media clients, refer to Chapter 13. The section "Controlling Connections" on page 73 explains how to limit connections according to outgoing bandwidth use, total number of players, and other general criteria.

Understanding Access Control

The access control feature associates permission to connect to certain ports with client addresses. For example, you can allow only certain groups in your organization to view clips by giving those groups' IP addresses access to application protocol ports on Helix Server. If a media player requests a clip through a port for which it has no access, it receives a message that the URL is invalid, or that the connection has timed out.

Rule Components

Helix Server uses rules to implement access control policy. Each access rule provides the following information:

- **Sorting Order**—Order in which a rule is implemented. Helix Server implements access rules in order, from the first to the last. This is important to keep in mind when establishing the order in which you wish your rules to apply.
- **Access**—Whether the client is allowed or denied access.

- **Client IP Address**—Client's address, or a range of addresses. This can also be an encoder's IP address.
- **Server IP Address**—Helix Server's address.
- **Ports**—Port numbers on Helix Server to which access is allowed or denied. For general content viewing, these numbers correspond to the RTSP, HTTP, and MMS ports. For encoders, these correspond to the port numbers in the broadcasting setup pages.

Predefined Access Rules

Helix Server predefines two access rules:

- `Allow localhost access`—This rule permits access to Helix Server from an application running on the same computer. You should not edit this rule. This rule should always come first in the access control list.
- `Allow all other connections`—This rule allows all clients to make any request on any port. Access is denied, though, if the content is secured, and the client does not supply a valid user name and password.

By default, the second rule allows all clients to make requests on all ports. Hence, access control checking is off. To turn access control on, you need to delete or modify the second rule, and implement new rules.

Access to Helix Administrator

When you implement access control, you may inadvertently lock yourself out of Helix Administrator by denying all client access to the Admin port. Therefore, if you decide to set up access control, the first rule to define should allow access to the Admin Port. This rule needs to come directly after the predefined `Allow localhost access` rule. The section "Granting Access to Helix Administrator" on page 266 explains how to create this rule.

Access Rule Methods

To use the access control feature, you must make decisions about the types of rules you will create. Then, you can create as many rules as you need. There are two general methods that you can use to restrict access to Helix Server:

- specific address denial

  In this method, you deny access to a specific group of IP addresses and ports, and allow access to everyone else. This is the better policy if you

want to block a small number of clients, while allowing most clients to make requests.

- specific address permission

  This method is the opposite of the preceding. Here, you allow access to a specific group of IP addresses and ports, and deny access to everyone else. This is the better policy if you want to block a large number of clients, allowing only a small number of clients to make requests.

When you create a rule, you select a specific client IP address. Optionally, you can extend the addressing by choosing a bit mask, as described in Appendix B. You then select the ports for which that set of clients is allowed or denied access. You may need only one access rule. Or, you may want to set up several.

Rule Order

When you create multiple access rules, you set a rule order using the up and down arrow buttons on the rule list. Helix Server carries out rules in order from first to last. When a client connects, Helix Server evaluates the connection starting with the first rule on the list. As soon as it finds a rule that matches the player's address, it allows or denies access according to that rule.

> **Tip:** When implementing an access control policy, make the rules at the top of the list more strict. Reserve lower positions for the more lenient rules.

IPv4 and IPv6 Access Rules

Helix Server supports access rule checking for both IPv4 and IPv6 addresses. If Helix Server runs on a machine that has both IPv4 and IPv6 addressing, you may need to create rules for both IPv4 addresses and IPv6 addresses:

- Client connections using an IPv4 address are checked against IPv4-based rules.
- Client connections using an IPv6 address mapped to an IPv4 address are checked against IPv4-based rules.
- Client connections using an IPv6 address are checked against IPv6-based rules.

Granting Access to Helix Administrator

If you decide to implement access control rules, the first step is to set up a rule that enables you to connect to Helix Administrator, regardless of the restrictions you create in other rules.

➤ To grant access to Helix Administrator:

1. If you do not know the Admin port number, click **Server Setup>Ports**. Or, click the **View** link at the bottom of the Access Control page. Note the value of the **Admin Port** field.
2. Click **Security>Access Control.**
3. Click the "+" icon in the **Access Rules** section.
4. In the **Edit Rule Description** box, enter a rule description such as `AccessToAdmin`.
5. In the **Access Type** pull-down list, select `Allow`.
6. In the **Client Hostname or IP Address/Netmask** box, you have two choices:
   a. Type `Any`. Although this appears to allow everyone access to Helix Administrator, administrator log-in is guarded by the randomly-generated Admin port number, as well as user name and password validation, as described in "Administrator Authentication" on page 270.
   b. For additional security, specify the IP address for permitted users. You can indicate a range of allowable addresses by adding a net mask after the address, separating the two entries with a forward slash (/).

      For More Information: For information on using a bit mask, see Appendix B.
7. In the **Server Hostname or IP Address** box, type `Any`.
8. In the **Ports** box, enter the Admin port number.
9. In the **Access Rules** area, click the up arrow to make the administrator access rule the second rule on the list, following the first predefined rule (`Allow localhost access`).
10. Click **Apply**.
11. Restart Helix Server.

Creating General Access Rules

Follow the steps in this section to allow or deny access to specific client IP addresses or address ranges.

> **Warning!** Be sure first to follow the steps in "Granting Access to Helix Administrator" on page 266, or you may not be able to access Helix Administrator after you implement your access rules.

➤ To limit client access requests by IP address:

1. Review the ports in use for RTSP (usually 554) and MMS (usually 1755). You'll need these numbers for Step 7. You can determine the port values by clicking **Server Setup>Ports**. Or, click the **View** link at the bottom of the Access Control page.
2. Click **Security>Access Control**.
3. Click the "+" icon and enter a short description for the new access rule in the **Edit Rule Description** box. This description is for your reference only.
4. From the **Access Type** list, indicate whether permission is to be granted or refused by selecting `Allow` or `Deny`.
5. In the **Client Hostname or IP Address/Netmask** box, type the IPv4 or IPv6 address of the client machine. To indicate a range of client IP addresses, add a net mask after the address, separating the two entries with a forward slash (/). To refer to all clients regardless of IP address, enter `Any`.

   > **For More Information:** For information on using a net mask, see Appendix B. See "IPv4 and IPv6 Access Rules" on page 265 for information about rule-checking when an IPv6 address has a mapped IPv4 address.

6. In the **Server Hostname or IP Address** box, type the IPv4 address, IPv6 address, or host name of Helix Server. You can type a specific address, or use the word `Any` to refer to any IP address Helix Server uses to listen for incoming requests.

   > **Note:** If you type a specific IP address or host name rather than `Any`, ensure that the address is in the IP binding list. See "Binding to an IP Address" on page 66 for more information.

7. List the Helix Server port numbers to which you want to restrict access. In the **Ports** box, type the port numbers you noted in Step 1, separating entries with commas. For example, type:

   ```
   1090, 554
   ```

8. In the **Access Rules** area, click the up arrow or down arrow to move the rule to its appropriate position on the list. General access rules should always come after the `Allow localhost access` rule, and the rule you created for allowing access to Helix Administrator. For more information, see "Rule Order" on page 265.

9. Click **Apply**.

10. Restart Helix Server.

CHAPTER 13

AUTHENTICATION

Helix Server authentication provides a way to control what or who can access your Helix Server, whether an encoder sending a broadcast stream, a colleague perusing Helix Administrator, or a user viewing paid content. This chapter explains how to set up user name and password authentication, as well as validation through media player IDs.

> **Tip:** Chapter 12 describes how to limit access to media based on media players' IP addresses. The section "Controlling Connections" on page 73 explains how to limit connections according to outgoing bandwidth use, total number of players, and other general criteria.

Understanding Authentication

Authentication verifies the identity of the users or software programs that make requests of Helix Server. It usually takes the form of user name and password validation, though this is not necessary in all cases. The following sections describe the major authentication features and components.

Types of Authentication

There are several types access requests that you can authenticate, such as viewers requesting media, or Helix Server users logging into Helix Administrator.

Media Viewer Validation

The most common use of authentication is to validate viewer access to on-demand clips or broadcasts. Helix Server can require a standard user name and password combination that the viewer enters when requesting secure content. Or, you can have viewers register their players' globally unique

identifiers (GUIDs). Helix Server then validates access requests automatically, without asking viewers for user names and passwords.

The following table lists supported media players and the types of authentication that you can use with them. `Basic`, `RealSystem 5.0`, and `Windows NT LAN Manager` are forms of user name and password validation, as described in "Authentication Protocols" on page 285.

Media Players and Supported Authentication Types

Media Player	Basic	RealSystem 5.0	Windows NT LAN Manager	player GUID
RealPlayer 3 and earlier	no	no	no	no
RealPlayer 4	no	no	no	**yes**
RealPlayer 5 and later	**yes**	**yes**	**yes**	**yes**
Windows Media Player	no	no	no	no
QuickTime Player	**yes***	no	no	no
Any other RTP-based player	no	no	no	no

* – Basic authentication functions for QuickTime Player running on Windows only.

For More Information: See the section "Setting Up Basic Media Authentication" on page 275 for the basics of user name and password authentication. "Validating Media Player IDs" on page 297 explains validation through player GUIDs.

Administrator Authentication

Accessing Helix Administrator requires a valid user name and password. As explained in "Starting Helix Administrator" on page 52, the URL used to access Helix Administrator contains the `/admin/` mount point, which automatically authenticates the login. The installation process creates the initial user name and password pair, but you can add additional user names and passwords to the `SecureAdmin` realm, as described in "Managing Users and Passwords" on page 278.

Encoder Validation

User name and password authentication for live or simulated live streams sent to Helix Server is generally required for these encoders:

- RealProducer running in account-based mode. See "Setting Up Account-Based Broadcasting" on page 140 for information about this broadcast mode.
- RealProducer G2 through 8.5. The section "Encoding with an Older Version of RealProducer" on page 142
- SLTA running in basic mode. For more information, see "Basic Mode Configuration" on page 217.
- Windows Media push broadcasting. For more information, see "Broadcasting Windows Media" on page 143.

Although these encoders can use the main user name and password you use to log into Helix Administrator, RealNetworks recommends that you add additional user name and password pairs to the appropriate realm for each live broadcast encoder. See "Managing Users and Passwords" on page 278 for more information.

Password Validation Not Part of the Authentication System

Communication between the following components and Helix Server typically require password validation:

- RealProducer running in push or pull mode.
- SLTA running in advanced mode.
- Transmitters and receivers in splitting arrangements.
- Third-party media encoders.

This validation is *not* performed by the authentication system, however. Generally, the required passwords are defined on the encoder in its interface or configuration file. On Helix Server, you use Helix Administrator to set up the passwords, which are stored in the configuration file.

Content Caching Subscriber Authentication

Content caching subscriber authentication is initiated by the content caching file system when making a request to the publisher server from the subscriber server. Like administrator and encoder authentication, the user name and password you enter during setup is also the default user name and password required for content caching subscribers. You can add additional user name and password pairs to the `SecureCDist` realm, as described in "Managing Users and Passwords" on page 278.

For More Information: The subscriber server sends the authentication user name and password with its request for content. For more information, see "Setting up Content Caching Subscribers" on page 113.

Authentication Components

As you set up authentication, you work with several components. Databases store privileges. Realms validate user names and passwords. Commerce rules determine which on-demand clips and live broadcasts are secure. And permissions grant access to content on a user-by-user basis.

Databases

On each authentication request, Helix Server verifies the user's password and permissions in a database. Helix Server installs a number of flat file databases, as described in "Using Databases" on page 282. It uses different databases for different types of authentication. One database holds permissions for media players, for example, while another verifies the identity of encoders that deliver live broadcast streams.

To implement authentication on a limited scale (a few hundred users, for example), use the predefined flat file databases. This requires no additional database configuration. For large-scale implementations of authentication, however, you can tie Helix Server's authentication system to an ODBC-compliant or mSQL relational database. On Windows NT systems, you can also tie authentication into an existing LAN manager database.

For More Information: For more about databases, see "Using Databases" on page 282. Appendix C explains the database structure, which you'll need to know to use a relational database for authentication. See also "Windows NT LAN Manager" on page 286.

Realms

An authentication realm indicates the database that stores a user's name and password, and specifies the authentication protocol used to validate the user's identity. An authentication protocol, which is not related to streaming protocols such as RTSP, determines how passwords are encrypted in the database. You can use a basic encryption protocol, or a more secure protocol that works with RealNetworks media players only.

Tip: Helix Server predefines several realms. Depending on your authentication needs, you may not need to change or add realms. For more on realms, see "Setting Up Realms" on page 284.

Commerce Rules

Commerce rules determine which on-demand clips or live broadcasts require authentication. Helix Server predefines a set of commerce rules, each rule creating a *protected path* that leads to secure content. One rule protects all on-demand clips residing in the default security directory, for example. You can set up additional commerce rules as needed. For example, you'll need to create a new commerce rule, or modify an existing one, to secure clips that reside on a network drive.

Note: Commerce rules work only for media access, and do not apply to other forms of authentication, such as encoder validation. For more information, see "Defining Commerce Rules" on page 287.

Permissions

Permissions attach to commerce rules to govern which users can view which clips or broadcasts. Using permissions, you can grant different users access to different secured clips. You can also grant viewing access that expires at a certain date and time, for example, or that is limited to a total amount of time. Although permissions are enforced by default, you can turn them off to give all authenticated users unlimited access to the content protected by a commerce rule. When you turn off permissions for on-demand content, for instance, all authenticated users have unlimited access to all secured, on-demand clips.

For More Information: For specifics about the permissions that you can grant, see "Handling User Permissions" on page 291.

Authentication Used with Other Features

Authentication works with all other Helix Server features. There are few special considerations for each feature, however.

Authentication Used with Other Features

Other Feature	Notes
On-Demand Streaming	All on-demand files stored in the Secure directory (or in any subdirectories) are authenticated automatically, once the authentication feature has been set up.
Live Unicasting	Once the authentication feature has been set up, live broadcasts are authenticated automatically if they include /secure/ as part of the path when you encode the events.
Archiving	Archived files are on-demand files that can be authenticated if they are moved to the correct location. They must be placed in the Secure directory or in a subdirectory of Secure, or the archiving feature must be configured to use the Secure directory.
SLTA	Just like any other live event, broadcasts created by SLTA can be authenticated, as long as you include /secure/ in the broadcast path.
Splitting	If you are sending a stream to a Helix Server that is acting as a receiver, you must put copies of all the databases that store authentication information on the receiver. This distributes the authentication load.
Multicasting	In back-channel multicasts, the user or client is authenticated through the initial control channel connection. Be sure the multicast (/) path is on the list of commerce rules. Authentication does not function with scalable multicasts.
Helix Proxy	Helix Proxy makes requests on behalf of clients, and caches the streams it receives. Although Helix Proxy stores the streamed data, it requires a control channel between the requesting client and Helix Server. Helix Server uses the control channel to request and receive authentication information.
Firewalls	Authentication is performed over the two-way control channel. As long as the client can establish a connection through the firewall to Helix Server, all material can be authenticated for clients behind firewalls.

(Table Page 1 of 2)

Authentication Used with Other Features (continued)

Other Feature	Notes
Access Control	Access control verification, which checks the client's IP address against a list of allowed addresses, occurs before authentication. So if a client's IP address is blocked, authentication will not take place. If users who should be able to view secure material receive error messages, check the list of access rules to see if their client addresses are disallowed.
ISP Hosting	Authentication of content cannot be applied to the files of ISP-hosted customers. This material is always available. Depending on the access needs, you may be able to apply access control rules so that customers can allow or deny certain users' access to content.
Monitoring	You can monitor which secure presentations are in use by viewing the paths of the files in Server Monitor. Those that contain the `/secure/` mount point are authenticated.
Reporting	Efforts to authenticate users are not included in the basic access log; records are created for successful serves. You can identify authenticated material in the basic access log by the `GET` statement. Secure material always contains the `/secure/` mount point in the path. In addition, connection attempts for authenticated material are stored in the `accesslog.txt` file in the `Logs` directory of appropriate data storage directory (if you are using the text file method), or in the `Access_log` table (if you are using the database method).

(Table Page 2 of 2)

Setting Up Basic Media Authentication

The following sections explain the basics of how to secure on-demand clips and broadcasts through user name and password validation. They explain where to place clips, how to secure a broadcast originating from an encoder, and how to write URLs to the secure content. When you use the Helix Server default settings, there are two setup steps you need to perform through Helix Administrator to implement user name and password checking:

1. Add user name and password combinations according to the instructions in "Managing Users and Passwords" on page 278.
2. Define user permissions (or turn them off), as described in "Handling User Permissions" on page 291.

In some cases, you may need to change certain default settings for databases and realms, depending on your authentication needs:

- If you are not using one of the predefined flat-file databases for storing passwords, you need to set up your database first, as described in "Using Databases" on page 282.
- When you create a new database, you must associate a new or existing realm with it, as described in "Setting Up Realms" on page 284.
- To validate access attempts from QuickTime Player, set the `Basic` authentication protocol in your authentication realms. For instructions on doing this, see "Creating or Modifying a Realm" on page 287.
- If you want to validate media player requests based on each player's ID, rather than on the users' names and passwords, refer to the instructions in "Validating Media Player IDs" on page 297.

Securing On-Demand Content

To require user name and password validation, you place clips in Helix Server's `Secure` directory instead of in its `Content` directory. In a default installation on Windows, the `Secure` directory is here:

```
C:\Program Files\Real\Helix Server\Secure
```

Installation paths vary on UNIX, but the `Secure` directory is under the main installation directory, as in this example:

```
/usr/local/Real/HelixServer/Secure
```

Helix Server identifies requests for secure, on-demand clips through a `/secure/` mount point that precedes the clip name in the request URL. This mount point is created during installation, and automatically requires authentication for all content in the `Secure` directory. The following is an example of a Web page URL to a secure clip:

```
http://helixserver.example.com/ramgen/secure/video1.rm
```

> **For More Information:** See "Writing Links to Content" on page 85 for background on link formats, mount points, and URLs used in Ram files.

Adding Subdirectories to Implement Permissions

You may want to create subdirectories within the `Secure` directory, then set up permissions as described in "Handling User Permissions" on page 291 to

define exactly which viewers can access which clips in which subdirectories. In this case, the request URL includes the subdirectory path, which can be any number of levels deep, after the /secure/ mount point:

```
http://helixserver.example.com/ramgen/secure/dailyvideo/video1.rm
```

Placing Secure Clips in Other Directories

You can store secure, on-demand clips in directories other than the default Secure directory. This allows you to store secured clips on other drives or network machines. To do this, you define a new, secure mount point used in place of the default /secure/ mount point.

➤ To create a new security mount point for on-demand clips:

1. Create a directory on your Helix Server machine or your network to store your secure clips. This directory must not be a subdirectory of the existing Content or Secure directory.
2. Add a new mount point, such as /secure2/, that points to the new directory. For instructions on doing this, see "Adding a Mount Point for On-Demand Clips" on page 95.
3. Follow the instructions in "Adding or Modifying Commerce Rules" on page 289 to create a commerce rule that provides access to this new protected path.
4. Set up permissions for individual users to view content in the new protected path, as described in "Handling User Permissions" on page 291.

As shown in the following example, a URL to a clip in the new secure directory uses the mount point you define:

```
http://helixserver.example.com/ramgen/secure2/video1.rm
```

When Helix Server receives a request to this protected path, it initiates authentication to verify that the user or player exists in the proper database according to the commerce rule that protects the /secure2/ path.

Securing Broadcasts

To secure a live or simulated live broadcast, you include the /secure/ mount point after the broadcast mount point in the request URL:

```
http://helixserver.example.com/ramgen/broadcast/secure/live.rm
```

Live or simulated live broadcasts do not correspond to actual files on Helix Server. When a broadcast mount point such as /broadcast/, /encoder/, or /live/

precedes the /secure/ mount point, Helix Server performs authentication, but does not search for a file in its Secure directory. Instead, it delivers the live stream associated with the broadcast mount point.

Specifying the Secure Path on the Encoder

The encoder that delivers the live or simulated live stream must include the /secure/ virtual path along with the broadcast file name to indicate that this stream requires authentication. In RealProducer, for example, you specify the live stream as secure/live.rm instead of just live.rm.

> **For More Information:** Predefined commerce rules govern access to broadcasts from different encoders. For more information, see "Default Commerce Rules" on page 288.

Managing Users and Passwords

The following sections explain how to manage the user names and passwords of authorized users for any type of authentication, whether individuals requesting secured clips, encoders connecting with live streams, or additional users of Helix Administrator.

> **Note:** If you are using Windows to manage the list of users, passwords, and groups, use those tools instead of the instructions below. To use Windows passwords, you need to set the NTLM authentication protocol in your selected realm, as described in "Creating or Modifying a Realm" on page 287.

Adding a User

Follow the procedure below to add a user and password to an authentication realm. To use a database other than one that is predefined, you must create that database and associate it with the proper realm before adding users. Refer to "Using Databases" on page 282 and "Setting Up Realms" on page 284 for more information.

> **Note:** The Helix Administrator interface does not provide a way to add multiple user names and passwords at one time.

➤ To add a user name and password:

1. Click **Security>Authentication**.

2. In the **Authentication Realms** list, select the realm to which you want to add a user. The following realms are predefined:

`SecureAdmin`	Helix Administrator users
`SecureCDist`	content caching subscribers
`SecureContent`	media player users
`SecureEncoder`	RealProducer G2 through 8.5 delivering live broadcast streams
`SecureRBSEncoder`	RealProducer delivering live broadcast streams, and SLTA delivering simulated live streams in basic mode
`SecureWMEncoder`	Windows Media Encoder delivering live broadcast streams

For More Information: Realms are described in "Setting Up Realms" on page 284.

3. Click **Add a User to Realm**.
4. In the pop-up window, define the user's name in the **Name** box. User names are case-sensitive. You can use separate words if, for example, you want to use full names of users.
5. In the **Password** box, supply the user's password. Passwords are case-sensitive. RealNetworks recommends following good password practices:
   - Avoid common words that are easy to guess.
   - Do not use a word associated with the user, such as a first name.
   - Do not use the same password for multiple users.
   - For highest security, use a random combination of letters and numbers in different cases.

     Tip: Keep track of the passwords you assign. Helix Administrator allows you to change passwords, but not to look them up.
6. In the **Confirm Password** box, type the password again.
7. Click **OK**.

Deleting a User

The following procedure explains how to delete a user from a database. Helix Administrator does not have a bulk delete feature.

➤ To remove a user:

1. Click **Security>Authentication**.
2. In the **Authentication Realms** list, select the name of the realm in which you want to delete a user. The predefined realms are described in "Setting Up Realms" on page 284.
3. Click **Remove a User from Realm**.
4. In the new window that appears, enter the user's name in the **Name** box.
5. Click **OK**.

Browsing All User Names

The browsing feature lists all user names defined for an authentication realm.

➤ To browse all users:

1. Click **Security>Authentication**.
2. In the **Authentication Realms** list, select the realm you want to browse. The default realms are described in "Setting Up Realms" on page 284.
3. Click **Browse Users in Realm**. The pop-up window lists all user names defined for that realm.

Changing a Password

The following procedure explains how to change the password for an existing user. The Helix Administrator interface does not allow you to look up existing passwords.

➤ To change a password:

1. Click **Security>Authentication**.
2. In the **Authentication Realms** list, select the name of the realm that contains the user. The predefined realms are described in "Setting Up Realms" on page 284.
3. Click **Change User Password**.
4. In the new window that appears, enter the user's name in the **Name** box.
5. In the **Password** box, specify the user's new password.
6. In the **Confirm Password** box, type the password again.

7. Click **OK**.

Using the Password Tool

When it uses the RealSystem 5.0 authentication protocol, Helix Server encrypts passwords, so you cannot look up the passwords directly. However, you can add or change passwords in a flat file or relational database by using a command-line utility. You can even create a password interface by integrating this utility with your own CGI scripts and Web pages.

For More Information: See "Authentication Protocols" on page 285.

➤ To use the password tool:

1. Open a command prompt and navigate to the `Bin` directory under Helix Server's main installation directory.
2. Enter the following command:

   `makepass` *`username realm`*

   using the following values:

`username`	The user name exactly as it is entered, or will be entered, in the authentication database.
`realm`	The value of the **Realm** variable specified in the relevant list. For RealProducer, this is given in the **Authentication** field under **Broadcasting>RealNetworks Encoding**. In the configuration file, it is given by the value of the `Realm` variable in the `G2_Encoders` list. For Helix Administrator users, use the value of the `Realm` variable in the `RealAdministrator_Files` list within the `FSMount` list in the configuration file.

3. A password prompt appears, followed by a prompt to type the password again. The resulting encrypted password is displayed on the screen.

   Helix Server encrypts passwords with the MD5 hashing algorithm. It uses the form `MD5("`*`username:realm:new_password`*`")`. On BSD systems and some other UNIX systems, you can generate these passwords with the following command:

   `echo -n "`*`username:realm:new_password`*`" | md5`

4. Add the resulting encrypted password into the appropriate field of the database:
   - For text files, place it in the password field of the `Users` directory. See "Users Directory" on page 405.
   - For databases, place it in the password field of the `Users` table. See "Users Table" on page 409.

Using Databases

In its default configuration, Helix Server uses a set of flat files to store user names, passwords, and permissions. For large-scale implementations of authentication, RealNetworks recommends that you tie Helix Server into an ODBC-compliant or mSQL database that stores this information. The following table lists the flat file databases automatically installed with Helix Server.

Default Databases

Name	User Names and Passwords	Purpose
`Admin_Basic`	Helix Administrator.	Validate access to Helix Administrator.
`CDist_Basic`	registered content caching subscribers	Verify subscribers requesting content from a publisher.
`Content_RN5`	content users	Validate users requesting secured on-demand or live content.
`Encoder_Basic`	RealProducer	Verify content creators delivering live or simulated live broadcasts.
`Encoder_Digest`	Windows Media Encoder	Verify content creators delivering live or simulated live broadcasts.
`Encoder_RN5`	RealProducer G2 through 8.5	Verify content creators delivering live or simulated live broadcasts.
`PlayerContent`	player GUIDs and user names for viewers accessing content	Validate unique IDs for players accessing content.

Supported Database Types

Helix Server provides interfaces to several types of databases. Appendix C contains details about the database structure, which you'll need to know to

integrate Helix Server's authentication system with a relational database, for example.

Flat File Database

The default databases used for authentication are flat text files, which work well for storing relatively small amounts of data, such as a few hundred user names and passwords. You may want to use them to learn the authentication data structure before linking Helix Server to a more robust relational database. If you choose to use the default flat files exclusively, you do not need to perform any additional configuration.

ODBC and mSQL

Helix Server includes templates for common relational databases, covering mSQL and ODBC-compliant databases. To use an ODBC or mSQL database, you must configure your database to comply with the appropriate table structure described in Appendix C.

RN5 DB Wrapper

If you used authentication features with RealSystem Server version 5, or if you have a data store plug-in created by a third-party company, you can use that plug-in with Helix Server Version 11.1.

Adding a Database

Follow the procedure below to add a new database that stores user names and passwords. If you are using the default flat file databases, it is typically not necessary to add a new database.

➤ To add a database:

1. Click **Security>Databases**.
2. Click the "+" icon, and type a description for the new database in the **Edit Database Name** box.
3. From the **Database Type** list, select the data storage method you want to use. Database types are described in "Supported Database Types" on page 282.
4. Depending on the database type method you chose, additional information is required.

a. **Flat File** needs only the path to the main text file directory. For example, the `enc_r_db` directory under the main Helix Server directory. For more information, see "Understanding Authentication Data" on page 403.

b. **mSQL** has two required names, and three optional items:

   Host Name—DNS name, IPv4 address, or IPv6 address of the computer where the database is stored.

   Database Name—Name of the database.

   Table Name Prefix—Prefix used to make field names unique, when used with an existing database.

   User Name—Name required by the database application.

   Password—Password required by the database application. Re-enter your password in the **Confirm Password** box to ensure that you typed it correctly.

c. **ODBC** uses the same information as mSQL, but ODBC does not ask for a host name. Refer to "Setting Up Other Types of Data Storage" on page 411 for further instructions.

d. For **RN5 DB Wrapper**, the following items are needed:

   Database Name—Name or location of the data storage plug-in. Consult your plug-in documentation for information about what should go here.

   Plugin Path—Location of the plug-in.

   User Name—Name required by the database application.

   Password—Password required by the database application. Re-enter your password in the **Confirm DB Login Password** box to ensure that you typed it correctly.

5. Click **Apply**.

Setting Up Realms

A realm connects users to databases. When you define passwords, you add them to a realm. The realm, in turn, specifies the encryption protocol, and indicates the database where information is stored. If you are using the default authentication databases, as described in "Using Databases" on page 282, you can use the default realms, too, changing the authentication protocols if

necessary. If you set up new databases, you need to create new realms, or point the existing realms to your new databases.

Predefined Authentication Realms

Realm	Authenticates	Realm ID	Protocol	Database
SecureAdmin	Helix Administrator	*servername*.AdminRealm	Basic	Admin_Basic
SecureCDist	content caching subscribers	*servername*.CDistRealm	Basic	CDist_Basic
SecureContent	content users	*servername*.ContentRealm	RealSystem 5.0	Content_RN5
SecureEncoder	live streams from RealProducer G2 through 8.5	*servername*.EncoderRealm	RealSystem 5.0	Encoder_RN5
SecureRBSEncoder	live streams from RealProducer	*servername*.RBSEncoder Realm	Basic	Encoder_Basic
SecureWMEncoder	live steams from Windows Media Encoder	*servername*.WMEncoder Realm	Digest	Encoder_Digest

Authentication Protocols

Authentication protocols determine the password encryption and storage method used by Helix Server. The server supports three protocols. Each realm uses just one protocol.

Basic

The Basic protocol sends the user's name and password over the public Internet in a simple manner, encoding them with the Base64 algorithm. Helix Server decodes and verifies the password. Information can be stored in a flat file or a relational database. This protocol works with RealNetworks media players, as well as the QuickTime Player.

Digest

Digest authentication is form of encrypted authentication commonly used between components communicating across the Internet. Windows Media Encoder uses this protocol to send authentication information to Helix

Server. The authentication information can be stored in a flat file or a relational database.

RealSystem 5.0

Also called *RN5*, this is a RealNetworks encryption protocol that works with RealPlayer 5 and later. It is more sophisticated and secure than the Basic protocol. Use it if your material will be served exclusively to users who have RealPlayer 5 or later. Information can be stored in a flat file or a relational database.

> **Tip:** For authentication of QuickTime Player or RealNetworks players earlier than version 5, use the Basic protocol.

Windows NT LAN Manager

The NTLM method is suitable for a Windows-based intranet. It enables Helix Server to use the existing Windows NT database of user groups. It also allows access control of content through NTFS file permissions. This method requires that Helix Server itself be installed on the Windows NT machine. When using NTLM authentication, be aware of the following:

- NTLM authentication works only with RealPlayer 5 and later.
- All users' accounts must exist on the local NT computer. NTLM authentication will not work with accounts on other servers within the domain, but it will authenticate against accounts on the primary domain controller.
- You add all user accounts through the Windows NT User Manager. Do not use the instructions in "Managing Users and Passwords" on page 278.
- The built-in guest account is not available for use in authentication.
- When you select NTLM authentication for Helix Administrator access, all groups are authenticated if you do not specify a user group.
- You cannot evaluate permissions on commerce rules when you use NTLM authentication.
- Blank passwords are not supported.
- You cannot use NTLM authentication to validate connection attempts by RealProducer. For encoder connections, define user names and passwords under a realm that uses `Basic` authentication.

Creating or Modifying a Realm

The following procedure explains how to create a new realm or modify an existing one. It is generally not necessary to do this unless you have created a new database. Use a one-to-one correspondence between realms and databases. Do not create two realms, for instance, that use the same database.

➤ To create or modify a realm:

1. Click **Security>Authentication**.
2. To create a realm, click the "+" icon and enter a name for this realm in the **Edit Realm Description** box. To modify an existing realm, select it in the **Authentication Realms** box.
3. In the **Realm ID** box, type a name that will be used in other areas of Helix Administrator. The realm name may also appear to users as part of the name and password prompt. The default realms conform to the following format:

   servername.Realm_Id

   > **Warning!** You do not have to use the default convention, but you must include a period (.) in the realm ID or the realm will not work properly.

4. In the **Authentication Protocol** list, select the authentication method you want to use for this realm, as described in "Authentication Protocols" on page 285.
   a. If you choose `Basic` or `RealSystem 5.0`, select the database in the **Database** box.
   b. If you choose `Windows NT LAN Manager`, Helix Server uses the NT list of names instead of a database. Type the appropriate provider in the **Provider** list, such as `NTLM`. Type the Group name in the **Group** box.
5. Click **Apply**.

Defining Commerce Rules

Commerce rules enforce authentication by defining protected paths. For example, the default commerce rule for on-demand clips defines the protected path `/secure/`, which corresponds to the `Secure` directory. When that path appears in a URL request, Helix Server authenticates the request, allowing

anyone with a valid user name and password to view clips in that directory. You can modify the default commerce rules, and create new commerce rules, to carry out tasks such as the following:

- Secure clips in locations other than the `Secure` directory. You can create a new commerce rule to secure the `Archive` directory, for example. As the section "Placing Secure Clips in Other Directories" on page 277 explains, you can also store clips on different network drives or machines, creating a new commerce rule to secure those assets.
- Secure all live broadcasts. Helix Server automatically secures broadcasts that use protected paths such as `/broadcast/secure/` or `/encoder/secure/`. However, broadcasts that use just the broadcast mount points, such as `/broadcast/` or `/encoder/`, are not secured by default.

  Tip: If you changed the protected paths to `/broadcast/` and `/encoder/`, you would secure all broadcasts automatically.

- Allow multiple viewers to use the same user name and password to access an on-demand clip or broadcast. This lets you grant quick access to secured content to everyone within a division or a company, for example.
- Validate player IDs instead of user name and password combinations. For more on this, see "Validating Media Player IDs" on page 297.

  Note: Commerce rules apply only to viewers attempting to access secure clips or broadcasts. They do not apply to encoders or Helix Administrator users, for example.

Default Commerce Rules

Helix Server defines a number of commerce rules that provide general access to secured content. For example, these rules allow access to on-demand clips in the `/secure/` path, or to broadcasts in the `/broadcast/secure/` path.

SecureG2LiveContent

The commerce rule `SecureG2LiveContent` grants access to live broadcasts encoded with RealProducer G2 through 8.5. The rule evaluates the protected path `/encoder/secure/`. Users requesting live broadcasts through this path are associated with the `SecureContent` realm by default, and must have a user name and password defined in this realm to gain access to the secured broadcast.

SecureLiveContent

SecureLiveContent grants access to live broadcasts encoded with RealProducer , as well as simulated live broadcasts delivered through SLTA. The rule evaluates the /broadcast/secure/ protected path. Users requesting live broadcast URLs that contain this path are associated with the SecureContent realm by default, and must have a user name and password defined in this realm to gain access to the secured broadcast.

SecurePlayerContent

This commerce rule, which is used for validating media player IDs, grants access to the player directory, a subdirectory of the /secure/ mount point. User and client information must exist in the player database before viewers can access the content. The section "Validating Media Player IDs" on page 297 explains player ID validation.

SecurePreG2LiveContent

You use the SecurePreG2LiveContent commerce rule for content encoded with versions of RealProducer earlier than RealProducer G2. The rule evaluates the protected path /live/secure/. Users requesting URLs containing this path are associated with the SecureContent realm and must have a user name and password defined in this realm to gain access to the secured content.

SecureUserContent

This commerce rule grants access to on-demand content in which the request URL contains the /secure/ mount point. Users requesting this content are associated with the SecureContent realm by default, and must have a user name and password defined in this realm to gain access to the secured content.

Adding or Modifying Commerce Rules

The following procedure explains how to add a commerce rule, or modify the overall attributes of existing commerce rules. The section "Handling User Permissions" on page 291 explains how to define permissions for a commerce rule.

➤ To create or modify a commerce rule:

1. Click **Security>Commerce**.

2. To create a new rule, click the "+" icon and edit the name that appears in the **Edit Rule Name** box. To modify a rule, select its name in the **Commerce Rules** box.

3. In **Protected Path**, enter the path that appears in the URL to invoke the commerce rule. For example, if you set up a `/secure2/` mount point for secured content, enter `/secure2` (no closing slash) in the box.

   > **For More Information:** For more on creating additional secure mount points, see "Placing Secure Clips in Other Directories" on page 277.

4. In **User Permissions Database**, select the database that will store the user permissions. The default database for on-demand content is `Content_RN5`. The table "Default Databases" on page 282 lists the other default databases.

   > **Note:** You can turn off individual user permissions for the commerce rule by selecting `Do Not Evaluate User Permissions`. In this case, all authenticated users have unlimited access to content protected by the commerce rule.

5. In the **Credential Type** list, select `Use User Authentication` to require user name and password validation. The other choice, `Use Player Validation`, validates access attempts according to player IDs, which is described in the section "Validating Media Player IDs" on page 297.

6. If you chose user name and password validation, you next select the realm in the **Realm** pull-down list. The default realm for on-demand content is `SecureContent`. The table "Predefined Authentication Realms" on page 285 lists the other default realms. User names and passwords must be defined in the selected realm, as described in "Managing Users and Passwords" on page 278.

   > **Warning!** You must select the realm associated with the database you chose in the **User Permissions Database** field.

7. If you selected user name and password validation, you can set **Allow Duplicate User IDs** to `Yes`. This allows multiple people to use a single user account simultaneously. If you want to distribute a video to all employees in a division, for example, you can allow duplicate IDs and supply all persons in that division with the same user name and password.

8. Click **Apply**.

Note: The **Player GUID Databases** section is used only with player ID validation. The section "Modifying the GUID Database" on page 300 explains how to use that part of the commerce rules page.

Handling User Permissions

Permissions are associated with commerce rules. They determine which authenticated viewers can see which content in a protected path. By setting up permissions, you can grant access to clips on a person-by-person and file-by-file basis, for instance. You can also create permissions that allow secured access to clips or broadcasts only at certain times, or for certain durations. Note the following about permissions:

- Permission checking is enabled by default. This means that you must define permissions for individual users to give them access to secured content. Even if users have valid user names and passwords, they will not be able to view *any* content if they have no permissions defined.
- You can turn off permission checking for a commerce rule by choosing `Do Not Evaluate User Permissions` in the **User Permissions Database** pull-down list. In this case, all users with valid user names and passwords have unlimited access to all content protected by the commerce rule. For instructions on doing this, see "Adding or Modifying Commerce Rules" on page 289.
- Helix Server does not evaluate permissions on commerce rules when you use the `Windows NT LAN Manager` authentication protocol. If you are using that protocol, there is no need to define permissions.

Permission Types

When you associate permissions with a user, you can set up two types of access. You can allow access to entire directories. You can also set up permissions file-by-file, or use a combination of these two methods.

Directory-Level Access

Directory-level access specifies that a user can view all of the content in a certain directory, as well as in its subdirectories, which inherit the permission. For example, you might give viewer A directory-level access to this path:

```
/secure/confidential/
```

The viewer then has access to files in that path, as well as in *any* subdirectory below that path. Thus, viewer A can access a clip in this path:

```
/secure/confidential/secret/
```

But viewer A does not have access to a path such as this:

```
/secure/topsecret/
```

File-Level Access

File-level access grants a user access to a specific file or live broadcast stream. As described below, you can also specify the duration or manner of this access. A single user may have both directory and file access for a clip in the same directory. If the directory access has expired, but the file access is still valid, the viewer can request the valid file, but no others in the directory. Conversely, if the file access has expired but the directory access is still valid, the viewer can request all files in the directory except the expired file.

Permission Access Types

Whether you give a viewer directory-level or file-level access, you can assign an access type as described in the following table.

Permission Types

Access Type	Permission Granted
Event	Unlimited viewing of a given clip or a directory of clips. This is the default.
Calendar	Permission expires on a certain date and time. If the expiration date and time arrive while the viewer plays a clip or broadcast, transmission stops and an error message appears.
Duration	Viewer gets a finite amount of time to view presentations. All viewing time is deducted from this amount. When the duration time elapses, permission is revoked.
Credit	Time spent viewing presentations is noted in the Helix Server basic access log, and the administrator can use this information later for billing purposes. Access is granted per presentation or directory, and is unlimited. For information on the authentication access logs, see "Logs Directory" on page 406.

Note: With Duration or Credit permissions, Helix Server accurately reports the total time a user views a secure file. If the

end time minus the start time does not equal the total viewing time, for example, the discrepancy can be attributed to the user buffering, pausing, or seeking within the file. However, the total viewing time is the value debited or credited against the user's account, depending on the permissions used.

Permissions for SMIL Presentations

With a SMIL presentation, users are authenticated once regardless of how many files in the presentation are protected. If access to any clip in the presentation expires during the presentation, the presentation halts until more viewing time is allotted. For this reason, it is a good idea to keep permissions on all clips within a SMIL presentation the same. The following are the best methods of implementing authentication for SMIL files:

- Authenticate the SMIL file but not its contents. This is useful if you do not need a high level of security on all clips. In this case, it's a good idea to keep the view source feature set to its default, so that it does not show the full paths to the unsecured clips.

  For More Information: See "Displaying Source Information" on page 101.

- If you are using duration access, use it only for the longest clip in the presentation, which is typically an audio or video clip. Apply event access for the other files.

SMIL Files and Directory-Level Duration Access

If you define directory-level duration access for a SMIL presentation, giving identical permission to all files (including the SMIL file) will not work as you may expect. As each clip plays, Helix Server subtracts the viewing time from the directory allotment. If each clip is 10 minutes long, and there are three clips in the presentation, Helix Server subtracts 30 minutes from the total viewing time. This means that in setting up this type of access, the time allotted must be the sum of all the clips.

Tip: Keeping track of all the clips, their lengths, and the total directory access time can be tricky. An easier solution is to limit the access time only for the SMIL file.

Granting Permissions

You can create user permissions for commerce rules by following the procedure listed below. Note the following about defining permissions:

- You can define more than one permission for each user. You can give a single user permission to view several different clips, for example. You must define each permission separately, though, using the same user name each time.
- Each permission is associated with a single commerce rule. Typically, different commerce rules govern access to different directories or broadcasts. If you want to give a user permission to view one on-demand content directory and one broadcast, for example, you define one permission on the commerce rule governing on-demand clip access, and another permission on the commerce rule governing live broadcasts.
- The user name must first be defined in the database used by the commerce rule. For instructions about adding and browsing user names, see "Managing Users and Passwords" on page 278.

➤ To grant permissions to a user:

1. Click **Security>Commerce**.
2. Select the appropriate commerce rule in the **Commerce Rules** box.
3. Click the **Grant User Permission** link.
4. In the pop-up window, enter the user's name in the **User Name** box.
5. For **Path Type**, select `Directory` or `File` depending on the type of permission you want to add. For more information, see "Permission Types" on page 291.
6. The **Path** box indicates the directory or file that the viewer can access. It initially lists the protected path associated with the selected commerce rule. For example, if you're adding permissions to the default commerce rule for on-demand content (`SecureUserContent`), the path looks like this:

   ```
   secure/
   ```

   a. If you've chosen directory-level access, add to the protected path the subdirectory the viewer can access, as in the following:

      ```
      secure/confidential
      ```

      Note: The viewer can view clips in all subdirectories of the selected path.

b. If you've selected file-level access, enter the subdirectory, if necessary, and the clip file name, as shown here:

```
secure/confidential/report.rm
```

7. The **Access Type** pull-down list corresponds to permission types described in "Permission Access Types" on page 292.
   a. Set the menu to `Event` to allow unlimited access to the content.
   b. If you choose `Calendar`, set an expiration date in this format:

```
mm/dd/yyyy:hh:mm:ss
```

   c. For `Duration`, set in seconds the total amount of viewing time allowed.
   d. Choose `Credit` to log the viewer's total access time in the access logs.
8. Click **OK**.

Editing User Permissions

After you assign permissions to a user, you can edit those permissions to change the viewing expiration date, for example, or give the viewer more or less total viewing time on a directory or file.

➤ To modify user permissions:

1. Click **Security>Commerce**.
2. Select the appropriate commerce rule in the **Commerce Rules** box.
3. Click the **Edit User Permission** link.
4. In the pop-up window, enter the user's name in the **User Name** box. See "Browsing All User Names" on page 280 for information about listing user names.
5. For **Path Type**, you can select `Directory` or `File` to change the type of permission. For more information, see "Permission Types" on page 291.
6. The **Path** box indicates the directory or file that the viewer can access. For more information on this, see Step 6 on page 294 in the procedure for granting permissions.
7. If you had set an expiration date on the user permissions, you can enter a new date in the **New Expiration Date** box in this format:

```
mm/dd/yyyy:hh:mm:ss
```

   After doing so, click **Set New Date**.

8. If you set a duration on the user permissions, you can modify the viewing time allowed. Choose `Add` or `Subtract`, and enter the amount of time in seconds. Then click **Change Time**.
9. Click **Close Window**.

Revoking User Permissions

You can revoke some or all permissions granted to a user. Keep in mind, though, that each revocation applies only to the selected commerce rule. If a user has permissions defined for other commerce rules, you must revoke those permissions separately. Helix Administrator does not provide a means for revoking permissions for several users or several commerce rules at a time.

> **Tip:** See "Browsing All User Names" on page 280 for information about listing user names.

➤ To revoke a single permission for a user:

1. Click **Security>Commerce**.
2. Select the appropriate commerce rule in the **Commerce Rules** box.
3. Click the **Revoke User Permission** link.
4. In the pop-up window, enter the user's name in the **User Name** box.
5. For **Path Type**, select `Directory` or `File` according to the type of permission you want to revoke. For more information, see "Permission Types" on page 291.
6. The **Path** box indicates the directory or file that the viewer can access. For more information on this, see Step 6 on page 294 in the procedure for granting permissions.
7. Click **OK**.

➤ To revoke all permissions for a user:

1. Click **Security>Commerce**.
2. Select the appropriate commerce rule in the **Commerce Rules** box.
3. Click the **Revoke All Permissions** link.
4. In the pop-up window, enter the user's name in the **User Name** box.
5. Click **OK**.

Validating Media Player IDs

For authentication of on-demand content or broadcasts, you can validate media player IDs rather than user name and password combinations. For the privacy reasons described below, this method of authentication is better suited for an intranet, although you can use it with the public Internet. With player ID authentication, users first register their player IDs through a special URL. They can then view protected content without having to enter a user name and password. Media player ID validation works only with RealPlayer 4 and higher.

Player IDs and User Privacy

For player ID validation to work, players must be configured to send a globally unique ID (GUID) to Helix Server. For privacy protection, the standard RealPlayer that viewers download over the Internet is set by default *not* to send a GUID, in accordance with RealNetworks' Consumer Software Privacy Statement, which is published on the following Web page:

http://www.realnetworks.com/company/privacy/software.html

Because sending a GUID is solely at the discretion of each user, you must request users to change their default GUID setting (on RealPlayer, the command is **Tools>Preferences>Connection>Internet Settings**) to register their GUIDs and enable player ID authentication for material delivered over the Internet. Users who decline to do so will be denied access to secure content. However, you can reroute these users to a non-secure Web URL that explains their options, as described in "Setting Up Commerce Rules" on page 299.

Using Player ID Validation on an Intranet

In an enterprise setting, you can control whether or not RealPlayer transmits its GUID when you use RealNetworks' RealPlayer Enterprise Manager. You can turn GUID registration on by default for all users, making it easier to set up player ID validation on an intranet. For more on RealPlayer Enterprise Manager, visit the following Web page:

http://www.realnetworks.com/products/rpem/index.html

Choosing a Database and Realm for User Names

Helix Server installs a default database, `PlayerContent`, for use with player ID validation. You can use this database, or add your own relational database.

You need to associate a realm with the player ID database only to populate it with user names initially. After that, Helix Server refers directly to the database to validate player IDs, and does not refer to the realm again. Note the following:

- If you do not want to use the default `PlayerContent` database (you want to use a relational database instead, for instance), set up your database first, as described in "Using Databases" on page 282.
- If you do not plan to implement user name and password validation for any on-demand content, you can associate the default `SecureContent` realm with the player ID database you use.
- If you think that you may enable user name and password validation in addition to player ID validation, create a new realm for player ID validation, and associate it with your player ID database. This leaves the `SecureContent` realm available for user name and password validation.

The section "Creating or Modifying a Realm" on page 287 explains how to create a new realm or modify the `SecureContent` realm. For the realm you decide to use, select the `PlayerContent` database (or the relational database you're using) in the **Database** pull-down list. You can set the **Authentication Protocol** to `Basic` or `RealSystem 5.0`. Passwords are not used, so either protocol setting has the same effect.

Adding User Names to the Player ID Database

After you select your realm for player ID validation, add user names to that realm for each user as described in "Adding a User" on page 278. Users will then register with Helix Server once using their user names. Helix Server associates each user name with the unique player ID in its database. It validates subsequent access attempts through the ID without asking the user to enter a user name. Because user names are associated with player IDs, though, you can use the names to set up permissions.

> **Note:** The database structure requires that you enter a password for each user, even though passwords aren't used with player ID validation. You can therefore enter the same password for each user.

Setting Up Commerce Rules

Helix Server provides a default commerce rule called SecurePlayerContent that authenticates access attempts through player IDs. You do not have to change any settings to use this default rule to secure on-demand content. To modify this rule or create a new one, refer to the instructions in "Adding or Modifying Commerce Rules" on page 289. Note the following about commerce rules used with player ID validation:

- By default, the **Protected Path** is /secure/player rather than /secure, which is used with user name and password validation. The /secure/player path corresponds to a player subdirectory of the Secure directory. This subdirectory is *not* created during installation, however, so you need to create it to use player validation.

  Tip: You can use another directory on your network. To do this, follow the instructions in "Placing Secure Clips in Other Directories" on page 277, modifying your commerce rule appropriately.

- By default, **User Permissions Database** is set to PlayerContent. Change the database choice if you are using a different database. If you do not want to evaluate permissions, and wish simply to grant all authenticated users access to all content in the protected path, select Do Not Evaluate User Permissions.

  Note: If you plan to use permissions for individual viewers, follow the instructions in "Granting Permissions" on page 294.

- For **Credential Type**, you must always select Use Player Validation for player ID authentication to work.
- Click **Redirect Unauthenticated Players** to enter a fully qualified HTTP or RTSP redirection URL for viewers who have not registered, or who have chosen not to transmit their player GUIDs.
- To validate live broadcasts through player IDs, create a new commerce rule as described in "Adding or Modifying Commerce Rules" on page 289, and set **Protected Path** to /broadcast/secure/player (for broadcasts from the latest version of RealProducer) or /encoder/secure/player (for broadcasts from RealProducer G2 through 8.5). On the encoder, specify the virtual path secure/player/ along with the stream name when starting the broadcast.

Modifying the GUID Database

The Helix Administrator Commerce page allows you to select the database that registers player IDs. You do not need to change the default setting if you are using the default `PlayerContent` database. You need to change the GUID database settings only if you are not using the default database, or you want to change the prefix used in URLs to register users.

➤ To modify the GUID database:

1. Click **Security>Commerce**.
2. Select an existing database name, or click the "+" icon to create a new database entry, naming the database in the **Edit Description** field. The name is for your reference only.
3. Enter a prefix in the **Player Registration Prefix** field. You'll use this prefix in URLs that viewers click to register their player IDs, as described in "Creating Registration URLs" on page 300.
4. Select the database to use through the **Player Database** pull-down list.
5. Click **Apply**.

Creating Registration URLs

To register individual RealPlayers, you create a link that viewers click once to add their players' GUIDs to the database. The registration URL, which users can enter in the **File>Open Location** box of RealPlayer, looks like this:

`rtsp://helixserver.example.com/register`*`Name`*`!`*`clip.rm`*

If you add the URL to a Web page link, you use an HTTP URL and `/ramgen/`, as shown here:

`http://helixserver.example.com/ramgen/register`*`Name`*`!`*`clip.rm`*

Note the following:

- The registration prefix is `register` by default, but can be changed as described in "Modifying the GUID Database" on page 300.
- After the Helix Server address, you combine the registration prefix with the viewer's user name. For example, you might use `registerMiguel21`.
- For *`clip.rm`*, specify a clip in the unsecured `Content` directory, not a clip in a secured directory. This can be any clip that the player can display. It may simply be a graphic that indicates a successful registration.

- Once a GUID is registered, subsequent registration attempts are ignored.

Writing Content URLs

After viewers contact Helix Server through the registration URL, they can begin to access content for which they have permission. Media clips go in the secure directory defined by the commerce rule, as described in "Setting Up Commerce Rules" on page 299. The default path is the `player` subdirectory (which must be created manually) of the `Secure` directory. Web page links to this content look like this:

```
http://helixserver.example.com/ramgen/secure/player/business/video1.rm
```

In the preceding example, `/business/` is an optional path that corresponds to an actual subdirectory in the protected path. These subdirectories can be any number of levels deep, and you can assign various users privileges for different subdirectories as described in "Granting Permissions" on page 294.

A Web page link to a live broadcast looks like this:

```
http://helixserver.example.com/ramgen/broadcast/secure/player/live.rm
```

Here, `/broadcast/` is the appropriate broadcast mount point, as described in Chapter 7. The encoder that sends the stream to Helix Server must supply the `/secure/player` virtual path along with the broadcast stream name.

> **Note:** As noted in "Setting Up Commerce Rules" on page 299, a commerce rule for the `/broadcast/secure/player` protected path is not predefined, so you must create it to secure broadcasts through player ID validation.

> **For More Information:** For background on URLs and links, see "Writing Links to Content" on page 85.

CHAPTER

14

ISP HOSTING

ISP Hosting features provide a way to allot connections to users. If you are an Internet Service Provider (ISP), you can use ISP hosting to deliver streaming media on behalf of your customers.

Understanding ISP Hosting

Helix Server works with your existing user accounts and directory structure to make users' media files available for streaming. You allocate a minimum and maximum number of connections for each account, based on the number of streams permitted by your license. Allocating on a per-connection basis, rather than by stream, ensures that all files, including SMIL files which reference multiple streams, will always be served.

User account information is stored in a text file, which lists path and connection information. You can list all user account information in a single file, or use separate files to make management easier. Within the user list file, you can create customized account path and connection information. Or, create a single entry that applies to all user accounts.

Links to Users' Hosted Content

Links to hosted content have the following format if used in a Web page:

`http://helixserver.example.com:`*`HTTPPort`*`/ramgen/~`*`username/filename`*`.rm`

The link which Helix Server uses, or which you can type directly into RealPlayer, has the following format:

`rtsp://helixserver.example.com:`*`RTSPPort`*`/~`*`username/filename`*`.rm`

For More Information: See "Writing Links to Content" on page 85.

Account Information

When Helix Server receives a request for streaming media, it looks at user account information, stored in user list files, to determine which user is hosting the requested content. User list files can list account information separately for each user, or can give generic information that applies to all users. Each account has three items associated with it:

- account name
- virtual path where the account's files will be stored
- minimum and maximum connections available to the account

Account information is stored in text files, called user list files. You can put all information into a single file or use separate files to make organization easier.

Connections Available for Each Account

Each account has a reserved number of connections and a maximum number of connections associated with it. The user list file can contain a generic account description that applies to all users, specific instructions about certain accounts, or a combination of the two.

The maximum setting refers to the highest number of connections that will be available for a particular customer's content. Anyone who tries to watch a clip after that account's maximum number of connections are in use will receive an error message, even if connections are available to other accounts.

The number of connections reserved for ISP hosting depends on the type of user record within the user list file:

- Specific user account descriptions
- Generic user account description

If you use a combination of account descriptions, be sure to read both topics in this section.

Specific User Accounts

The reserved setting ensures that the specified number of connections will always be available to clients that attempt to view a particular user's hosted media.

All reserved connections are subtracted from the overall number of connections available to Helix Server. The remaining connections are available for non-ISP-hosted content, or for hosted content that hasn't yet been

requested. For example, if your Helix Server is licensed for 50 connections, and you reserve 20 connections through the ISP hosting feature, there are 30 connections available for general use. Helix Server can use those remaining connections for streaming regular clips, or for users of ISP-hosted material that isn't yet reserved.

Reserved connections are only activated for accounts listed in the user files, and are activated as soon as Helix Server starts.

> **Tip:** To guarantee that connections will always be available for certain customers, list those account names in the user file, rather than using a generic scheme. Be careful to leave enough streams available for other use, however.

Users whose accounts are not specifically listed in the user list file default to the generic account description.

Generic User Account

For accounts not described in the user list file, minimum connections are not reserved until content is played from a user's account.

Other Considerations

It is possible to reserve more connections than are included in your license. In this case, connections are distributed on a first come, first served basis.

For example, if your Helix Server is licensed for 50 connections, and you create a generic account that reserves a minimum of 3 connections for all 25 customers, all the connections will be reserved for ISP hosting customers. Since 75 connections are reserved, but only 50 connections are available, the first 50 customers who connect will be able to play content, but anyone connecting after that will not.

Tracking Account Usage

Like any content it serves, Helix Server creates a record for each file it serves in the access log. The fourth field in each record of the access log, identified by the `GET` statement, lists the path and filename of each clip served. Compare this information to the path information you've set up to determine how many clips have been streamed from each account.

In most cases, Helix Server creates one record for each clip served. However, SMIL presentations served from your clients' accounts may generate more than one record. You can see which records are part of a SMIL presentation by

looking at the final number in the record (present if Logging Style is 5). These numbers will match if they are from the same SMIL presentation. See "SMIL Files" on page 325.

Account-Based ISP Hosting

The `GET` statement will include the ISP hosting mount point and the user's account name (beginning with the ~ character). For example, a file with the URL:

```
http://helixserver.example.com/ramgen/~chris/file.rm
```

would appear in the access log as:

```
GET ~chris/file.rm
```

Dedicated ISP Hosting

Because dedicated ISP hosting Helix Servers can only stream content for users, and not stream any other type of content, the access log will only show material streamed for ISP customers. The mount point will always appear.

The GET statements will show the directory portion of the URL.

A file with the following URL:

```
http://helixserver.example.com/ramgen/r/ra/rabrams/file.rm
```

would appear in the access log as:

```
GET r/ra/rabrams/file.rm
```

Dedicating Helix Server to ISP Hosting

Helix Server can be dedicated to only serving hosted content. If you use this option, Helix Server cannot stream media files from any other directories.

This option requires that users' directories are arranged in a hierarchy. Features available in dedicated hosting are the same as in account-based hosting.

URLs used in this type of hosting have a different format. Rather than use a tilde (~) to alert Helix Server to an upcoming ISP request, this method relies on a directory structure shown in the URL.

```
http://helixserver.example.com/ramgen/s/sa/sandy/media/filename.rm
```

or

```
rtsp://helixserver.example.com/s/sa/sandy/media/filename.rm
```

A comparison of the two styles is shown below. Use only one style on a particular Helix Server.

Comparison of Account Identification Styles

Issue	Account-Based Hosting	Dedicated Hosting
Hosted material	Can host content for ISP users; can also serve ordinary streamed content.	Can only host content from user accounts. Cannot serve other content.
User directory structure	Works with any directory structure; enables different structures or locations. Users may have their own subdirectories.	Works with a hierarchical directory structure, especially an alphabetic one. Organization of directories must be the same for all users. Users may have their own subdirectories.
Reserving connections	Can reserve number of connections available for material streamed from certain accounts.	Cannot guarantee any reserved connections.
User settings	Some users can have customized settings, while generic connection settings describe all other users.	All users have identical settings.

Compatibility with Earlier Versions of RealSystem Server

If you used ISP Hosting in RealSystem Server versions 3 through 5, you can still use the UserList from your previous configuration file. Refer to "Creating User Lists From Earlier Versions" on page 314 for instructions on how to use your existing UserList.

Earlier versions of RealSystem Server listed minimum and maximum settings for the number of streams available to each account. In Helix Server, those settings now refer to the number of connections available to each account. This enables customers to serve SMIL files—which may reference several streams simultaneously—without running out of streams.

This manual uses new terminology for the methods of referring to account structures described in editions of *RealServer Administration Guide* previous to version 8.

- "Naming Convention One" is now described as the usual method of configuring user list files.

- "Naming Convention Two" is described here as a dedicated hosting Helix Server, a special case.

Although they have different names in this manual, the user directory structures and user list structures in each method are functionally identical to the methods used in earlier versions of RealSystem Server.

Example ISP Hosting Scenario—Northwest ISP

Throughout this chapter, we'll use the example of an ISP who sets up Helix Server to host its users' media files. Northwest ISP hosts content for customers in a three-state area in the United States' Pacific Northwest. Users' directories are organized according to the state in which the users live—Washington, Oregon, and Idaho:

```
C:\home\washington
C:\home\oregon
C:\home\idaho
```

Individual accounts are located immediately below these directories:

Chris Anderson's account:	`C:\home\washington\canderson`
Pat Brown's account:	`C:\home\washington\pbrown`
Lee Adams' account:	`C:\home\oregon\ladams`
Sandy Chu's account:	`C:\home\oregon\schu`

Other accounts:	`C:\home\idaho\alex`
	`C:\home\idaho\sam`
	`C:\home\idaho\tracy`

The links to these users' files look different than other Helix Server links. These all contain a tilde (~) and the user's usernames or account names:

```
http://helixserver.example.com/ramgen/~chris/file.rm
http://helixserver.example.com/ramgen/~lee/file.rm
http://helixserver.example.com/ramgen/~pat/file.rm
http://helixserver.example.com/ramgen/~sandy/file.rm
```

Users' Directory Structures

Helix Server matches your existing directory locations of users' files, even if you use different structures for different users. Typically, user directories are named with the username of the account; the username is included in the URL.

Customers' media files are stored in their directories. If they place files in a subdirectory of their main directory, that subdirectory must be included in the URL.

Directory Structures in Dedicated Hosting

A Helix Server used exclusively for hosting users' streamed media from accounts based on a strict directory structure uses an alternate method of identifying accounts. In the user list, you identify how far down the directory path to look for individual user accounts; this requires that the accounts must all be at the same level.

In the following example, accounts are divided into separate directories, according to an alphabetic arrangement:

```
...
/UserAccounts/r/ra/rabrams
/UserAccounts/r/ra/radams
...
/UserAccounts/s/sa/sanderson
/UserAccounts/s/sb/sbraun
/UserAccounts/s/sb/sbrown
/UserAccounts/s/sc/schu
...
```

Users may have their own subdirectories. If they place files in a subdirectory of their main directory, that subdirectory must be included in the URL.

Of course, if you use the account-based style of identifying customer directories rather than the method described in this section, you can also dedicate Helix Server to only hosting streamed media for customers, but other streaming options are still available.

Setting Up ISP Hosting

There are three steps for configuring Helix Server to host users' media files:

1. Create the user list file.

   This file establishes account information, such as reserved connections and maximum connections. For more information, see "Creating the User List" on page 310.

2. Configure Helix Server.

The configuration file indicates where to find the user lists, and completes the pathing information needed to located the users' media. For more information, see "Configuring Helix Server" on page 314.

3. Creating the links to content.

   You will need to tell customers what format they should use in creating their links. For more information, see "Linking to ISP Content" on page 316.

Creating the User List

Create the user list, and store it anywhere that is accessible to Helix Server.

The user list is a text file with the following format:

```
UserList [
{account, /path/, minimum_connections, maximum_connections}
]
```

where:

account is either a specific user name, or ~* to indicate that all accounts will use the same settings. See "Using Multiple User List Files" on page 312 for examples of how multiple accounts can be shown in a user list.

> **Note:** Dedicated hosting Helix Servers use a slightly different format. Refer to "Dedicated Hosting User File Format" on page 313 for the correct format to use.

/path/ gives information about the location of users' media files. It does not necessarily refer to an actual location or portion of a location; instead, it is a logical method of grouping the users.

minimum_connections is the minimum number of connections reserved for this user. 0 indicates that no connections are reserved. See "Connections Available for Each Account" on page 304 for more information.

maximum_connections is the maximum number of connections available to this user. 0 indicates that no connections may be used. See "Connections Available for Each Account" on page 304 for more information.

> **Tip:** You can include comments in the file by preceding a line with a semi-colon (;).

Example—User List File

In this user list file (shown in the left column of the table), users are grouped according to their geographic location. Two users, Chris and Pat, are in the Washington (wa) group. Two other users, Lee and Sandy, are in the Oregon group.

Sample User List

User List File Contents	Matching Customer Name
`UserList [`	
`{chris, /wa/canderson/, 2, 5},`	Chris Anderson
`{lee, /or/ladams/, 0, 100},`	Lee Adams
`{pat, /wa/pbrown/, 2, 50},`	Pat Brown
`{sandy, /or/schu/, 1, 35},`	Sandy Chu
`]`	

Listing Individual Accounts

If each account has different settings, create a separate record for each user, as in the example above.

Listing Generic Accounts

If you have a large number of accounts to create, and they will all use the same number of connections, create a single entry that refers to all accounts generically:

```
UserList [
{~*, /path/, minimum_connections, maximum_connections}
]
```

In the following example, one connection is reserved for each person, and the maximum number of connections available for any account is 35. (There are some restrictions on whether the connections are actually reserved; see "Connections Available for Each Account" on page 304.)

```
UserList [
{~*, /users/, 1, 35}
]
```

Combining Individual Account Listings with a Generic Listing

Custom account information and generic settings can be combined in a single user list. Combining them is convenient if most users have the same settings,

but a few have different number of connections reserved, or use different paths:

```
UserList [
{username1, /path/, minimum_connections, maximum_connections}
{username2, /path/, minimum_connections, maximum_connections}
{username3, /path/, minimum_connections, maximum_connections}
...
{~*, /path/, minimum_connections, maximum_connections}
]
```

In the following example, customized accounts for four users have been created, and all other accounts will use the default settings shown in the last entry:

Sample User List

User List	Customer Name
`UserList [`	
`{chris, /wa/canderson/, 2, 5},`	Chris Anderson
`{lee, /or/ladams/, 0, 100},`	Lee Adams
`{pat, /wa/pbrown/, 1, 35},`	Pat Brown
`{sandy, /or/schu/, 1, 5},`	Sandy Chu
`{~*, /id/, 1, 35}`	All others not specified above
`]`	

Using Multiple User List Files

You can create as many user lists as you want; using multiple files can make administration easier. For example, an ISP provider might include commercial accounts in one user list file and personal accounts in another file.

Helix Server loads the user lists in the order they appear in the configuration file, and any settings in subsequent files override settings in previously-loaded files. If the same user name appears in more than one list, Helix Server uses the settings in the last user list.

Because of this behavior, bear in mind the following considerations when using multiple user list files:

- An account name must not appear in more than one file.
- The generic account information (an entry beginning with ~*) must be used carefully. Include it only in the first-loaded user list file. If you include it in the last file, Helix Server will ignore all the other user lists.

Re-Reading an Updated User List File

Once you have created the user list file and the ISP hosting feature is in use, you must instruct Helix Server to re-read the user list.

- On Windows-based platforms, after you edit a user list file, you must restart Helix Server for the changes to take effect.
- On UNIX-based platforms, you can use the SIGHUP command to instruct Helix Server to re-read the user list files. See "Helix Server Restart" on page 396.

Dedicated Hosting User File Format

The format of the user list file in dedicated hosting is nearly the same as the account-based method, with these exceptions:

- Create only one user file. You cannot use more than one with the dedicated hosting server.
- Create only one entry in the user file. This entry applies to all user accounts.
- Instead of giving an account name, you indicate how far to traverse the user directory structure in order to find the unique user directories.

Use the following format:

```
UserList [
{*n, /path/, minimum_connections, maximum_connections}
]
```

where *n* is a number that represents the level of directory at which individual user directories appear.

Example

In the following example, all user accounts are located under a subdirectory of the UserAccounts directory. The unique directories are located at the fourth directory level (rabrams, radams, sanderson, and so on).

```
...
/UserAccounts/r/ra/rabrams
/UserAccounts/r/ra/radams
...
/UserAccounts/s/sa/sanderson
```

```
/UserAccounts/s/sb/sbraun
/UserAccounts/s/sb/sbrown
/UserAccounts/s/sc/schu
...
```

The user list for this example uses `4` for the value of *n*:

```
UserList [
{*4, /UserAccounts/, 1, 15}
]
```

URLS created with this method have the following format:

`rtsp://helixserver.example.com/`*directory1/directory2/directory3/filename*

For example,
`rtsp://helixserver.example.com/UserAccounts/r/ra/rabrams/band.rm`

Creating User Lists From Earlier Versions

Recycle your UserList entry from the configuration file of earlier versions. If your UserList is long, you may want to create more than one file.

After you create the new user list file, follow the instructions in "Configuring Helix Server".

➤ To create a user list from existing settings:

1. Open your old configuration file in a text editor.
2. Locate the `UserList` entry.
3. Copy and paste the existing `UserList` setting into a new text file.
4. Save the file. You can store it in any directory that is available to Helix Server.

Another item from the previous configuration file, `UserDir`, does not have an equivalent.

Configuring Helix Server

You will need to make a note of the values for `/path/` that you used in the user list file.

These instructions describe how to create a separate mount point for each customer category, which means customer files can be stored in separate base paths or drives.

➤ To configure Helix Server for ISP Hosting:

1. First, look in the user list files at the /path/ settings you have used. You will need this information in Step 11.
2. Click **Server Setup>Mount Points**.

   You will add a mount point for each path in the User List file, give it a name, and indicate a base path.
3. Click the "+" icon and enter a mount point description in the **Edit Description** box.
4. In the **Mount Point** box, type a name for the new mount point.

   In our example, type /wa_isp/.
5. In the **Base Path** box, type the location in which these paths should be mapped.

   In our example, type C:\home\washington.
6. Repeat Step 3 through Step 5 for each mount point and base path combination.

   In our example, we used the following settings:

Example Settings

Mount Point	Description	Base Path
/wa_isp/	ISP Content (Washington users)	C:\home\washington
/or_isp/	ISP Content (Oregon users)	C:\home\oregon
/id_isp/	ISP Content (Idaho users)	C:\home\idaho

7. Click **Apply**.
8. Click **Content Management>ISP Hosting**.
9. In the **Translation Mounts** area, click the "+" icon and type a description in the **Edit Translation Mount Description** box.
10. From the **Mount Point** list, select the mount point that you want to use for this Translation Mount. (You created these in Step 2 through Step 6.)
11. In the **User Path** box, type the value of /path/ from the user list file.

    For each /path/ that appears in the user list file, repeat Step 9 through Step 11 to associate the /path/ with a translation mount.

In our example, we have created a separate user path for /wa/, for /or/, and for /id/.

Example User Paths

Translation Mount Description	Mount Point	User Path
Washington users	/wa_isp/	/wa/
Oregon users	/or_isp/	/or/
Idaho users	/id_isp/	/id/

12. In the User List files section, click the "+" icon. A generic user list name appears in the **Edit User List File Name** box. Type the correct path to the user list you created in "Creating the User List" on page 310. Be sure to give the full path. To add more than one user list this for each list you want to add. In dedicated hosting, reference only one user list file.
13. Click **Apply**.

Linking to ISP Content

You'll need to tell your customers what format to use for their links.

Links in a Web page use this format:

http://*address:HTTPPort*/ramgen/*~account/path/file*

Helix Server URL Components

Component	Meaning
http	The protocol used to initiate streaming.
address	Machine and domain name of Helix Server. IP address may be substituted.
HTTPPort	Port number where Helix Server listens for requests sent via the protocol listed at the beginning of the URL. This value is usually 80 or 8080; see "Defining Communications Ports" on page 63.
ramgen	Required when you link in a Web page.
account	User's account name.
ramgen	The mount point tells Helix Server how the clip should be served.
path	Optional.
file	The file name itself, including the extension.

For samples of links to use in the Web page, see "Example ISP Hosting Scenario—Northwest ISP" on page 308.

Links typed directly in the player or used in a Ram or SMIL file use the following format:

rtsp://*address*:*RTSPPort*/~*account*/*path*/*file*

The format is nearly the same as the link used in the Web page: the protocol is different, the port number (if any) matches the protocol, and Ramgen is omitted.

Dedicated Hosting Server

Links in a Web page use this format:

http://*address*:*HTTPPort*/ramgen/*directory1*/*directory2*/*path*/*file*

where:

Helix Server URL Components

Component	Meaning
http	The protocol used to initiate streaming.
address	Machine and domain name of Helix Server. IP address may be substituted.
HTTPPort	Port number where Helix Server listens for requests sent via the protocol listed at the beginning of the URL. This value is usually 80 or 8080; see "Defining Communications Ports" on page 63.
ramgen	The mount point tells Helix Server how the clip should be served.
directory1	Each directory that is part of the hierarchy of directories. The number of directories you list must match the *n* number in the user list file.
directory2	
path	Optional. Represents any subdirectories of the user's home directory.
file	The file name itself, including the extension.

Using the example in "Dedicated Hosting User File Format" on page 313, a link to file.rm in the user directory /UserAccounts/r/ra/rabrams would look like the following:

http://helixserver.example.com:HTTPport/ramgen/r/ra/rabrams/file.rm

ISP Hosting Used with Other Features

Users inherit many features of your Helix Server: hosting of on-demand content, and access control are available.

Other Feature	Notes
Authentication	Not available for hosted material.
Live unicasting, multicasting, and splitting	Not available for hosted material.
Monitoring statistics	As the administrator, you are able to view how many clients are connected to all material served by your Helix Server, using Server Monitor. In order to discern which material belongs to users, you must examine the paths of the individual clips in use. You also can see which clips have been served by reading the access log file.
Helix Proxy	Helix Proxy is able to cache your users' content, just as it can cache any on-demand files served by this Helix Server.

PART

VI

LOGGING AND MONITORING

This section deals with compiling statistics, creating reports, and monitoring Helix Server. You can create real-time reports about media streamed by Helix Server, for example, or just collect error messages in a simple log file.

CHAPTER 15

BASIC LOGGING

This chapter explains how Helix Server records information about client connections and other events. Using the log files, you can compile reports about system activity, gathering the statistical information you need.

> **Tip:** As described in Chapter 16, you can also create advanced access and error logs to record specific events that you want to monitor.

Understanding Basic Logging

The basic access log records statistics about client connections. The basic error log records error and informational messages about Helix Server operation. The log files are written as text files that you can open with any text editor, or parse with a script or application. As accesses or errors occur, Helix Server appends information to the end of the appropriate log file. The following sections introduce you to the basic log files and their features.

Basic Access Log

The basic access log records information about requests by media players, as well as browser requests for Helix Administrator pages. Using these logs, you can find out what clips were played, the times when media players connected, and so on. This information can help you determine which clips are most popular, for example.

The default access log is `rmaccess.log`, which is located in the `Logs` subdirectory of the main Helix Server installation directory. However, information about which authenticated files have been accessed is stored in `reglog.txt` and `accesslog.txt`, which are described in "Logs Directory" on page 406. Requests for streams that will be cached are stored in the cached requests log.

Logged Information

The basic access log provides seven logging styles that determine the amount of information gathered on each access attempt. In general, each style builds on the preceding style, adding more information. For instance, logging style 0 gathers the least amount of information. Logging style 1 includes the style 0 information, and adds more information, and so forth. You choose just one logging style for the entire log.

> **For More Information:** The section "Basic Access Log File Format" on page 326 explains the logging styles and information fields.

Media Player Statistics

All logging styles can record statistics about a media player's playback experience. These statistics let you learn how many media packets were dropped, for instance, or whether the viewer paused the clip. There are four types of client statistics. You can use any combination of these statistics types, up to all four. Or, you can turn off client statistics gathering entirely. As well, users may choose not to report statistics.

> **For More Information:** See "Client Statistics" on page 340.

Basic Error Log

The basic error log contains information and error messages about Helix Server operation. By looking for patterns of errors, you can troubleshoot and correct possible problems on your site. The default file name is `rmerror.log`, and the file is generated in the `Logs` subdirectory of the main Helix Server installation directory. Helix Server records an entry in this log only when an error occurs. Until an error happens, the file does not exist. The error log uses the following syntax:

```
***date time servername(process_ID): error_message
```

The following table explains these fields in the error log file.

Basic Error Log Fields

Entry	Meaning
***	Three asterisks indicate an error. Informational messages are not preceded by asterisks.
date	Date on which the error occurred, given in the form dd-Mmm-YY, as in 26-Apr-02.
time	Time the error occurred on the Helix Server clock, given in the form HH:MM:SS.xyz, as in 21:05:10.614.
servername(process_ID)	Helix Server name, followed by the process ID in parentheses.
error_message	Text of the message.

Note: If you receive a message that refers to a fatal error, contact the RealNetworks Technical Support Department for assistance.

For More Information: For details about error messages recorded in the error log, refer to *Helix Server and Helix Proxy Troubleshooting Guide*.

Log File Rolling

The access and error log files can grow indefinitely as they accumulate data. To keep log files manageable, you can limit a log file to a specific size. You can also create a new log at a preset interval, such as every six hours or two weeks, depending on the amount of data you expect to log. Helix Server begins, or *rolls*, a new log file when the limit is reached. Rolled log files are named with the following format:

file_name.log.*timestamp*

The name and extension are set through Helix Administrator, as described in "Setting Up Basic Access and Error Logs" on page 349. The timestamp has the following format, using a 24-hour clock:

YYYYMMDDHHMMSS

For example, the following file was created on June 22, 2006, at 1:49.53 P.M:

rmaccess.log.20060622134953

Basic Access Log Used with Other Features

The following sections describe how information about various features appears in the access log file.

Helix Proxy

If clips are proxied, the access log provides additional information about the connection. For more information, see "Proxied Clip Information" on page 336.

Live Unicasting

Clients transmit data for live events at the conclusion of a broadcast. Entries will not appear in the access log until the live event is over, or the user clicks **Stop**. As described in "GET Statements" on page 337, the `GET` statement shows unicasted events starting with the live mount point (usually `/broadcast/` or `/encoder/`). Statistics type 3, which show user actions such as fast-forward and pause, are not available for live events.

SLTA

If you broadcast a prerecorded clip in an infinite loop using SLTA, and the client remains connected, no access record is created until the broadcast stops or the client halts.

Splitting

On transmitters, the access log does not show any records pertaining to the receiver connections. However, if the same event is transmitted to multiple Helix Servers, records will be created in the transmitter's access log. On receivers, the access log contains records for each clip delivered, and shows the splitting mount point.

Back-Channel Multicasts

Clips that were broadcast using back-channel multicasts can be identified with the *protocol* statement, such as `RTSPM`. The same clip delivered over unicast will show `RTSPT` if the TCP transport was used, or `RTSP` if UDP was used. For more information, see "File Name and Protocol" on page 330.

Scalable Multicasts

To prevent large, scalable multicasts from overwhelming Helix Server when statistics are sent, collect statistics with a Web server designed to handle large numbers of HTTP posts simultaneously. See "Gathering Client Statistics" on page 166 for instructions on configuring this feature.

Access Control and Authentication

The access log does not show whether access control rules are in use. Only clients with IP addresses approved by the access control rules, and that supplied the proper name and password (if required) are allowed to receive content. Authenticated content is identified by the `/secure/` mount point in the path shown in the `GET` statement. For more information, see "GET Statements" on page 337.

ISP Hosting

You can identify which on-demand files were served by the ISP hosting feature by comparing the file name in the `GET` statement to the `/path/` value in the user list file.

Monitoring

The Server Monitor shows files that are being viewed presently, whereas the access log provides a historical report of files that have been served. All of the files that Server Monitor shows will appear in the access log when they finish playing.

Helix Administrator

The access log file shows all files served by Helix Server, including all Helix Administrator pages. These appear in the `GET` statement and begin with `admin`. For more information, see "GET Statements" on page 337.

When the logging style is 5 or 6, the end of each record gives a presentation ID. For Helix Administrator pages, this number associates the elements on a particular page. All of the images that go with each page also appear in the access log. All files served that are related to a particular page are numbered sequentially.

SMIL Files

Each file in a SMIL presentation, including the SMIL file itself, generates a record in the access log. When the logging style is 5 or 6, all files have the same ID number. For example, the files `presentation.smil`, `presentation.rt`, `presentation.rp`, and `presentation.rm` all have the same number in the `presentation_ID` field, such as 432. If the SMIL file was requested through a URL that used Ramgen, an additional record is created for the Ramgen statement, and shows a different value for the `presentation_ID` field.

Ram Files

All of the files listed in a Ram file generate a record in the access log. Because Ram files may be served by a Web server, there may be no record created in the access log for the Ram file. When the logging style is 5 or 6, all of the files listed in a Ram file have the same `presentation_ID` number.

Basic Access Log File Format

Helix Server records each access request in a separate record written to a new line in the basic access log. Fields within a record are separated by spaces or by pipes (|). At least one record is created for every clip served. If the client requests a presentation that includes several clips, one record is created for each clip in the presentation. Two records are created for every proxied clip delivered by proxy pull-split or cache.

Logging Style

Helix Server provides seven logging styles, numbered 0 through 6. Styles 1 through 4 each include the information of lower logging styles. For example, style 3 collects the same information as styles 0, 1, and 2, as well as some additional information. The default is style 5, which adds a `presentation_ID` field to the information in style 2. The following sections describe which fields each logging style collects. The section "Access Log Fields" on page 329 explains the information logged in each field.

> **Tip:** Although square brackets in syntax typically indicate optional material, the square brackets shown in the following access log syntax actually appear in the access log records.

> **Note:** In the following examples, client statistics are not logged, so each entry shows `[UNKNOWN]` where the statistics fields would be. If you collect client statistics, therefore, each log entry will contain additional information. For more information, see "Client Statistics" on page 340.

Logging Style 0

Logging style 0 uses this format:

```
IP_address - - [timestamp] "GET filename protocol/version" HTTP_status_code
bytes_sent [client_info] [client_stats_results]
```

Here is an example of an actual log record, showing that 858,636 bytes of the requested clip were sent over RTSP:

```
207.188.7.125 - - [26/Jun/2006:10:31:44 -0700] "GET realvideo10.rm RTSP/1.0"
200 858636 [WinNT_5.0_6.0.10.714_RealPlayer_RN92PD_en_686] [UNKNOWN]
```

Logging Style 1

Logging style 1 follows this format:

```
IP_address - - [timestamp] "GET filename protocol/version" HTTP_status_code
bytes_sent [client_info] [client_stats_results] file_size file_time connected_time
resends failed_resends
```

The following sample log record shows the same information as logging style 0, but adds information on file size, clip timeline length, actual time streamed, and resent packages:

```
207.188.7.125 - - [26/Jun/2006:10:06:33 -0700] "GET realvideo10.rm RTSP/1.0"
200 858636 [WinNT_5.0_6.0.10.714_RealPlayer_RN92PD_en_686] [UNKNOWN]
926322 217205 1 0
```

Logging Style 2

This is the format for logging style 2, which is identical to style 1, except that it records a client ID, which may be a global ID or an ID set through a cookie:

```
IP_address - - [timestamp] "GET filename protocol/version" HTTP_status_code
bytes_sent [client_info] [client_ID] [client_stats_results] file_size file_time
connected_time resends failed_resends
```

Here is an example:

```
207.188.7.125 - - [26/Jun/2006:10:07:42 -0700] "GET realvideo10.rm RTSP/1.0"
200 858636 [WinNT_5.0_6.0.10.714_RealPlayer_RN92PD_en_686]
[8e07b707-19b7-448b-96b6-96c90151f2a6] [UNKNOWN] 926322 217 205 1 0
```

Logging Style 3

Logging style 3 follows this format. It builds on style 2 by adding information about the streams and the Helix Server that delivered the clip:

```
IP_address - - [timestamp] "GET filename protocol/version" HTTP_status_code
bytes_sent [client_info] [client_ID] [client_stats_results] file_size file_time
connected_time resends failed_resends [stream_components] [start_time]
server_address
```

This example shows the server and stream information added to the end of the record:

```
207.188.7.125 - - [26/Jun/2006:10:09:09 -0700] "GET realvideo10.rm RTSP/1.0"
200 858636 [WinNT_5.0_6.0.10.714_RealPlayer_RN92PD_en_686]
[8e07b707-19b7-448b-96b6-96c90151f2a6] [UNKNOWN] 926322 217 205 1 0
[1 1 0 0] [26/Jun/2006:10:05:14] 208.147.89.157
```

Logging Style 4

Logging style 4 adds information about the clip's average bit rate and number of packets sent:

```
IP_address - - [timestamp] "GET filename protocol/version" HTTP_status_code
bytes_sent [client_info] [client_ID] [client_stats_results] file_size file_time
connected_time resends failed_resends [stream_components] [start_time]
server_address average_bitrate packets_sent
```

Here is an example:

```
207.188.7.125 - - [26/Jun/2006:10:10:04 -0700] "GET realvideo10.rm RTSP/1.0"
200 858636 [WinNT_5.0_6.0.10.714_RealPlayer_RN92PD_en_686]
[8e07b707-19b7-448b-96b6-96c90151f2a6] [UNKNOWN] 926322 217 205 1 0
[1 1 0 0] [26/Jun/2006:10:05:14] 208.147.89.157 34816 488
```

Logging Style 5

Logging style 5, which is the default style, does not build on the preceding styles. Instead, it copies style 2 and adds a presentation ID that helps you keep track of presentations that contain multiple clips:

```
IP_address - - [timestamp] "GET filename protocol/version" HTTP_status_code
bytes_sent [client_info] [client_ID] [client_stats_results] file_size file_time
connected_time resends failed_resends presentation_ID
```

The following is an example of a logging style 5 entry:

```
207.188.7.125 - - [26/Jun/2006:10:11:03 -0700] "GET realvideo10.rm RTSP/1.0"
200 858636 [WinNT_5.0_6.0.10.714_RealPlayer_RN92PD_en_686]
[8e07b707-19b7-448b-96b6-96c90151f2a6] [UNKNOWN] 926322 217 205 1 0 124
```

Logging Style 6

Logging style 6 includes all available fields. To the fields found in logging style 4, it adds the presentation ID found in logging style 5, and appends two additional fields to the end of the entry:

```
IP_address - - [timestamp] "GET filename protocol/version" HTTP_status_code
bytes_sent [client_info] [client_ID] [client_stats_results] file_size file_time
connected_time resends failed_resends [stream_components] [start_time]
server_address average_bitrate packets_sent presentation_ID
bitrate_adaptations media_adaptations
```

Here is an example:

```
207.188.7.125 - - [26/Jun/2006:10:10:04 -0700] "GET realvideo10.rm RTSP/1.0"
200 858636 [WinNT_5.0_6.0.10.714_RealPlayer_RN92PD_en_686]
[8e07b707-19b7-448b-96b6-96c90151f2a6] [UNKNOWN] 926322 217 205 1 0
[1 1 0 0] [26/Jun/2006:10:05:14] 208.147.89.157 34816 488 124 1 0
```

Access Log Fields

The following table summarizes the various logging fields that may appear in an access record, and indicates which logging styles include the fields. The following sections describe the basic access log fields in detail.

Access Log Fields

Log Field	Logging Styles	Reference
IP_address	0, 1, 2, 3, 4, 5, 6	page 330
[timestamp]	0, 1, 2, 3, 4, 5, 6	page 330
"GET filename protocol/version"	0, 1, 2, 3, 4, 5, 6	page 330
HTTP_status_code	0, 1, 2, 3, 4, 5, 6	page 330
bytes_sent	0, 1, 2, 3, 4, 5, 6	page 331
[client_info]	0, 1, 2, 3, 4, 5, 6	page 331
[client_ID]	2, 3, 4, 5, 6	page 332
[client_stats_results]	1, 2, 3, 4, 5, 6	page 334
file_size	1, 2, 3, 4, 5, 6	page 334
file_time	1, 2, 3, 4, 5, 6	page 334
connected_time	1, 2, 3, 4, 5, 6	page 334
resends	1, 2, 3, 4, 5, 6	page 334
failed_resends	1, 2, 3, 4, 5, 6	page 334
[stream_components]	3, 4, 6	page 335
[start_time]	3, 4, 6	page 335
server_address	3, 4, 6	page 335
average_bitrate	4, 6	page 335
packets_sent	4, 6	page 335
presentation_ID	5, 6	page 335
bitrate_adaptations	6	page 336
media_adaptations	6	page 336

IP Address

The IP_address field gives the IPv4 or IPv6 address of the client, such as 123.45.123.45 or 1080:0:0:0:8:800:200C:417A. If media is being proxied to the client, the log displays the IP address of the proxy. For more information, see "Proxied Clip Information" on page 336. Following the IP address are two hyphens for compatibility with standard Web server log formats.

Timestamp

The [timestamp] field indicates the time that the record was written to the log file according to the Helix Server clock. It uses the following format:

```
[dd/Mmm/yyyy:hh:mm:ss TZ]
```

Here, TZ is the time zone expressed as the number of hours relative to Coordinated Universal Time (Greenwich, England). For example:

```
[26/Jun/2003:10:10:04 -0700]
```

File Name and Protocol

The "GET filename protocol/version" field lists the file name and path requested by the client. The path is everything in the URL after the port number. If the client requests a file that doesn't exist, UNKNOWN appears in place of the file name. Possible values for the application-layer protocol used to send the clip to the client are RTSP, MMS, and HTTP. In addition, a letter at the end of the string indicates which transport type was used:

(blank)	UDP connection
T	TCP connection
H	HTTP connection
M	Multicast

For example, RTSPT means that the clip was streamed using the RTSP protocol over a TCP connection. The version number indicates the edition of the protocol.

For More Information: See "GET Statements" on page 337.

HTTP Status Code

The HTTP_status_code field holds a return code that uses the HTTP standard error codes. It usually returns 200.

Bytes Sent

The bytes_sent field records the number of bytes transferred to the client. If the media is being proxied to the client using proxy caching or pull-splitting, the log displays this field in two records for each client connection, as described in "Proxied Clip Information" on page 336.

Client Information

The [client_info] field describes the version and type of client being used.

RealNetworks Clients

For RealNetworks clients, [client_info] uses the following format:

[*platform_version_client_type_distribution_language_CPU*]

The following information is recorded:

platform	Operating system client software runs on, such as WinNT, Mac, and so on.
version	Operating system version number.
client	Version number of the client software.
type	Type of client software.
distribution	Distribution code of the client software.
language	Language setting in client software.
CPU	Type of processor on which the client is running. If the processor does not have a hardware Floating Point Unit, the string no-FPU is appended to the end of the CPU field with no delimiter.

For example:

```
[WinNT_5.0_6.0.10.714_RealPlayer_RN92PD_en_686]
```

> **Note:** RealAudio Player 1 logs just two fields for [client_info]. They are *platform* and *client*.

Windows Media Player

For Windows Media Player, the [client_info] field records the player version like this:

```
[NSPlayer/7.1.0.3055]
```

QuickTime Player

For QuickTime Player, the client information records the player version and the operating system. For example:

```
[QTS (qtver=5.0.2;os=Windows NT 5.0)]
```

Unknown Clients

If client information can't be gathered because the request came from a client that chose not to send statistics, or from a browser requesting Helix Administrator pages, [UNKNOWN] appears in the [client_info] field.

Proxied Media

If media is proxied to the client, the log displays, at a minimum, the version and type of the proxied client. For media delivered by proxy pull-splitting or caching, RealNetworks Broadcast Receiver or the Helix Proxy version and type appear in an additional record. For example:

```
[linux-2.2-libc6-i586-server_RealProxy_9.0.2_RealNetworks]
```

For More Information: See "Proxied Clip Information" on page 336.

Client Identifier

For [client_ID], the basic access log can record an identification number for each media player. This may be a globally unique ID. For RealNetworks clients, it may be an ID based on a cookie. The following sections explain the possible field entries.

Cookie-Based ID for RealNetworks Media Players

The default settings for Helix Server and RealNetworks clients record a cookie-based ID for each client access attempt. Users can control whether IDs are transmitted and cookies are accepted, however. As well, you can disable the logging of GUIDs and the setting of cookies through Helix Server regardless of client configurations.

For More Information: See "Modifying the Basic Access Log" on page 349 for instructions on turning off client ID logging.

Globally Unique Identifier (GUID) for RealNetworks Media Players

If a RealNetworks client is configured to send a globally unique ID, it does so. For privacy protection, however, RealPlayer is set by default *not* to send a GUID. Because sending a GUID rests solely at the discretion of each user, users must change their default GUID settings for their GUIDs to appear in the access logs. In RealPlayer, the user command for controlling GUID reporting is **Tools>Preferences> Connection>Internet Settings**.

> **For More Information:** To review RealNetworks' Consumer Software Privacy Statement, see the Web page located at **http://www.realnetworks.com/company/privacy/software.html**

Cookie-Based IDs for RealNetworks Clients

If a RealNetworks client's GUID reporting is not enabled, Helix Server logs an ID based on a client cookie, setting the cookie on the first client access. In subsequent access attempts, the client returns the ID set in the cookie. The cookie-based ID is in the same format as the GUID, and is unique to each Helix Server that delivers content to that client. However, the same client returns a different ID to each Helix Server it contacts.

Although cookies are enabled by default in RealNetworks clients, a user can elect not to accept them. In this case, Helix Server records the ID value it attempted to set, but does not know that the cookie was refused. When the same client does not return a cookie-based ID on its next media request, Helix Server sends another cookie, again recording the ID of a cookie that is rejected. This happens on each media request from that media player, resulting in multiple records that list different IDs because the same media player refused the cookie each time.

> **Note:** In RealPlayer, the user command for controlling cookies is **Tools>Preferences> Connection>Internet Settings**.

Windows Media Player and QuickTime Player IDs

If Windows Media Player and QuickTime Player are configured to send their GUIDs, Helix Server records those ID values. If the players do not send GUIDs, Helix Server generates an ID for the log. In this case, the same media player may be identified by multiple IDs in the log. Windows Media Player and QuickTime Player do not support cookie-based IDs.

Unknown IDs

When Helix Server can't gather an ID because the client does not support GUIDs or cookies (as with a browser requesting Helix Administrator pages), empty square brackets—`[ ]`—appear in the `[client_ID]` field. If GUID reporting is disabled on the server or media player side, and Helix Server does not attempt to set a cookie-based ID, the `[client_ID]` field shows a series of zeroes instead of a unique client identifier:

```
00000000-0000-0000-0000-000000000000
```

Statistics Results

The [client_stats_results] field holds connection statistics sent by the client when it finishes playing a clip, as described in "Client Statistics" on page 340. If the client blocks connection statistics, or the statistics cannot be collected, the field appears as [UNKNOWN].

File Information

The file_size, file_time, and connected_time fields hold information about the requested clip or broadcast.

file_size

The file_size field lists the size of the file as reported by the Helix Server operating system. This reported size includes the media data as well as the file header and other non-media information. For live broadcasts, file_size is always 0.

file_time

The file_time field gives the total length, in seconds, of media stored in the media file. For live broadcasts, file_time is always 0. For SMIL files, this is always 20.

connected_time

The connected_time field (formerly called "sent_time") expresses in seconds how long the media player was connected to the server. Because RealNetworks media players close the connection when the clip reaches the end of its timeline or the viewer stops the clip, the connected_time value expresses the duration of the streaming session. Other media players, including QuickTime Player and Windows Media Player, keep the connection open until the viewer chooses another clip or closes the player. The values recorded for these players may therefore include time after which the clip stopped but the player remained idle.

Resend Information

The resends field lists the number of packets successfully resent because of transmission errors. The failed_resends field gives the number of packets not successfully resent in time to correct transmission errors.

Stream Components

The field [stream_components] is recorded only for RealNetworks media players. It explains the type of material sent, indicated in the following pattern:

```
RealAudio_stream RealVideo_stream Event_stream Image_maps
```

A value of 1 indicates that the clip includes this type of stream. The value 0 indicates that it does not. Thus, a clip that includes RealVideo and RealAudio but no event streams or image maps would appear in the access log as this:

```
1 1 0 0
```

Start Time

The [start_time] field gives the timestamp of when the clip began to stream, according to the Helix Server clock. It is identical in format to the timestamp at the beginning of each access record, but does not list the time difference from Coordinated Universal Time. Here is an example:

```
[26/Jun/2006:10:05:14]
```

Server Address

The server_address field lists the IPv4 or IPv6 address of the Helix Server that delivered the clip.

Average Bit Rate

The average_bitrate field lists the average bit rate of the stream in bits per second.

Packets Sent

The packets_sent field lists the total number of packets sent to the client.

Presentation ID

The presentation_ID field records a number used by all clips in the same SMIL or Ram presentation. SMIL files are also included in the log, and use the same number as their clips. For example, if the log entry is for a SMIL file, a video clip, and a GIF image all list presentation ID 437, you can conclude that the SMIL presentation consisted of that video and image. Helix Server assigns the IDs, which are recorded only with logging styles 5 and 6, when it transmits the clips.

Bit Rate Adaptations

The `bitrate_adaptations` field records an integer value that indicates how many times the stream speed upshifted or downshifted during the playback session. This occurs only in clips that encode multiple bit rates, such as SureStream RealVideo. For example, if a media player connection first uses a 350 Kbps stream, drops to a 225 Kbps stream, then returns to the 350 Kbps stream, the field value is `2`, indicating two shifts in encoding speed. A value of `0` means that no bandwidth shifting occurred, or that the media clip is not encoded for multiple bit rates.

> **Tip:** These values indicate the basic quality of service. Recurring, high numbers may indicate persistent network congestion problems.

Media Format Adaptations

The `media_adaptations` field is used with media formats that support multiple codecs for the same clip. The field value is an integer that indicates the number of times that the meda player shifted between the different formats. If a presentation starts out streaming a high-speed MPEG-4 encoding, for example, before shifting to a lower-speed H.263 encoding, the field value is `1`. A value of `0` means that no media format shifting occurred, or that the media clip was encoded using a single codec.

> **Note:** Shifting between media formats in a clip is not currently supported by Helix Server.

Proxied Clip Information

If clips are proxied, the basic access log provides additional information about the connection by creating two records for every clip delivered by proxy pull-splitting or caching. Proxied clips delivered by passthrough are recorded in the access log with one record. The following table outlines proxy-specific data

contained in the access log by the type of Helix Proxy delivery, and the access log field.

Access Log Data for Proxied Clips

Proxy Delivery	Access Log Record	Access Log Field		
		IP_address	bytes_sent	[client_info]
live pull-split	proxy control channel	Helix Proxy IP address	0	version and type of proxied client
	proxy live data channel	Helix Proxy IP address	number of bytes sent to proxy	RealNetworks Broadcast Receiver
on-demand cache	proxy control channel	Helix Proxy IP address	0	version and type of proxied client
	proxy cache data channel	Helix Proxy IP address	number of bytes sent to proxy	version and type of proxy server
passthrough	proxied client connection	Helix Proxy IP address	number of bytes sent to proxy	version and type of proxied client

Note: In the records demonstrating the control channel between Helix Proxy and Helix Server, bytes_sent is 0 (zero) because the actual data is sent on a separate data channel.

GET Statements

The GET statement within a basic access log record shows the path and file name of each file that Helix Server served, as well as the protocol and protocol version used to stream or broadcast the file. The following sections show sample entries for GET statements used with different types of on-demand and live content.

For More Information: To see the GET statement in context, refer to "Logging Style" on page 326.

On-Demand Content

The following table lists the formats in which each type of on-demand content is shown in the GET statements of the access log. For a SMIL presentation, a separate record is generated for the SMIL file and for each file in the presentation. When the logging style is set to 5 or 6, you can identify which

files are in the same presentation through the numeric identifier at the end of each access record.

GET Statements for On-Demand Content

Feature	Protocol	Example Statement in Access Log
On-demand streamed content	RTSP	"GET presentation/presentation.rm RTSP/1.0"
	HTTP	"GET presentation/presentation.rm PNH/10"
SMIL files (1 record for the SMIL file, one record for each file listed within the SMIL file)	RTSP	"GET presentation/presentation.smi" "GET presentation/presentation.rt" "GET presentation/presentation.rp" "GET presentation/presentation.rm"
ISP hosting—account-based	RTSP	"GET ~schu/music.rm RTSP/1.0"
	HTTP	"GET ~schu/music.rm PNH/10"
ISP hosting—dedicated	RTSP	"GET s/sc/schu/music.rm RTSP/1.0"
	HTTP	"GET s/sc/schu/music.rm PNH/10"
Helix Administrator activity	HTTP	"GET admin/index.html HTTP/1.0"
View source request (for on-demand and live clips)	HTTP	"GET viewsource/template.html HTTP/1.0"
Authenticated on-demand streamed content	RTSP	"GET secure/topsecret.rm RTSP/1.0"
	HTTP	"GET secure/topsecret.rm PNH/10"

Live Broadcasts

The following table summarizes the format in which each type of live content is shown in the basic access log. For live RealMedia streams from RealProducer , the broadcast mount point is typically `broadcast/` or `redundant/`. For RealProducer G2 through 8.5, it is `encoder/`. For earlier encoders, it is `live/`. For Windows Media broadcasts, the mount point is typically `wmtencoder/`, and for QuickTime and MPEG broadcasts it is `rtpencoder/`.

While most clips generate one access log record apiece, clips delivered over scalable multicasting generate two records for each client. One is for the `.sdp` file, and the other is for the live broadcast stream. However, if the user saves the `.sdp` file and connects by opening that file, rather than by clicking a link

on a Web page, only the live broadcast stream generates a record. For more on scalable multicasting, see "Setting Up Scalable Multicasting" on page 164.

Sample GET Statements for Live Content

Feature	Protocol	Example Statement in Access Log
Unicasted content, from RealProducer	RTSP	`"GET broadcast/live.rm RTSP/1.0"`
	HTTP	`"GET broadcast/live.rm PNH/10"`
Unicasted, redundant content	RTSP	`"GET redundant/live.rm RTSP/1.0"`
	HTTP	`"GET redundant/live.rm PNH/10"`
Unicasted content, from RealProducer G2 through 8.5	RTSP	`"GET encoder/live.rm RTSP/1.0"`
	HTTP	`"GET encoder/live.rm PNH/10"`
Unicasted content, from pre-G2 encoding source	RTSP	`"GET live/live.rm RTSP/1.0"`
	HTTP	`"GET live/live.rm PNH/10"`
SLTA content	any	same as live unicasted content
Authenticated live streamed content	RTSP	`"GET secure/broadcast/live.rm RTSP/1.0"`
	HTTP	`"GET secure/broadcast/live.rm RTSP/1.0"`
Push splitting—transmitter's access log	RTSP	No record is created.
Push splitting—receiver's access log	RTSP	`"GET broadcast/Japan/broadcast/live.rm RTSP/1.0"`
Pull splitting—transmitter's access log	RTSP	No record is created.
Pull splitting—receiver's access log	RTSP	`"GET broadcast/pull/Japan:2030/encoder/live.rm RTSP/1.0"`
Multicasting—back-channel	RTSP	`"GET encoder/live.rm RTSPM/1.0"`
Multicasting—scalable (two records are usually created)	HTTP and RTP	`"GET concert.rm.sdp HTTP/1.0"` `"GET concert.rm RTP/2.0"`

For More Information: Chapter 7 explains broadcast mount points.

Client Statistics

All logging styles can include client statistics, which are shown in the preceding sections as [client_stats_results]. There are four types of statistics, and the access log can record any combination of them. Each set of statistics is enclosed in square brackets, and begins with a prefix such as Stat1. If you log all four types of statistics, for example, the [client_stats_results] field looks like this:

[Stat1:*statistics_1*][Stat2:*statistics_2*][Stat3:*statistics_3*][Stat4:*statistics_4*]

Note that although other access log fields are separated by spaces, there is no space between the closing square bracket of one statistics type and the opening square bracket of the next statistics type. The following example shows logging style 5 (see page 328) collecting statistics type 1:

```
207.188.7.125 - - [26/Jun/2006:10:11:03 -0700] "GET realvideo10.rm RTSP/1.0"
200 858636 [WinNT_5.0_6.0.10.714_RealPlayer_RN92PD_en_686]
[00000000-0000-0000-0000-000000000000] [Stat1: 487 2 1 2 0
44_kbps_Stereo_Music_High_Response_-_RA8] 926322 217 205 1 0 124
```

The following sections describe the information gathered by each of the four statistics types. Statistics 1 and 2 report basic information about playback. Statistics 3 provides information about viewer actions. Statistics 4 reports advanced playback information from RealPlayer, including data that is useful for gauging quality of service. The default logging setting gathers statistics type 4. The following table lists the media players and versions that can send the various statistics types.

Media Players and Supported Client Statistics Types

Media Player	Statistics 1	Statistics 2	Statistics 3	Statistics 4
RealPlayer 2 and earlier	**yes**	no	no	no
RealPlayer 3 and later	**yes**	**yes**	no	no
RealPlayer 5 and later	**yes**	**yes**	**yes**	no
RealOne Player and RealPlayer 10	**yes**	**yes**	**yes**	**partial**
RealPlayer 11 and later	**yes**	**yes**	**yes**	**full**
Windows Media Player	**limited**	**limited**	**yes**	no
QuickTime Player	no	no	no	no
any other RTP-based player	no	no	no	no

Note the following about client statistics:

- As noted in the following sections, some statistics are not collected for Windows Media Player. In each case, `0` is typically entered for that statistic.
- Helix Server does not record client statistics for QuickTime Player. For each statistics type, `[UNKNOWN]` is logged.
- Users can choose not to send client statistics. On RealPlayer 10 and later, the command is **Tools>Preferences>Connection>Internet/Privacy**. If users select this option, `[UNKNOWN]` appears in place of that statistics field.
- The statistics interval, which is set in the Helix Server configuration file, affects how often statistics are reported. For more information, refer to the logging chapter of *Helix Server Configuration and Registry Reference*.

Statistics Type 1

Statistics type 1 gathers basic information about how successfully the media player received a stream. It also tells what codec the client used to decode the audio portion of the stream. For RealNetworks media players, these statistics apply only to a clip's audio stream. The fields are the following:

```
[Stat1: received out_of_order missing early late codec]
```

These fields provide the following information:

`received`	Total number of packets received by the client.
`out_of_order`	Number of packets received by the client out of order. These packets are reordered as the client plays the clip. This information is not recorded for Windows Media Player.
`missing`	Number of packets that the client requested, but that did not arrive.
`early`	Number of requested packets received early by the client. This information is not recorded for Windows Media Player.

late	Number of packets received too late by the client. This information is not recorded for Windows Media Player.
codec	For Windows Media Player, the names of the audio and video codecs used. For RealNetworks clients, the name of the audio codec used to encode the soundtrack. Possible values for RealNetworks players include: raac—RealAudio 10 AAC format (RealOne Player and later) ralf—RealAudio Lossless format (RealOne Player and later) cook—RealAudio version 6 format (RealPlayer G2 and later) atrc—RealAudio ATRAC3 format (obsolete) sipr—RealAudio version 5 formats (obsolete) dnet—RealAudio version 3 formats (obsolete) 28.8—RealAudio version 2, 28.8 format (obsolete) lpcJ—RealAudio version 2, 14.4 format (obsolete)

Statistics Type 2

Statistics type 2 provides details about the success of stream delivery, giving information about bandwidth requests. Resent packets are described in detail. This statistics type identifies which transport type was used to make the connection, and which audio codec played the clip. For RealNetworks media players, these statistics apply only to a clip's audio stream. This set of statistics uses the following format:

```
[Stat2: bandwidth available highest lowest average requested received late
rebuffering transport startup codec]
```

The fields provide the following information:

bandwidth	Clip bandwidth in bits per second.
available	Average bits per second available to the user while the clip was playing. This information is not recorded for Windows Media Player.
highest	Highest time between the client resend packet request and the packet resend arrival, in milliseconds. This information is not recorded for Windows Media Player.
lowest	Lowest time between the client resend packet request and the packet resend arrival, in milliseconds. This information is not recorded for Windows Media Player.
average	Average time between the client resend packet request and the packet resend arrival, in milliseconds. This information is not recorded for Windows Media Player.
requested	Number of resend packets requested by the client.
received	Total number of resent packets received by the client.

late	Number of resent packets received by the client too late.
rebuffering	Rebuffering percentage for the clip.
transport	Transport type for the connection. Values are: 0—UDP 1—TCP 2—IP Multicast
startup	Time after the media request that the client received the first clip data package, in milliseconds. The data may arrive before the clip starts playing.
codec	For Windows Media Player, the names of the audio and video codecs used. For RealNetworks clients, the name of the audio codec used to encoded the soundtrack. Possible values include: raac—RealAudio 10 AAC format (RealOne Player and later) ralf—RealAudio Lossless format (RealOne Player and later) cook—RealAudio version 6 format (RealPlayer G2 and later) atrc—RealAudio ATRAC3 format (obsolete) sipr—RealAudio version 5 formats (obsolete) dnet—RealAudio version 3 formats (obsolete) 28.8—RealAudio version 2, 28.8 format (obsolete) lpcJ—RealAudio version 2, 14.4 format (obsolete)

Statistics Type 3

Statistics type 3 provides detailed information about viewer action while playing clips, but not while receiving live broadcasts. It addresses advanced streaming features, notably ads and image maps. For example, you can find out when a viewer clicked on an image map or stopped the clip. Because each user may carry out several actions, the access log file may grow rapidly when you collect these statistics. Be sure to review the log file frequently, or set up log file rolling to keep the logs to a manageable size. This statistics type uses the following format:

[Stat3:timestamp|elapsed_time|action|;...*additional entries*...]

Records of activity are separated by a semicolon (;). Thus, the Stat3 record of a viewer pausing, resuming play, and watching to the clip's end looks like the following:

[Stat3:4360|2107|PAUSE|;8401|2107|RESUME|;12608|6321|STOP|;]

Timestamp

The initial timestamp field gives the time in milliseconds when the action occurred. It is relative to the connection time of the client. In the preceding

example, the first timestamp is 4360, meaning the action occurred at 4.360 seconds after the client connected.

Elapsed Time

The elapsed_time field records how many milliseconds into the clip timeline the action occurred. In the preceding example, the PAUSE action occurred at 2107, or 2.107 seconds into the clip timeline. Notice that the RESUME action also lists the same elapsed time because this action restarts the clip at the same point where it paused.

Action

The action field records one of several different actions such as STOP or PAUSE, as described below.

CLICK

Viewer clicked on the image map. Further information includes:

x-coord	Horizontal coordinate of the click.
y-coord	Vertical coordinate of the click.
action	Action that occurred. This is one of the following: PLAYER="*url*"—The URL of a media link the viewer clicked. URL="*url*"—The URL of a browser link the viewer clicked. SEEK="*destination*"—The seek destination point, in milliseconds.

PAUSE

The viewer paused the client.

RESUME

Resume play after a pause, seek, or stop.

SEEK

The seek destination point, in milliseconds.

STOP

End of clip reached.

RECSTART

Media player began recording the clip.

RECEND

Media player stopped recording the clip.

Statistics Type 4

Sent only from RealOne Player and later, statistics type 4 gathers most of the same information included in statistics type 1 and type 2, adding packet and bandwidth information for each stream, including the visual tracks of video clips. RealOne Player through RealPlayer 10 use the following format for statistics type 4:

```
[Stat4:stream_number|mime_type|codec|received|lost|resent|average_bandwidth
|current_bandwidth|;...information for next stream...|transport turboplay duration
clip_end]
```

The following is an example type 4 log entry for a RealVideo Clip sent by RealOne Player through RealPlayer 10:

```
[Stat4:2 audio/x-pn-realaudio|44_kbps_Stereo_Music_High_Response_-_RA8
|44100|940|0|0|;video/x-pn-realvideo|N/A|180889|2918|0|0| 1 0|1|0| 90 2]
```

RealPlayer 11 and later report additional fields to Helix Server 11 and later. These extra statistics are useful for determining media player start-up times and calculating end-to-end latency in live broadcasts. The following shows the format for type 4 statistics reported by RealPlayer 11 and later. The additional fields are shown in **bold**:

```
[Stat4:stream_number|mime_type|codec|received|lost|resent|average_bandwidth
|current_bandwidth|;...information for next stream...|transport turboplay duration
clip_end] startup_time play_time rebuffering_time average_latency|
minimum_latency|maximum_latency]
```

The following is an example type 4 log entry for a RealVideo Clip sent by RealOne Player through RealPlayer 10:

```
[Stat4:2 audio/x-pn-realaudio|44_kbps_Stereo_Music_High_Response_-_RA8
|44100|940|0|0|;video/x-pn-realvideo|N/A|180889|2918|0|0| 1 0|1|0| 90 2]
1228 754100 0 0|0|0
```

Stream Number

The `stream_number` field indicates how many media streams the clip contains. A video clip might have two streams, for example, one for the audio track and one for the visual track. Following this, information for each stream is reported.

Stream Information

Helix Server reports information for each stream. Information ends with a semicolon. For each stream, the following fields are reported:

`mime_type`	MIME type, such as `audio/x-pn-realaudio`.
`codec`	Codec used for the stream, such as `44_kbps_Stereo_Music_High_Response_-_RA8`.
`received`	Number of packets received.
`lost`	Number of packets lost.
`resent`	Number of packets resent.
`average_bandwidth`	Average bandwidth over the course of clip playback in bits per second.
`current_bandwidth`	The bandwidth in bits per second used when the statistics are reported.

Transport

The `transport` field indicates the transport protocol used for the connection. Values are:

`0`	IP Multicast
`1`	UDP
`2`	TCP
`3`	HTTP cloaked

TurboPlay

Three `turboplay` fields indicate the use and results of the RealPlayer TurboPlay feature. The three fields are separated by pipes, as shown here:

```
1|513234|1120
```

The following table lists the possible field values. Values for the second and third field vary depending on whether TurboPlay is on or off, as indicated in the first field.

TurboPlay Field Values

Field 1	Field 2	Field 3
0 (off)	Reason TurboPlay is off: 1—User preference. 2—Available bandwidth below 256 Kbps. 3—SureStream in use. 4—Excess rebuffering. 5—Presentation not enabled for TurboPlay. 6—Server not enabled for TurboPlay. 7—Live presentation not supported.	0 (not used)
1 (on)	Accelerated delivery rate in bits per second requested by TurboPlay.	Average buffering time in milliseconds for start of playback, seeking, and so on.

Duration

The `duration` field gives the time in milliseconds between the initial media player request and the first data packet received by the player. The player typically needs to receive several data packets before playing the stream on the viewer's computer.

> **Note:** The `startup_time` field records the time between request and data display.

Clip End

The `clip_end` field lists the reason the presentation ended. Possible values are:

0	end of presentation reached
1	stop command issued
2	reconnection required
3	redirection
PNR_*n*	error code *n* occurred

Startup Time

Reported by RealPlayer 11 and later, the startup_time field indicates the time in milliseconds from the initiation of the media request to the point when media begins to play.

Play Time

The play_time field records the total time in milliseconds that the media player played the streamed media. This excludes the initial buffering time, any rebuffering time, and viewer-initiated pausing. Only RealPlayer 11 and later report this statistics.

Rebuffering Time

For rebuffering_time, RealPlayer 11 and later report the cumulative time in milliseconds spent rebuffering the stream.

Latency Statistics

The three latency values provide information about how quickly RealPlayer rendered data for a live broadcast or a simulated, live broadcast using SLTA. Values are in milliseconds and are reported only by RealPlayer 11 and later. For a prerecorded, on-demand clip, all three fields always record the value 0. The three fields hold the following information:

average_latency	Indicates the average time to render a data packet once it has been received by RealPlayer.
minimum_latency	Indicates the fastest rendering of a data packet received by RealPlayer.
maximum_latency	Indicates the slowest rendering of a data packet received by RealPlayer.

Note: These fields report live broadcast latency on RealPlayer only. Measuring full broadcast latency requires measuring latency introduced by RealProducer, Helix Server, and the network. For more information, refer to "End-to-End Latency Reduction" on page 192, as well as to the broadcast chapter in *RealProducer User's Guide*.

Setting Up Basic Access and Error Logs

The following sections explain how to set up basic access and error logging. These logging templates are turned on by default. You may want to change certain default options, however. You can turn off the log files generated through these templates, but you cannot delete the templates.

Modifying the Basic Access Log

The basic access log is preconfigured to gather basic client statistics for media player requests and write them to a text file. You may want to change the logging style and client statistics types, as well as set up log file rolling.

➤ To modify access logging:

1. Click **Logging & Monitoring>Basic Logging**.
2. For **Logging Style**, choose a number from 0 to 6. The default is `5`. For information about the logging style, see "Logging Style" on page 326.
3. If you do not wish to collect client identifiers, choose `Yes` from the **Disable Client GUIDs** pull-down list. This eliminates the collection of client global identifiers, as well as cookie-based IDs. For more information, see "Client Identifier" on page 332.

   Tip: Cookie-based IDs are also disabled on Helix Server if you choose `Yes` for **Disable Client Guids**.

4. The **Domain Cookie ID** pull-down list is set to `Enabled` by default. This means that Helix Server attempts to set a cookie on each client. This cookie provides a client ID logged in place of a suppressed GUID when the client requests content. To disable cookie setting, select `Disabled`. For more information, see "Client Identifier" on page 332.
5. In the **Client Stats** check boxes, select the types of client statistics that each media player reports. You can choose any combination of statistics, or deselect all boxes to gather no client statistics. The default settings are `Type 4`. For more information, see "Client Statistics" on page 340.

   Tip: If you gather statistics type 3 or 4, the access log file size will grow rapidly. In this case, be sure to review the log file frequently, or use log file rolling.

6. You can choose to create a new log file at certain intervals, as described in "Log File Rolling" on page 323.
   a. To create a new log file at regular intervals, set the period through the **Log Rolling Frequency** pull-down lists. You can roll the log hourly, daily, weekly, or monthly.
   b. To limit the log file by size, type the maximum number of Megabytes in the **Log Rolling Size** box.

      Tip: Generally, you limit log files by frequency or size. You can select both methods, however, to create log files according to the first limit reached. For example, you can create a new log file whenever the preceding file reaches 10 Megabytes in size, or has recorded 3 days of activity, whichever comes first.

7. For **Access Log File**, you can specify the path and file name for the log file. If you leave this field blank, Helix Server records access information in a file named `rmaccess.log` in the `Logs` subdirectory of the Helix Server main directory.
8. Click **Apply**.

Modifying the Basic Error Log

The basic error log captures error information and writes it to a text file. It requires no configuration and cannot be turned off. You may want to set up log file rolling, though, or specify a different location and name for the log file. For information about the error log syntax, see "Basic Error Log" on page 322.

➤ To modify the basic error log:

1. Click **Logging & Monitoring>Basic Logging**.
2. You can choose to generate a new error log file at certain intervals, as described in "Log File Rolling" on page 323.
   a. To create a new log file at regular intervals, set the period through the **Log Rolling Frequency** pull-down lists. You can roll the log hourly, daily, weekly, or monthly.
   b. To limit the log file by size, type the maximum number of Megabytes in the **Log Rolling Size** box.

Tip: Generally, you limit log files by frequency or size. You can select both methods, however, to create log files according to the first limit reached. For example, you can create a new log file whenever the preceding file reaches 10 Megabytes in size, or has recorded 3 days of activity, whichever comes first.

3. In the **Error Log File** field, you can specify the log file name, along with a relative or absolute path. Leave this field blank to use the default path of the `Logs` subdirectory under the main Helix Server directory. The default file name is `rmerror.log`.
4. On Windows NT systems, you can send error and informational messages to the Windows Event Viewer. For **NT Event Log Filter**, select the NT error level you want to assign to Helix Server error messages.
5. Click **Apply**.

CHAPTER

16

ADVANCED LOGGING

The advanced logging feature allows you to monitor specific types of events and information that occur on Helix Server. You can use this feature to create reports about any type of activity you choose. This chapter explains how to use the advanced logging templates.

Understanding Advanced Logging

The highly flexible advanced logging feature allows you to gather the exact information you want, reporting it at any time to different outputs such as the screen or a text file. You can use this feature to gather information about current Helix Server client connections, for example. Logging templates define the information and format for a log report.

> **Tip:** The basic access and error logs, which Chapter 15 describes, are easy to set up and capture a great range of information. You may find them easier to use than defining your own logging templates as described in this chapter.

The Helix Server Registry

To get information for reports, advanced logging relies on information stored in the Helix Server registry, which is distinct from the main registry on Windows operating systems. The registry contains information about most aspects of Helix Server. Although the registry is an extension of Helix Administrator, there is no link to it from any Helix Administrator page. However, you can display the registry by opening the following URL in a browser, and entering your user name and password for Helix Administrator:

`http://`*`address:AdminPort`*`/admin/regview.html`

Registry Variables

The Helix Server registry stores information in variables such as `LiveConnections.Count`. Each variable reports a specific type of value or setting, such as real-time data on client connections and server health. When you create a logging template, you add variables to your report by selecting them from a pop-up HTML list. Within a report template, variables are always preceded and followed by percent signs, as in `%LiveConnections.Count%`.

Global Variables

Through the variables list, you can select global variables, such as the time of day, that are derived from the operating system rather than extracted from the Helix Server registry. The following table lists the global variables that you can include in reports.

Global Variables

Variable	Description
`%Date%`	Indicates the current date in the format `MM/DD/YY`.
`%Time%`	Provides the current time of day in the local time zone in the format `HH:MM:SS`.
`%GMTime%`	Displays the current Greenwich Mean Time in the format `HH:MM:SS`.
`%TZDiff%`	Indicates the difference between local time and Greenwich Mean Time. For example, the output for Pacific Standard Time is `-0800`.
`%Hour%`	Displays the current hour by local time zone in the format `HH`.
`%Min%`	Indicates current minute in the format `MM`.
`%Sec%`	Adds the current second in the format `SS`.
`%%`	Creates a percent sign (%).

Template Types

You add the registry variables that you want to track to a report template, which defines how often the selected information is reported, as well as where the report is delivered, such as to a file or to the console. You can use four types of templates:

- Interval

  Interval templates log information at regular intervals, such as every hour. You can define exactly how much time elapses between report output.

Interval reports are useful for producing regular status reports about Helix Server health, for example.

- Watch

  With a watch template, you can set a watch on a certain variable, or group of variables, generating a report when information changes. A watch template is useful for reporting errors, for example, because a report is generated only when an error occurs.

- Client Stats

  A client statistics template periodically reports statistics from all media player connections. It can generate reports about the number of resent packets and a media players average bit rate, for example. You can generate a report when each media player disconnects, or at periodic intervals, such as every minute.

  > **For More Information:** The section "Generating Client Statistics Reports" on page 365 provides an example of how to gather client statistics.

- Session

  When a system component connects or disconnects, Helix Server dynamically adds and deletes variables from its registry. A session template reports on this activity when a component other than a media player (such as a live encoder) connects or disconnects.

  > **For More Information:** See "Using Session Templates" on page 356 for more information about these templates.

Report Formats

Using an advanced logging template, you format a report, adding boilerplate text around selected variables if you wish. For example, you might create an entry like the following:

```
With a total of %LiveConnections.Count% player connections, Helix Server is using
%Server.Bandwidth.Output% bits per second of bandwidth.
```

In this example, `%LiveConnections.Count%` and `%Server.Bandwidth.Output%` are variables, and the rest of the text is boilerplate. When Helix Server generates the report, it replaces the variable entries with values from its registry. The resulting report looks like the following example:

```
With a total of 50 player connections, Helix Server is using 2,800,000 bits per
second of bandwidth.
```

Using Session Templates

A session template reports on registry variables that are dynamically added and deleted. Helix Server creates registry variables when encoders, transmitters, and other components connect to it. These variables store information about the component. Using a session template, you can create a report when one of these components connects, disconnects, or both.

> **Tip:** To report statistics for each media player, you use a client statistics template instead.

Choosing a Watch Type

When you create a session template, you select a watch type, which specifies the type of component connection that generates the report. The following table describes the possible values that you can choose.

Session Template Watch Types

Watch Type Value	Registry Values Watched
`Broadcast Receiver [BroadcastReceiver.Statistics]`	splitting receivers
`Broadcast Transmitter [BroadcastDistribution.Statistics]`	splitting transmitters
`Broadcast [LiveConnections]`	live connections
`Broadcast Archiver [LiveArchiving.Archiver]`	live broadcast archiving
`Configuration Change Log [Server.ConfigLog]`	configuration file changes

For example, if you choose `Configuration Change Log [Server.ConfigLog]` as the watch type, you can generate a report every time Helix Administrator updates the Helix Server configuration file. In your report, you then choose which server registry variables you want to log.

> **For More Information:** The section "Logging Server Configuration Changes" on page 364 provides an example of how to log configuration changes made through Helix Administrator.

Selecting the Output Format Type

For each session template, you can choose whether to generate the report when the watched component connects, when it disconnects, or both. When you set up the template, you choose an output format from a pull-down list:

`Session Added Output Format`	Generate a report with the specified variables when the watched component connects.
`Session Deleted Output Format`	Generate a report with the specified variables when the component disconnects.

Defining Output Methods

The advanced logging output methods determine how Helix Server publishes the report. There are several options, and you can select multiple delivery methods for each custom log report. Additionally, multiple report templates can write to the same output, such as the same file. Most outputs require configuration. For example, if you send your report to a file and a local TCP port, you specify a file name and a port number.

Console

The `Std Error` (Standard Error) and `Std Out` (Standard Output) options both publish the report to the command line console. No configuration is required.

File

When you select the `File` output method, Helix Server publishes the report to a text file, continuously appending new results to the end of the file unless you set up log rolling. You configure the following variables:

`File name`	The log file name. The default location is the main Helix Server installation directory. You can specify a relative or absolute path using the syntax appropriate for your operating system.
`Log Rolling Frequency`	How many hours, days, weeks, or months pass before a new log file is created (optional).
`Log Rolling Size`	Maximum size in Megabytes that the log file can become before a new file is created (optional).

Using Log File Rolling

Log files can grow indefinitely as they accumulate data. To keep log files manageable, you can limit them to a specific size, or create a new log at a preset interval, such as every six hours or two weeks, depending on the amount of data you expect to log. Helix Server begins, or *rolls*, a new log file when the limit is reached. Log rolling is optional, but recommended if you expect to report statistics frequently.

> **Tip:** If multiple templates write to the same file log file, define log rolling in just one template.

Log Rolling Methods

Generally, you limit log files by frequency or size. You can select both methods, however, to create log files according to the first limit reached. For example, you can create a new log file whenever the preceding file reaches 10 Megabytes in size, or has recorded 3 days of activity, whichever comes first.

Timestamps

When you implement log rolling, Helix Server appends a timestamp to the end of the file name to indicate when the file was created. Suppose that you specify the file name `serverstats.txt`. Your log directory may contain several files with the same base file name, but each with a unique timestamp that looks like this:

```
serverstats.txt.20030622134953
```

The timestamp is in the format `YYYYMMDDHHMMSS`, using a 24-hour clock. Hence, the file in the preceding example was created on June 22, 2003, at 1:49.53 P.M.

HTTP Post

With the `HTTP Post` method, Helix Server publishes the report to a Common Gateway Interface (CGI) program. You configure the following variables:

`URL`	URL location (excluding `http://`) of the CGI program. For example: `logger.example.com/cgi-bin/report.py`
`Port`	Number of the HTTP port on the Web server receiving the log.

> **Note:** If the attempt to contact the CGI program fails, Helix Server writes the message `Failed to write log data to HTTP POST Socket` to its error log and does not attempt to republish the

report. For more information, refer to *Helix Server and Helix Proxy Troubleshooting Guide*.

TCP Broadcast

The `Outbound TCP` and `Inbound TCP` output destinations let you send the report to an application listening on a specific TCP port. The `Outbound TCP` method publishes the log on a remote computer. For this method, you configure the following variables:

`Destination`	Host name, IPv4 address, or IPv6 address of the computer that receives the log.
`Port`	Number of an open port on the specified computer.

The `Inbound TCP` method publishes the log on the local computer. You configure the following variable:

`Port`	Number of an open port on the local computer.

UDP Broadcast

The `Outbound UDP` and `Multicast UDP` methods publish the report to a UDP socket on a remote computer using unicast or multicast UDP, respectively. You configure the following variables:

`Destination`	Host name, IPv4 address, or IPv6 address of the computer where the report should be published. For `Multicast UDP`, enter a Class D IPv4 multicast address.
`Port`	Number of an open port on the specified computer.

UNIX Pipe and System Log

On UNIX operating systems, the `Pipe` and `Syslog` methods make the log available to another process, or publish the information to the system log, respectively. For `Pipe`, you configure the following variable:

`Command`	Pipe command to the application or script where the information can be post-processed.

For `Syslog`, you choose one the following priorities, each of which corresponds to an entry type in the UNIX system log:

- `LOG_EMERG`

- LOG_ALERT
- LOG_CRIT
- LOG_ERR
- LOG_WARNING
- LOG_NOTICE
- LOG_INFO
- LOG_DEBUG

Windows NT Event Log

If you choose NT Event Log, Helix Server publishes the report to the event log that corresponds to the priority selected. Each option corresponds to an entry type in the Windows NT Event Log:

- LOG_ERR
- LOG_WARNING
- LOG_INFO

Creating Logging Templates

The following procedure explains how to create a new logging template, or modify the predefined templates. You'll need to be familiar with the information in the preceding sections to set up your custom template.

➤ To create or modify an advanced logging template:

1. Click **Logging & Monitoring>Advanced Logging**.
2. To create a new template, choose Interval, Watch, Client Stats, or Session from the **Add Template** pull-down list. The option you choose affects other options that appear on the page, as described in Step 8 through Step 11. To select an existing template, highlight its name in the **Templates** area.

   For More Information: See "Template Types" on page 354.

3. If you are creating a new template, edit the name in the **Template Name** box. This name is for your reference only.

4. The **Template Type** pull-down list indicates the type of template you chose (`Interval`, `Watch`, `Client Stats`, or `Session`). You can change the template type here if needed.

5. From the **Template Status** box, select `On` or `Off` to enable or disable the logging report, respectively. An existing template starts or stops reporting as soon as you change its status and click **Apply**.

6. Optionally, you can enter a description in the **Template Description** box. This is for your own reference only, but is highly recommended.

7. You next select one or more output types for the report to determine where Helix Server sends the report information:

   a. Select an output type from the **Add Output Type** pull-down list.

   b. Optionally, edit the name in the **Output Name** box. This name is for your reference only.

   c. For the selected output type, enter the necessary configuration parameters, as described in "Defining Output Methods" on page 357.

8. If you chose `Interval` in Step 2, follow this step. Otherwise, skip to the next step. For the `Interval` template, set the appropriate combination of hour, minute, and seconds for the report interval in the **Output Interval** boxes. If you leave a box blank, the setting for that box is considered to be `0`. Next, follow the instructions in Step 12.

9. If you chose a `Watch` template in Step 2, follow this step. Otherwise, skip to the next step. For the `Watch` template, click `Property List` in the **Watches** area. In the list that appears, choose the variable or variables that you want to watch for changes.

   The optional minimum and maximum output intervals for the `Watch` template let you generate the report at regular intervals. If you do not define either field, Helix Server creates the report only when a watched registry variable changes:

   - **Minimum Output Interval**

     This field holds a number in the format `HH:MM:SS` that defines the smallest possible time that must pass between log outputs. Changes to the watched variables are not reported until the minimum interval has elapsed. If you set `00:05:00`, for example, watched variables that change are reported every five minutes. If no watched variable changes

its value in five minutes, though, the report is not created until a variable changes, or the maximum interval is reached.

- **Maximum Output Interval**
  This entry contains a number in the format `HH:MM:SS` that defines the longest possible time allowed to pass before the report output is generated. Changes to variables being watched will generate the report output even if the maximum interval has not been reached, however. If you set `01:00:00`, for instance, and no watched variable changes within an hour after the last report, Helix Server generates the report when the hour elapses.

Next, follow the instructions in Step 12.

> **Note:** Place each watched variable on a separate line in the **Watches** list. Helix Server determines which variables to watch by matching the string that appears on each line of the **Watches** list.

10. If you chose `Client Stats` in Step 2, follow this step. Otherwise, skip to the next step. For the `Client Stats` template, set the appropriate combination of hour, minute, and seconds for the report interval in the **Output Interval** boxes. If you leave a box blank, the setting for that box is considered to be `0`. Next, follow the instructions in Step 12.

    > **Note:** The report intervals are relative to the start of the player session. So if the interval is five minutes and player A connects at 12:00 whereas player B connects at 12:01, the first reports for players A and B are generated at 12:05 and 12:06, respectively.

11. If you chose `Session` in Step 2, select the appropriate watch type from the **Watch Type** list box. As described in "Choosing a Watch Type" on page 356, the watch type you select determines which dynamic event triggers the report output.

12. For any template type, click **Property List** in the **Output Format** area to pick the variables from the Helix Server registry included in the template. Note that you can define two report outputs for some template types:

    - If you're setting up a `Session` template, you can specify one output format by selecting **Session Added Output Format**, and a second format by choosing **Session Deleted Output Format**. These output

formats for the beginning and end of the watched session can be identical or different.

- For a `Client Stats` template, you can choose **Periodic Output Format** to specify an output format generated whenever the **Output Interval** time elapses. By selecting **Disconnect Output Format**, you can create another output format that is generated whenever a media player stops playing a presentation.

Do the following to define the output format for any template type:

a. In the property list window, navigate to the variable that you want to include in the template.

b. When you click on a variable, a string identifying that variable appears in the **Output Format** text box. Helix Server reports values for variables in the exact order the variables appear in the text box. To organize the order of variables, cut and paste them in the order that you want them to appear in your report.

> **Tip:** A variable added to the **Output Format** text box is surrounded by percentage signs (`%Server.Bandwidth.Output%`). If you reorganize the order of the variables, be sure to include the percent signs that surround each variable name.
>
> **For More Information:** For descriptions of the registry properties, refer to the registry properties section of *Helix Server Configuration and Registry Reference*.

c. Optionally, format the report output by adding boilerplate text.Two tags help with formatting the output sting. Use a `\n` tag to move output to a new line. Carriage returns you enter in the box are also recognized as new lines. Use a `\t` tag to insert a tab.

13. Click **Apply**.

Sample Templates

This section explains how to use the preconfigured template that comes with Helix Server. It also provides examples of setting additional templates to log client statistics and changes to Helix Administrator.

Using the Server Stats Templates

The preconfigured Sever Stats template is an example of an interval template. It is designed to send basic server statistics to the output console every hour. To use it, you must enable it and, optionally, customize it by changing the output destination or modifying the reporting variables. The report output looks like this:

```
Server Stats (06/17/02 10:33:52)
   Uptime: 1234274 seconds
   CPU Percent Usage: 5
   Players Connected: 32
   Players Connected in the Last 10 Seconds: 2
   Players Connected by Protocol: 22 RTSP, 10 MMS, 0 HTTP (0 Cloaked)
   Players Connected by Transport: 0 TCP, 32 UDP, 0 MCast
   Total Subscribed Bandwidth Output: 9385984 bps
   Total Actual Bandwidth Output: 9244432 bps
   Average Bandwidth Output Per Player: 293312 bps
   Memory Stats: 14294824 Bytes In Use
```

Logging Server Configuration Changes

Using a Session template, you can log changes to Helix Server made through Helix Administrator. Follow the instructions for setting up a Session template as described in "Creating Logging Templates" on page 360, setting any desired output, such as the screen console or a text file. For **Watch Type**, choose Configuration Change Log [Server.ConfigLog]. For **Session Added Output Format**, enter the following to capture all configuration changes:

```
%Server.ConfigLog.*.Entry%
```

The report indicates the IP address and user name of the person who made the change, along with the date, time, and browser version. It then lists the changes made to the Helix Server registry, which are recorded in the configuration file. The following is a sample report entry:

```
127.0.0.1 - FBLACK.AdminRealm/fblack [15/Aug/2003:12:26:28 -0700]
"POST admin/configvar.set.html HTTP/1.0" - - [Mozilla/4.0
(compatible;MSIE 6.0;Windows NT 5.1;.NET CLR 1.0.3705)]
Set config.DiffServ.Control=0 [OK]
Set config.DiffServ.Media=17 [OK]
```

> **Tip:** The IP address, date, time, and browser fields are the same as those used in the basic access log. For more information, see "Access Log Fields" on page 329.

Generating Client Statistics Reports

This example illustrates how to use a `Client Stats` template to log information about each media player request. This sample template generates periodic updates about the session status, such as the current number of lost packets. When the player disconnects, the disconnection report provides information about the entire session, such as the total number of packets lost, the requested URL, the streaming protocol, and so on.

Periodic Client Statistics Report

For **Periodic Output Format**, you define the report information you want to collect for each media player whenever the **Output Interval** time elapses. You create a report using boilerplate text and variables chosen from the pop-up property list. Note that `\n` adds a new line to the report. You can also use `\t` to insert tabs. The following is the sample template definition in Helix Administrator:

```
\n\n****PERIODIC CLIENT STATISTICS****
Date and Time: %Date%, %Hour%:%Min%.%Sec%
Client GUID: %Client.*.GUID%
Average Bit Rate: %Client.*.Session.*.AvgBitrate%
Bytes Sent: %Client.*.Session.*.BytesSent%
Packets Sent: %Client.*.Session.*.PacketsSent%
Packets Lost: %Client.*.Session.*.PacketsLost%
Failed Resends: %Client.*.Session.*.FailedResends%
Successful Resends: %Client.*.Session.*.SuccessfulResends%
```

The following text is an example of a report generated from the preceding template:

```
****PERIODIC CLIENT STATISTICS****
Date and Time: 08/21/03, 15:50.04
Client GUID: 3ab16eb2-9f30-4c1f-acb6-25dfba5ba0da
Average Bit Rate: 225000
Bytes Sent: 1798645
Packets Sent: 588
Packets Lost: 1
Failed Resends: 0
Successful Resends: 1
```

For More Information: For details about how Helix Server determines the client GUID, see "Client Identifier" on page 332.

Disconnection Statistics Report

For **Disconnect Output Format**, you define the report information you want to collect when a client disconnects. Here's the template defined in Helix Administrator:

```
\n\n****CLIENT DISCONNECT****
Date and Time: %Date%, %Hour%:%Min%.%Sec%
**CLIENT INFORMATION**
Client GUID: %Client.*.GUID%
Client ID: %Client.*.ClientID%
Type: %Client.*.User-Agent%
Address: %Client.*.Addr%
Preferred Language: %Client.*.Language%
**CONNECTION STATUS**
Average Bit Rate: %Client.*.Session.*.AvgBitrate%
Bytes Sent: %Client.*.Session.*.BytesSent%
Packets Sent: %Client.*.Session.*.PacketsSent%
Packets Lost: %Client.*.Session.*.PacketsLost%
Failed Resends: %Client.*.Session.*.FailedResends%
Successful Resends: %Client.*.Session.*.SuccessfulResends%
**CLIP INFORMATION**
Requested URL: %Client.*.Session.*.PlayerRequestedURL%
Start Time: %Client.*.StartTime%
Clip Size in Bytes: %Client.*.Session.*.FileSize%
Playing Duration (seconds): %Client.*.Session.*.DurationSeconds%
Title: %Client.*.Session.*.FileHeader.Title%
Author: %Client.*.Session.*.FileHeader.Author%
Copyright: %Client.*.Session.*.FileHeader.Copyright%
Stream Count: %Client.*.Session.*.FileHeader.StreamCount%
**TRANSPORT INFORMATION**
Protocol: %Client.*.Protocol%
Port: %Client.*.Port%
UDP used (0=no, 1=yes): %Client.*.IsUDP%
```

The following is output generated from the preceding template. Helix Server generates this report only when a media player stops playing a presentation:

```
****CLIENT DISCONNECT****
Date and Time: 08/21/03, 15:51.00
**CLIENT INFORMATION**
Client GUID: 3ab16eb2-9f30-4c1f-acb6-25dfba5ba0da
Client ID: WinNT_5.0_6.0.11.818_RealPlayer_RN10PD_en-us_686
Type: RealMedia Player Version 6.0.9.1753 (win32)
Address: 207.188.7.125
Preferred Language: en-us
```

```
**CONNECTION STATUS**
Average Bit Rate: 225000
Bytes Sent: 2478645
Packets Sent: 643
Packets Lost: 1
Failed Resends: 0
Successful Resends: 1
**CLIP INFORMATION**
Requested URL: rtsp://208.147.89.157:554/video1.rm
Start Time: 21/Aug/2003:15:48:50
Clip Size in Bytes: 2479645
Playing Duration (seconds): 130
Title: Introductory Video
Author: RealNetworks, Inc.
Copyright: ©2006 RealNetworks, Inc.
Stream Count: 1
**TRANSPORT INFORMATION**
Protocol: RTSP
Port: 7180
UDP used (0=no, 1=yes): 1
```

For More Information: For descriptions of the client registry properties, refer to the client properties chapter of *Helix Server Configuration and Registry Reference*.

CHAPTER

17

ACTIVITY MONITORS

To manage activity on your Helix Server, you'll want to know which clips are popular, what the stream load is, and whether viewers are being turned away. Helix Server includes a Java-based Server Monitor and, for Windows NT users, an NT Performance Monitor that help make system management easier.

> **For More Information:** To generate reports of historical activity, see Chapter 15, "Basic Logging" or Chapter 16, "Advanced Logging".

Using the Server Monitor

The Server Monitor is a configurable graph that displays real-time information about the number of connected clients, the resources used, and the clips being streamed. It shows who is using Helix Server, when it is most used, and which files are the most requested. Start Server Monitor by clicking **Logging & Monitoring>Server Monitor**.

Server Monitor uses a Helix Server port that you can change, as described in "Defining Communications Ports" on page 63. It uses the password you selected for Helix Administrator during installation. It does not prompt for the password, though, if you are already logged into Helix Administrator.

> **Note:** The Server Monitor password is stored in the `MonitorPassword` variable of the configuration file, and can be changed by modifying the configuration file. For more information about the configuration file, see Appendix A.

> **Tip:** You can also create other external Server Monitors to track more than one Helix Server. A monitoring message displays along the bottom of each window, telling you which Helix Server is being monitored.

Selecting Server Monitor Modes

The Server Monitor can run as an applet or application. Running it as an applet though Helix Administrator is the most common approach. But you may also want to run it as a Java application.

Running Server Monitor as an Applet

When you select **New Window** from the **Options** menu, the new Server Monitor runs as an applet. This mode has the following features and limitations:

- Can be run from inside a Web browser.
- Can be run from any remote machine with a Java-enabled browser.
- Settings may not be saved when you switch among the Helix Administrator's Web pages.
- Developers can use a scripting language and the parameters below to customize the Server Monitor applet to their specifications.

Applet Parameters

Parameter	Possible Values	Default Value
dragZoom	enabled, disabled	enabled
viewPanel	keyPanel, resourcePanel, clientPanel, filePanel, minimized, disabled	keyPanel
StatusBar	enabled, disabled	enabled
PlayerCount	enabled, disabled	enabled
FileCount	enabled, disabled	enabled
EncoderCount	enabled, disabled	enabled
MonitorCount	enabled, disabled	enabled
SplitterCount	enabled, disabled	disabled

Running Server Monitor as an Application

You can also run Server Monitor as an application, which offers the following features and requirements:

- No Web browser needed.
- Can switch among different servers without spawning new windows.
- Java class files, available for free download from Sun, must be installed on the local machine. They are described below.

➤ To run Server Monitor in application mode:

1. Download and install version 1.1 of the Java Development Kit, available from Sun's Web site at **http://java.sun.com/javase/index.jsp**. Follow the installation instructions on the Web site to install the Java Development Kit on your system.
2. In a command prompt, navigate to the directory where the newly installed Java class files are located. Change to the `Bin` subdirectory.
3. At a system prompt, type the following:
   ```
   jre -cp Monitor.jar Monitor
   ```
4. In the logon screen, enter the following items:
   - **Helix Server name**—use the host name, IPv4 address, or IPv6 address of the machine on which Helix Server is installed.
   - **Monitor Port**—you can find this number by clicking **Server Setup> Ports** in Helix Administrator.
   - **Monitor Password**
5. Click **OK**.
6. Server Monitor starts.

Server Monitor Used with Other Features

Server Monitor displays all on-demand and live presentations that are currently being streamed or broadcast. It does not differentiate among the delivery methods—whether streaming, unicasting, splitting, or multicasting.

Server Monitor used with Other Features

Other Feature	Notes
Live Archiving	Server Monitor does not indicate whether live files are being archived.
SLTA	Server Monitor does not distinguish the source of a clip; thus it never shows whether a broadcast is coming from an event in progress or SLTA.

(Table Page 1 of 2)

Server Monitor used with Other Features (continued)

Other Feature	Notes
Splitting	On the transmitter, no connections are displayed in Server Monitor. On the receiver, the split connection will appear under the Connections tab as an encoder.
Multicasting	Server Monitor can show clients that are receiving back-channel multicasts, just as it shows clients receiving any other type of broadcast or stream. However, it will not show the number of clients receiving scalable multicasts.

(Table Page 2 of 2)

Displaying Server Monitor Information

There are several ways you can control what Server Monitor displays. This section describes the commands present on the Server Monitor display area and their functions.

Choosing Display Options

Select the pull-down **Options** menu in the upper-left corner of Server Monitor to configure the Monitor's features, or spawn an external Server Monitor that runs outside of the browser

Options Menu Commands

Command	Action
New Window	Create a new, external monitor. You can then minimize the browser and resize the new monitor.
Pause	Freeze the graph. Server Monitor continues to receive data, but the graphical display of data does not change. Click **Resume** from **Options** to resume the graphing.
Reset	Clear the graph and reset all peak data.
Configure	Display the configuration screen. Here, you can specify the update frequency in seconds and the time scale for server activity in minutes. You can also select which statistics to monitor.
Autofit	Rescale the graph so that it fits within the viewable area. Note that whenever you zoom, the autofit feature is disabled. Select **AutoFit** again to re-enable the feature.

(Table Page 1 of 2)

Options Menu Commands (continued)

Command	Action
Zoom In	Zoom in on the graph. Use the mouse to select a range over the graph to zoom in for a closer view. Hold down the **CTRL** key on your keyboard, and click the mouse to Autofit the graph.
Zoom Out	Zoom out from the graph.

(Table Page 2 of 2)

Monitoring Activity

The **Key**, **Performance**, **Connections**, and **Files** tabs each have a specific focus, providing you with an overall picture of server performance. Clicking the active tab expands or collapse the tab information and show only the tab name, leaving more room for the monitor.

Key Tab

Through the **Key** tab, you can control how Helix Server information is graphed. By clicking different options in the **Line** column, you can control what colors and line widths are used to display Helix Server information

Key Tab Columns

Column	Purpose
Line	Controls the order, width, and color of monitoring lines in the graph. Click the left-most up arrow to move a row up. Click the line graphic to change the line width, and click the arrows at the right to choose one of 16 possible colors for the line.
Name	Indicates the item being monitored: players, monitors, encoders, files, and receivers.
Current	Shows the number of the current connections.
Peak	Shows the peak number of files monitored, along with the time and date.

Performance Tab

The **Performance** tab provides statistics on Helix Server performance

Performance Tab Columns

Column	Purpose
CPU Usage	Shows current CPU usage.
Memory Usage	Indicates Helix Server memory usage in Kilobytes.

(Table Page 1 of 2)

Performance Tab Columns (continued)

Column	Purpose
Bandwidth	Displays the amount of data being sent in Kilobits per second.
Players Connected	Lists the numbers of clients connected.
File Usage	Lists the number of files being served.

(Table Page 2 of 2)

Connections Tab

This tab provides background on connected clients and the files they are accessing.

Connections Tab Columns

Column	Purpose
IP Address	Gives the client software's IPv4 or IPv6 address.
Type	Indicates the type of browser or client software.
Duration	Shows the amount of time the client has been connected.
Filename	Provides the name of the file being served.

Files Tab

The files tab provides statistics on all files being served.

File Tab Column

Column	Purpose
Filename	Provides the name of the file being served.
Current	Indicates the number of clients currently connected.
Total	Shows how many times the file was served during this monitoring session.
Peak	Shows the peak number of files monitored, along with time and date.

Windows Performance Monitor

Helix Server is designed to work with the Windows Performance Monitor to show activity on one or more Helix Servers. This option is available if you are running Helix Server on Windows, and are viewing it from that same computer. A Performance Monitor file containing the Helix Server statistics, `rmserver.msc`, is supplied.

You can also configure the Performance Monitor to show Helix Server status from any computer on your network. The Performance Monitor can show the types of information listed in the following table.

Windows Performance Monitor Information

Information	Indicates
Clients and protocol	The number of active clients. Also shown are the protocols used by the clients to receive streams.
Connection type	The number and type of connections, whether TCP or UDP.
Multicast connections	The number of active multicast connections.
Total bandwidth	The number of bits per second being consumed.
Percent of processor	How much processor time Helix Server is using.
Connections	How many encoders, monitors, and receivers are connected.
Incoming bandwidth	Bandwidth of streams arriving from encoders.
Files playing	Number of files playing, including all the files in a SMIL presentation. Live files are also shown.
Files archiving	Number of live files being saved.

Using the Performance Monitor, you can display any combination of this information in any of the following formats:

- A chart that graphs activity over time.
- Alerts that notify the administrator through e-mail, or run programs based on criteria.
- Log files that list activity on Helix Server.
- Reports based on activity information.

> **For More Information:** For information on configuring these formats, see the online help in the Performance Monitor.

CHAPTER

18

SNMP

Using Simple Network Monitoring Protocol (SNMP), you can monitor Helix Server from an SNMP management system. This allows you to change Helix Server configuration from a third-party tool, and send notice of important events to an external program. This chapter explains how to set up the SNMP monitoring plug-in and the Helix Server master agent.

Understanding SNMP

The following sections describe the components of the Helix Server SNMP monitoring system. Before implementing SNMP on Helix Server, be sure that you understand the basics of SNMP monitoring and know how to operate your chosen SNMP management system.

SNMP Plug-in

Helix Server includes an SNMP plug-in that monitors its registry for configuration values and events. The plug-in communicates to the master agent using a proprietary protocol. It can send important information about Helix Server operation to the master agent, and update the Helix Server configuration as instructed by the management system. You must configure the plug-in before it can operate.

> **For More Information:** The section "Configuring the SNMP Plug-In" on page 379 explains how to set up the plug-in. Refer to "License File Information" on page 56 for more about licensed features.

Master Agent

The SNMP plug-in communicates with the master agent, an executable program included with Helix Server. The master agent then communicates

with the management system using the SNMP protocol. The SNMP plug-in and the management system never communicate directly. The master agent can run as an independent application or a Windows service. Once configured, the master agent generally runs without the need for user intervention.

For More Information: See "Configuring the Master Agent" on page 381.

SNMP Protocol

The master agent uses the SNMP protocol to communicate with the management system. It supports SNMP version 1 (SNMPv1), version 2c (SNMPv2c), and version 3 (SNMPv3). Versions 1 and 2 of the SNMP protocol do not encrypt messages between the two components, and are therefore recommended only when both Helix Server and the management system reside behind a firewall on the same private network.

Note: Helix Server does not support SNMP over Internet Protocol version 6 (IPv6). Components must use IPv4 addresses.

SNMP Version 3 Protocols

SNMPv3 is suitable for communications over an unprotected network. The User-based Security Model (USM) for SNMPv3 defines two authentication protocols, both of which are supported for Helix Server SNMP:

- HMAC-MD5-96

  This protocol is based on MD5. Operations using MD5 occur faster than those using SHA.

- HMAC-SHA-96

  This protocol is based on SHA-1. SHA provides a stronger security mechanism than MD5.

SNMP Version 3 Security Levels

SNMPv3 defines three levels of security. The lowest level (`noAuthNoPriv`) does not provide authentication or privacy, and is comparable to SNMP version 1.

The second level (AuthNoPriv) provides authentication, but no privacy. The third level (AuthPriv) provides authentication and encryption for all messages.

SMMP Versions and Authentication Modes

SNMP Version	Authentication Mode	Operation
version 1 (SNMPv1)	noAuthNoPriv	Authentication is performed by matching an unencrypted community string. This method is not suitable for communication across an unsecured network.
version 2 (SNMPv2c)	noAuthNoPriv	
version 3 (SNMPv3)	noAuthNoPriv	
	authNoPriv	This mode provides authentication based on the HMAC-MD5 or HMAC-SHA algorithm.
	authPriv	This mode provides authentication based on the HMAC-MD5 or HMAC-SHA algorithm. It also uses encryption based on the CBC-DES (DES-56) standard.

Management System and Management Information Base (MIB)

You can use any third-party SNMP monitoring tool as your management system. The management information base (MIB) determines the Helix Server configuration variables that the management system monitors and controls. It also defines the event traps that the SNMP plug-in can report to the master agent. Helix Server ships with a MIB configuration file named helixserver.my, located in the main Helix Server installation directory.

> **For More Information:** The section "Running a Management System" on page 388 explains the monitoring trees that appear in the management system.

Configuring the SNMP Plug-In

You configure the SNMP plug-in through Helix Administrator. The configuration connects the plug-in to the master agent, and defines which events to report to the management system.

➤ To configure the SNMP plug-in:

1. Click **Logging & Monitoring>SNMP**.
2. To use SNMP monitoring, set the value of **Enable SNMP** to Yes.

3. In the **Master Agent Hostname or IP Address**, enter the DNS name or IPv4 address used by the master agent program. The agent typically resides on the Helix Server machine in the Helix Server `Bin` directory, so you can use the localhost address of `127.0.0.1`.

4. For **Master Agent Port**, enter the master agent port used for the AgentX protocol. The default is port `705`.

   Note: This value must match the value for the `AgentXProtocolPort` variable entered in the master agent configuration file. See "Configuring the Master Agent" on page 381 for more information.

5. For **Send SNMP Traps**, select `Yes` if you want the SNMP plug-in to notify the management system when specific events occur, such as CPU utilization above a certain percentage. If you leave the pull-down list set to `No`, the SNMP plug-in responds to management system requests, but does not report Helix Server events. In this case, you can ignore the remaining fields, which define the event values to trap, and click **Apply** to save your changes.

   Tip: You can disable an individual trap by setting its value to `0`.

6. The **Trap Interval** field determines how frequently a trap is triggered while the trap condition remains active. The value is in seconds, with 20 as the default. For example, if you trap CPU utilization, the trap triggers every 20 seconds as long as the CPU stays above the specified level of use.

7. The **Trap Server Startups** pull-down determines if the SNMP plug-in notifies the management system of Helix Server restarts. Change the value to `No` if you do not want to set this trap.

8. In the **Trap CPU Utilization Above** field, enter an integer in the range from 0 to 100 that indicates the percentage of total machine CPU utilization that triggers a trap. A value of `75`, for example, initiates a trap when CPU utilization on the Helix Server machine reaches 75 percent or higher.

   Tip: CPU utilization is reported by the operating system. On multiprocessor machines, this figure represents the aggregate load for all processors.

9. For **Trap Connection Counts Above** field, enter an integer that indicates a number of simultaneous media player connections that triggers a trap. A

value of `1000`, for example, initiates a trap when 1,000 or more media streams are being delivered.

10. The three fields for **Memory Usage Watermarks** allow you to set up to three traps reported when Helix Server memory usage rises above a specified amount. Use an integer value that represents Kilobytes. If you set `128000`, `180000`, and `230000`, for example, the SNMP plug-in reports memory usage that exceeds approximately 128, 180, and 230 Megabytes, respectively.

    > **Tip:** Set the traps at 50, 70, and 90 percent, respectively, of memory allotted to Helix Server. The values shown above represent these percentages on a machine in which Helix Server has a dedicated allotment of 256 Megabytes.

11. For **Bandwidth Usage Watermarks**, you can define up to three traps that trigger when the Helix Server outgoing bandwidth use rises above the amount specified in Kilobits per second (Kbps). Each field accepts an integer value. If you set `2000`, `4000`, and `8000`, for example, the SNMP plug-in reports when outgoing bandwidth exceeds approximately 2, 4, and 8 Megabits per second, respectively.
12. Click **Apply**.

Configuring the Master Agent

The master agent is the intermediary through which the SNMP plug-in and the management system communicate. It must always run on the Helix Server machine. The following sections explain how to modify the master agent configuration file to define your system addresses, users, and security model.

Modifying the Master Agent Configuration File

You use the `master.cfg` file installed in the Helix Server installation directory to configure the master agent. This allows the master agent to communicate with the SNMP plug-in and the management system. It also defines the security level for each person who uses the management system. Edit this XML-formatted text file using any text, HTML, or XML editor. The following example shows the default configuration file:

```
<?xml version="1.0" encoding="US-ASCII"?>
<preferences version="0.5">
   <config ManagerAddress="127.0.0.1" ManagerSNMPPort="162"
      LocalSNMPPort="161" AgentXProtocolPort="705" EngineID="XXX"/>
   <security CommunityString="public"/>
   <SecurityModel ModelType="USM">
      <users UserName="xxx">
         <Authentication Type="MD5" Password="yyy"/>
         <Privacy Type="DES" Password="zzz"/>
      </users>
      <users UserName="unsecureUser">
         <Authentication Type="NONE" Password=""/>
         <Privacy Type="NONE" Password=""/>
      </users>
   </SecurityModel>

   <SecurityToGroup SecurityModel="USM" User="unsecureUser" Group="v3Group"/>
   <SecurityToGroup SecurityModel="USM" User="test" Group="testGroup"/>
   <SecurityToGroup SecurityModel="v2" User="vishal" Group="v1v2group"/>
   <SecurityToGroup SecurityModel="v1" User="public" Group="v1v2group"/>

   <SecurityModel ModelType="VACM">
      <groups Name="v3Group" SecurityModel="USM" SecurityLevel="1"
         Context="" Notify_View="testView" Read_View="testView"
         Write_View="testView"/>
      <groups Name="testGroup" SecurityModel="USM" SecurityLevel="1"
          Context="" Notify_View="testView" Read_View="testView"
          Write_View="testView"/>
      <groups Name="v1v2group" SecurityModel="v1" SecurityLevel="1"
           Context="" Notify_View="v1NotifyView" Read_View="v1ReadView"
           Write_View="v1WriteView"/>
      <groups Name="v1v2group" SecurityModel="v2" SecurityLevel="1"
            Context="" Notify_View="v1NotifyView" Read_View="v1ReadView"
            Write_View="v1WriteView"/>

      <views Name="testView" OID="1.3" Mask="" Included="1"/>
      <views Name="v1ReadView" OID="1.3" Mask="" Included="1"/>
      <views Name="v1WriteView" OID="1.3" Mask="" Included="1"/>
      <views Name="v1NotifyView" OID="1.3" Mask="" Included="1"/>

   </SecurityModel>
</preferences>
```

Defining Master Agent Addresses and Ports

The following lines in the master agent configuration define the basic communication between the master agent, the SNMP plug-in, and the management system:

```
<config ManagerAddress="127.0.0.1" ManagerSNMPPort="162"
   LocalSNMPPort="161" AgentXProtocolPort="705" EngineID="XXX"/>
```

The following table explains the values you should set for these attributes.

Master Agent Address and Port Attributes

Attribute	Value
ManagerAddress	The IPv4 address of the management system. The master agent uses this address to send traps to the management system. You must specify an IP address, not a DNS name. The master agent does not support IPv6 addresses.
ManagerSNMPPort	The port used by the management system to listen for communication from the master agent. The default is port 162.
LocalSNMPPort	The local port used by the master agent for SNMP communications. The default is port 161. If another application uses an SNMP process, port 161 may be in use. In this case, specify a free port. Do not use port 162, as this can cause a system slowdown.
AgentXProtocolPort	The master agent port for AgentX, which is the protocol used to communicate with the Helix Server SNMP plug-in. The default is 705.

Setting Up SNMP Security

The following lines set the parameters for SNMP security. The USM security model defines the access rights for each person running the management system. The configuration file predefines two users. The first user operates with no security, which is equivalent to using SNMP version 1. The second user defines authentication and privacy, the highest security under SNMPv3. You can modify or delete these predefined users, as well as create additional users by adding new <users>…</users> lists within the USM section:

```
<security CommunityString="public"/>
<SecurityModel ModelType="USM">
   <users UserName="xxx">
      <Authentication Type="MD5" Password="yyy"/>
```

```
      <Privacy Type="DES" Password="zzz"/>
   </users>
   <users UserName="unsecureUser">
      <Authentication Type="NONE" Password=""/>
      <Privacy Type="NONE" Password=""/>
   </users>
</SecurityModel>
```

The following table defines the master agent configuration attributes that define the SNMP security level and permissions.

Master Agent Address and Port Attributes

Attribute	Value
CommunityString	Password used with SNMP version 1 or 2. You can ignore the <security/> tag if you are using SNMP version 3. If you are using version 1 or 2, you can ignore the USM settings.
UserName	Name of the user as defined in the management system.
Authentication Type	Type of authentication used with SNMPv3. Valid values are MD5 for the HMAC-MD5 algorithm, or SHA for the HMAC-SHA algorithm. A value of NONE indicates an unsecured user.
Privacy Type	For privacy type, you can enter NONE for no privacy or DES for CBC-DES encryption.
Password	Password for authentication or privacy. SNMPv3 uses separate passwords for authentication and privacy. You do not need to define a certain password, however, if you used NONE as the authentication or privacy type.

Defining a View Access Control Model

The view access control model (VACM) available through SNMPv3 allows you to define precisely which Helix Server SNMP objects each viewer can see and control. VACM is optional, and you should be familiar with how it works within your SNMP management system before you define view privileges through the master agent configuration file. The following sections provide an example of how to set up view access for a specific user.

Assigning a User to a Group

Each person who uses VACM must be defined in the <SecurityModel> list as an SNMPv3 user. The following example shows a user defined to use SNMPv3 with authentication but no privacy:

```
<SecurityModel ModelType="USM">
   <users UserName="Maria">
      <Authentication Type="MD5" Password="tl73jkIL98"/>
      <Privacy Type="NONE" Password=" "/>
   </users>
   ...other users defined here...
</SecurityModel>
```

Using a <SecurityToGroup/> tag, you assign each user to a group name that you create. In the following example, the user Maria is assigned to a group named v3Group:

```
<SecurityToGroup SecurityModel="USM" User="Maria" Group="v3Group"/>
```

The following table explains the <SecurityToGroup/> tag attributes.

VACM <SecurityToGroup/> Tag Attributes

Attribute	Value
SecurityModel	Security model for this user. Choose one of the following: – v1 for SNMP version 1 – v2c for SNMP version 2c – USM for SNMP version 3
User	The user's name.
Group	The group to which the user is assigned. Groups are defined with <groups/> tags.

Creating Groups

Within the <SecurityModel> list, a <groups/> tag defines each group. A group has three views, indicating which parts of Helix Server the user can monitor and control. In the following example, v3Group is assigned the fullView view for receiving traps and reading Helix Server variables. It is part of the noView view for writing configuration changes to the Helix Server registry:

```
<SecurityModel ModelType="VACM">
   <groups Name="v3Group" SecurityModel="USM" SecurityLevel="1"
      Context="" Notify_View="fullView" Read_View="fullView"
      Write_View="noView"/>
   ...more groups and views defined here...
</SecurityModel>
```

The following table explains the <groups/> tag attributes.

VACM <groups/> Tag Attributes

Attribute	Value
Name	The group name. Users are assigned to this group by including the name in the <SecurityToGroup/> tag.
SecurityModel	Security model for this group. Choose one of the following: – v1 for SNMP version 1 – v2c for SNMP version 2c – USM for SNMP version 3
SecurityLevel	Security level for this group. Choose one of the following: – 0 for noAuthNoPriv – 1 for authNoPriv – 2 for authPriv
Context	An optional, named subset of object instances in the management information base.
Notify_View	The name of the view assigned to the group for receiving traps.
Read_View	The name of the view assigned to the group for reading SNMP objects values corresponding to Helix Server registry values.
Write_View	The name of the view assigned to the group for writing changes to SNMP object values and thereby changing Helix Server configuration values.

Defining Views

Within the <SecurityModel> list, a <views/> tag defines each view. The view identifies a group of objects by an OID from the management information base (MIB). All objects that fall under that OID are included in the view. In the following example, the fullView view is included while the noView view is excluded, allowing no access:

```
<SecurityModel ModelType="VACM">
   ...groups defined here...
   <views Name="fullView" OID="1.3" Mask="" Included="1"/>
   <views Name="noView" OID="1.3" Mask="" Included="0"/>
   ...more views defined here...
</SecurityModel>
```

The following table explains the <views/> tag attributes.

VACM <views/> Tag Attributes

Attribute	Value
Name	The view name. Groups are assigned up to three views (Notify_View, Read_View, and Write_View) in each <groups/> tag.
OID	Object ID of a node. All objects that fall under that node in a tree are available in the view. The MIB file used by the SNMP management system lists the OIDs of all nodes.
Mask	Optional mask value that applies to the OID. You can use this mask to provide finer control over the objects available in the view.
Included	A true or false value that includes or excludes the view. Use 1 to make the view available, 0 to turn exclude the view from use.

Running the Master Agent on Windows

On Windows, you can run the master agent as a service or as an application. The following sections explain how to start the master agent in either mode.

Restarting the Master Agent Service

If you installed the master agent as a Windows Service as described in "Installing Helix Server" on page 44, the agent starts up automatically. If you change the master agent configuration, restart the agent service by locating the master agent service with **Settings>Control Panel>Administrative Tools>Services**. In the **Services** dialog, right-click on **SNMP Master Agent** and choose **Restart**.

> **Tip:** Right-click **SNMP Master Agent** and choose **Properties** to change the master agent operation. Using this dialog, for example, you can disable automatic start-up or restart the service automatically if it fails.

Starting the Master Agent as an Application

The following procedure explains how to start the master agent as a Windows application. Do this only if the master agent has not been installed as a service, or you have disabled the service through the **Services** dialog.

➤ Starting the master agent as a Windows program:

1. Open a command prompt using **Start>Program>Accessories>Command Prompt**.

2. Navigate to the Helix Server installation directory. For example:

   ```
   cd "C:\Program Files\Real\Helix Server"
   ```

3. The master agent uses the executable name `master.exe` and resides in the `Bin` subdirectory. Start it by entering the following:

   ```
   Bin\master.exe master.cfg
   ```

4. Start Helix Server as described in "Starting Helix Server" on page 47.

Starting the Master Agent on UNIX

The following procedure explains how to start the master agent as a UNIX background process.

➤ To start the master agent on UNIX:

1. If Helix Server is running, shut it down as described in "Stopping Helix Server" on page 50.

2. Log in as `root`.

3. From the command line, navigate to the Helix Server installation directory. For example:

   ```
   # cd /usr/local/Real/HelixServer
   ```

4. The master agent uses the executable name `master` and resides in the `Bin` subdirectory. Start it as a background process:

   ```
   # ./Bin/master master.cfg &
   ```

5. Start Helix Server as described in "Starting Helix Server" on page 47.

Running a Management System

Once you have the SNMP plug-in and master agent configured and running, you can use your management system to monitor and control Helix Server. From the management system, locate the MIB file, which is named `helixserver.my` and resides in the Helix Server installation directory. Compile the MIB file if necessary for your management system. You then connect the management system to the master agent using the Helix Server IP address and port defined in the master agent configuration file.

> **For More Information:** For information about compiling the MIB file and connecting your management system to the master agent, refer to your SNMP manager documentation.

The section "Defining Master Agent Addresses and Ports" on page 383 explains the master agent port usage.

Monitor Tree

The MIB file produces three main trees in the SNMP management system. Through these monitoring trees, you can monitor Helix Server operation and control the settings of certain variables. The `hsMonitor` tree contains objects related to Helix Server monitoring. These objects, described in the following table, cannot be changed by the management system.

Monitor Tree Objects

Object	Information
`hsClients`	Number of media players currently connected. Covers all supported players and communications protocols.
`hsRTSPClients`	Number of media players currently communicating through the RTSP control protocol.
`hsHTTPClients`	Number of media players currently communicating through the HTTP control protocol.
`hsUDPTransports`	Number of media players currently using the User Datagram Protocol (UDP).
`hsTCPTransports`	Number of media players currently using the Transmission Control Protocol (TCP).
`hsMulticastTransports`	Number of media players connected on multicast.
`hsBandwidthUsage`	Total outgoing bandwidth in Kilobits per second (Kbps) used by Helix Server.
`hsPercentCPUUsage`	Percentage of CPU used by Helix Server processes.
`hsMemoryUsage`	Amount of memory used by Helix Server in bytes.
`hsPlatform`	Helix Server operating system.
`hsVersion`	Helix Server software version.
`hsAccumulatedBandwidth`	Network bandwidth use in Kilobits per second
`hsEncoderCount`	Number of media encoders currently delivering live streams.
`hsUptime`	Time since last Helix Server restart in seconds.
`hsMMSClientCount`	Number of media players currently communicating through the MMS control protocol.

Configuration Tree

The hsConfig tree contains objects that map to Helix Server configuration variables, such as Helix Server ports and the various traps that can be set. You can monitor these objects using any version of SNMP.

Configuration Tree Objects

Object	Information
hsPorts	This subtree allows you to change the main Helix Server communications ports. For more information on these values, refer to "Defining Communications Ports" on page 63.
hsTrap	The objects in this subtree correspond to the event traps defined for the SNMP plug-in. For more information, refer to "Configuring the SNMP Plug-In" on page 379.

Tip: A configuration change occurs immediately without the need to restart Helix Server. However, to make the change permanent, you need to write the change to the configuration file using the Control Tree.

Control Tree

The hsControl tree contains objects that you can use to control certain Helix Server operations. By setting hsRestartServer to true, for example, you can restart Helix Server remotely.

Control Tree Objects

Object	Information
hsWriteConfigFile	Writes configuration changes made by the management system to the configuration file when set to true or 1. This ensures that the configuration changes remain in effect after a restart. For more on the configuration file, refer to Appendix A.
hsStopServer	Stops Helix Server when set to true or 1. For information on shutdown options, refer to "Defining a Delayed Shutdown" on page 71.
hsRestartServer	Restarts Helix Server when set to true or 1.

PART

VII

APPENDIXES

The following appendixes contain useful reference information.

APPENDIX

A

CONFIGURATION FILE

When you start Helix Server, it reads a configuration file to gather system settings. When you change Helix Server configuration information, Helix Administrator updates the configuration file automatically. This appendix provides general information about the configuration file.

> **For More Information:** For details about configuration lists and variables, see the *Helix Server Configuration and Registry Reference*.

Understanding the Configuration File

The configuration file holds the Helix Server information in a series of XML-formatted lists and variables. The default file is `rmserver.cfg`, but you can specify an alternate file at startup, as described in "Starting Helix Server" on page 47. The alternate file might be one that you have manually edited, using `rmserver.cfg` as a starting point.

The Helix Server installation directory contains a backup copy of the configuration file named `default.cfg`. This is a mirror image of the default `rmserver.cfg` file that was created during installation. You can restore your configuration file from the backup if you make changes that you want to undo, or if you accidentally delete the main copy.

> **Note:** Be sure to store the configuration file where only authorized users can make changes to it. The default location is the main Helix Server's installation directory.

> **Tip:** If you have multiple servers, you may want to name each configuration file differently to identify which server you're working with.

Editing the Configuration File

You can change Helix Server settings by editing the configuration file with any text editor or XML editor. Some third-party plug-ins may require that you add parameters and variables manually to the configuration file, for example. The configuration file's tags are based on XML (eXtensible Markup Language), and the file is organized into sections for clarity. There are four types of tags in the file:

1. XML declaration tag
2. optional comment tags
3. list tags
4. variable tags

Of these four types, only lists and variables make up the instructions to Helix Server. All values for lists and variables are enclosed in double quotation marks.

> **Tip:** When you edit the configuration file manually, be sure to use correct syntax. Helix Server looks for exact spellings and correct use of angle brackets. Helix Server does not display messages related to syntax errors. Instead, it ignores any settings that it does not recognize.

> **Note:** Because Helix Administrator reflects the settings of the configuration file in use, exit Helix Administrator before opening the configuration file with a text editor.

XML Declaration Tag

The XML declaration tag indicates which version of XML is in use. Helix Server Version 11.1 uses XML version 1.0. The declaration tag looks like this:

```
<?XML Version="1.0" ?>
```

Comment Tags

Optional comment tags are used in the configuration file to identify tag functions. Identical to comment tags in HTML, they begin with <!-- and end with -->. For example, the following comment tag lets the administrator know that the parameters after it refer to the path settings:

```
<!-- P A T H S -->
```

To disable a feature, convert the feature's tag or tags to a comment. Rather than converting each tag to a comment, you can place the comment's begin tag in front of the feature's first opening tag, and the comment's end tag after the feature's closing tag:

```
<!-- The following feature is commented out
  ...feature lists and tags here...
-->
```

> **Warning!** Do not nest comment tags within other comment tags.

List Tags

Lists are used for instructions that have several parts, such as the MIME types or the multicast instructions. A list tag is followed by one or more list tags or variable tags. The list tag uses the following syntax:

```
<List Name="name">
...
</List>
```

Here, *name* is the list title. Using the correct capitalization for *name* is important.

Other lists or variables follow the list. The `</List>` tag signifies the end of the list. If other lists are inside the original list, they must also have closing `</List>` tags. The `MIMETypes` list is an example of a list that contains other lists.

> **Tip:** Indenting list items is not required, but is recommended for clarity.

Variable Tags

Variable tags use the following syntax:

```
<Var name="value"/>
```

Here, *name* is the variable name, and *value* is a string or a number, depending on the variable. Capitalization for both *name* and *value* is important. Unlike lists, variables do not have a closing tag; instead, a forward slash mark (/) appears before the closing angle bracket (>).

Variables can be independent elements (such as `LogPath`), or they may appear inside a list. When variables appear within a list, their meaning is determined

by the value of the list name, even though they may appear identical in syntax to variables that are not inside lists. If there are multiple variables within a list that do similar things, their names must be unique. For example, the `Extension` variables within each `MIMETypes` list must have different names. This is accomplished by adding a number to the end of each, such as `Extension_01`, `Extension_02`, and so on.

> **Tip:** If you've restarted Helix Server and it's not responding to a change you've made to a variable, make sure the variable has a closing forward slash mark, and that there is no space between them.

Helix Server Restart

You typically need to restart Helix Server, as described in "Restarting Helix Server" on page 53, after you modify the configuration file manually. If you change the Helix Server file manually on a UNIX computer, you can use `SIGHUP` to upload the changes to Helix Server without breaking any open connections, as long as the changes do not require a full server restart.

To have Helix Server re-read the configuration file, use the following `SIGHUP` command:

```
kill -HUP processID
```

in which *`processID`* is the Helix Server process number, as shown in the `Logs/rmserver.pid` file. For more on this, see "Process ID (PID)" on page 50.

> **Note:** When you issue the `SIGHUP` command, Helix Server closes current log files and opens the log files specified in the updated configuration. If log file names have changed, Helix Server creates the new logs under the new names.

> **Tip:** Mount point changes typically require a full restart. Helix Administrator indicates when configuration changes require a full restart. Use it as your guide to changes that you can and cannot upload with `SIGHUP`.

APPENDIX

B

ADDRESS SPACE BIT MASKS

In a number of Helix Server features, you can identify a range of IPv4 addresses by assigning a bit mask to an IP address. Helix Server interprets the bit mask as a single, contiguous block of address spaces. This appendix describes how to create a bit mask for the purpose of identifying a range of IP addresses.

Understanding Basic IP Address Construction

To understand how bit masks work, it is helpful to review the basic concepts for constructing an IP address. Each IP address is 32 bits, divided into four 8-bit octets. Each bit in an octet is assigned a value between 128 and 1, from left to right. To indicate whether a value is in use, the bit is set to 1. The sum of all bit values for each octet determines the octet's dotted decimal value. The following defines values for each bit in an octet:

	Bit 1	**Bit 2**	**Bit 3**	**Bit 4**	**Bit 5**	**Bit 6**	**Bit 7**	**Bit 8**
Bit Value:	128	64	32	16	8	4	2	1

It is possible to make any number between 0 and 255 simply by indicating whether each bit in an octet is set to 1 or 0. For example, both of the following expressions indicate the same IP address:

dotted decimal:	192.0.1.2
32-bit binary equivalent:	11000000 00000000 00000001 00000010

Using a Bit Mask to Identify an Address Space

To indicate a range of IP addresses, you must first identify the *lowest* IP address in your range, and then indicate the number of bits that are identical between that address and the highest IP address in the range. Consider the range of IP addresses 192.0.1.255 to 192.0.1.0.

These two addresses indicate a range of 256 possible address (in practice only 254, because the all-zero and all-one addresses are reserved). Between the two indicated addresses, `192.0.1.0` is the lowest in the range, and the first three octets (the first 24 bits) are exactly the same for both addresses. Consider the following addresses and the bit mask expressed in binary.

Addresses and Bit Mask

Address and Mask	Dotted Decimal	32-Bit Binary Equivalent
Highest Address	`192.0.1.255`	`11000000 00000000 00000001 11111111`
Lowest Address	`192.0.1.0`	`11000000 00000000 00000001 00000000`
Bit Mask	`24 Bits`	`11111111 11111111 11111111 00000000`

Notice that the first 24 bits in the highest and lowest addresses are exactly the same. The same would be true if you had used an address with any decimal number (0-255) in the last octet. The bit mask uses 1's to indicate bits to be evaluated, and 0's to indicate bits to be masked. Thus, assigning a bit mask of `255.255.255.0` to the lowest IP address in the range indicates an address space of 256 possible IP addresses.

Slash Notation

In the preceding table, the bit mask appears in both its dotted decimal and 32-Bit binary form. However, this same address space can also be articulated with *slash notation* like this:

`192.0.1.0/24`

Slash notation uses the lowest IP address in the range, followed by a slash and a number that indicates how many bits should be evaluated. This is helpful to understand because in Helix Server you indicate a bit mask in a similar manner. You select the number of bits—from 0 bits through 32 bits—from a pull-down list.

Address Space Size

The size of the address space is determined by the number of bits included in the bit mask. The fewer bits used, the more addresses that are included in the address space. An 8-bit mask includes 2^{24} power addresses, while a 24-bit mask includes only 2^{8} power addresses.

Bit Boundaries

Bit boundaries also affect which address can be included in an address space. To understand how bit boundaries work, recall that each octet includes 8 bits, and that each bit has an assigned value. Ranges correspond to the value of each bit in an octet. Further, these ranges cannot cross bit boundaries. Consider the following addresses:

Bit Boundary with an Inappropriate Mask

Address and Mask	Dotted Decimal	32-Bit Binary Equivalent
Highest Address	192.0.0.2	11000000 00000000 00000000 00000010
Lowest Address	192.0.0.1	11000000 00000000 00000000 00000001
Bit Mask	31 Bits	11111111 11111111 11111111 11111110

Although there are only two consecutive addresses in the range shown in the preceding table, you cannot create a range of two with these addresses, because bit 31 is different for each address. This is a bit boundary. To create a range for these addresses, use the sixth bit in the fourth octet, or a bit mask of 30. Note, though, that by using 30 bits, you also end up including more addresses:

Bit Boundary with an Appropriate Mask

Address and Mask	Dotted Decimal	32-Bit Binary Equivalent
Highest Address	192.0.1.3	11000000 00000000 00000001 00000011
	192.0.1.2	11000000 00000000 00000001 00000010
	192.0.1.1	11000000 00000000 00000001 00000001
Lowest Address	192.0.1.0	11000000 00000000 00000001 00000000
Bit Mask	30 Bits	11111111 11111111 11111111 11111100

Determining Bit Boundaries

The chart below identifies every literal bit range available. Look up the bit for the octet you are working with in the **Bit** column on the left. Then use the corresponding **Literal Bit Range** column to look up the decimal values available for each range.

For example, the problem described above arose from attempting to use bit 7 in the fourth octet (Bit 31). However, in row 7 of the table below, no range includes decimal 1 through 2. For a range that works, you need to use the

sixth bit in the fourth octet (Bit 30). Notice that in row 6, there is a decimal range that includes 1 through 2 (range 0-3).

Literal Bit Ranges

Bit	Literal Bit Ranges
1	Ranges of 0-127; or 128-255
2	Ranges of 0-63; 64-127; 128-191; or 192-255
3	Ranges of 0-31; 32-63; 64-95; 96-127; 128-159; 160-191; 192-223; 224-255
4	Ranges of 0-15; 16-31; 32-47; 48-63; 64-79; 80-95; 96-111; 112-127; 128-143; 144-159; 160-175; 176-191; 192-207; 208-223; 224-239; 240-255
5	Ranges of 0-7; 8-15; 16-23; 24-31; 32-39; 40-47; 48-55; 56-63; 64-71; 72-79; 80-87; 88-95; 96-103; 104-111; 112-119; 120-127; 128-135; 136-143; 144-151; 152-159; 160-167; 168-175; 176-183; 184-191; 192-199; 200-207; 208-215; 216-223; 224-231; 232-239; 240-247; 248-255
6	Ranges of 0-3; 4-7; 8-11; 12-15; 16-19; 20-23; 24-27; 28-31; 32-35; 36-39; 40-43; 44-47; 48-51; 52-55; 56-59; 60-63; 64-67; 68-71; 72-75; 76-79; 80-83; 84-87; 88-91; 92-95; 96-99; 100-103; 104-107; 108-111; 112-115; 116-119; 120-123; 124-127; 128-131; 132-135; 136-139; 140-143; 144-147; 148-151; 152-155; 156-159; 160-163; 164-167; 168-171; 172-175; 176-179; 180-183; 184-187; 188-191; 192-195; 196-199; 200-203; 204-207; 208-211; 212-215; 216-219; 220-223; 224-227; 228-231; 232-235; 236-239; 240-243; 244-247; 248-251; 252-255
7	Ranges of 0-1; 2-3; 4-5; 6-7; 8-9; 10-11; 12-13; 14-15; 16-17; 18-19; 20-21; 22-23; 24-25; 26-27; 28-29; 30-31; 32-33; 34-35; 36-37; 38-39; 40-41; 42-43; 44-45; 46-47; 48-49; 50-51; 52-53; 54-55; 56-57; 58-59; 60-61; 62-63; 64-65; 66-67; 68-69; 70-71; 72-73; 74-75; 76-77; 78-79; 80-81; 82-83; 84-85; 86-87; 88-89; 90-91; 92-93; 94-95; 96- 97; 98-99; 100-101; 102-103; 104-105; 106-107; 108-109; 110-111; 112-113; 114-115; 116-117; 118-119; 120-121; 122-123; 124-125; 126-127; 128-129; 130-131; 132-133; 134-135; 136-137; 138-139; 140-141; 142-143; 144-145; 146-147; 148-149; 150-151; 152-153; 154-155; 156-157; 158-159; 160-161; 162-163; 164-165; 166-167; 168-169; 170-171; 172-173; 174-175; 176-177; 178-179; 180-181; 182-183; 184-185; 186-187; 188-189; 190-191; 192-193; 194-195; 196-197; 198-199 200-201; 202-203; 204-205; 206-207; 208-209; 210-211; 212-213; 214-215; 216-217; 218-219; 220-221; 222-223; 224-225; 226-227; 228-229; 230-231; 232-233; 234-235; 236-237; 238-239; 240-241; 242-243; 244-245; 246-247; 248-249; 250-251; 252-253; 254-255
8	Only an exact match is possible.

Working with 0-Bit and 32-Bit Masks

There are two masks that create somewhat special cases: 0-Bits and 32-Bits. When an IP address has a 32-Bit mask, it creates a literal range of 1. For example, consider the following address and 32-Bit mask:

```
192.0.1.1 /32
```

When Helix Server evaluates incoming IP addresses against this IP address, there is only one possible match: `192.0.1.1`. For a match, the incoming address must match all 32-Bits in the original address.

Just as there is only one possible match for addresses with a 32-Bit mask, the opposite is true for addresses with a 0-Bit mask. An IP address with a 0-bit mask essentially tells Helix Server to match any addresses. Although not required, you should also enter an origin address of all zeros, like so:

```
0.0.0.0 /0
```

This works because Helix Server uses a Boolean *and* operation to evaluate incoming addresses. In this type of algorithm, anything and zero equals zero, so all incoming addresses end up equal to the all-zero address entered as the origin address.

APPENDIX

C

AUTHENTICATION DATA STORAGE

After Helix Server has authenticated user access, it can check whether the user has permission to view specific clips or directories. You can use this information for applications such as pay-per-view. Permission information is stored in a separate database, and this appendix describes the data storage methods that you can use with the authentication feature.

Understanding Authentication Data

To authenticate visitors, the Helix Server stores user IDs and passwords or client IDs, and their associated access permission information. When a client tries to access a clip, the Helix Server looks up this information to see whether the client or visitor is authorized to view the clip. The information can be stored in either a series of text files or in a database. Templates for common databases are installed during installation.

This section describes the methods for storing user name and password data. Templates for common databases are created during installation, that correspond to the database types listed in "Supported Database Types" on page 282.

- **Text file storage**—this default method uses a combination of directory structure and text files to achieve a sensible data storage method. See "Using Text Files" on page 404 for details.
- **Database templates**—the supplied templates use a structure similar to the text file method, in a more familiar database format. Refer to "Using a Database" on page 408 for more information.

Using Text Files

The default configuration uses the text file storage method to provide storage for all default realms. The following directories contain the text files which store data. The center letter indicates the authentication protocol: r is for RealSystem 5.0, b is for Basic.

Supplied Data Storage Directories

Directory Name	Information Stored
enc_b_db	encoder authentication for current versions of RealProducer
enc_r_db	encoder authentication for RealProducer G2 through 8.5
enc_w_db	encoder authentication for Windows Media Encoder
adm_b_db	Helix Administrator user authentication
con_r_db	content authentication

The following table describes the contents of these directories.

Text File Storage Directory Structure

Directory	Contents	File or Directory Description
Main directory (con_r_db, enc_b_db, enc_r_db, enc_w_db, or adm_b_db)	ppvbasic.txt	The text file indicates to Helix Server that this is the storage area for the list of authenticated names.
users	(initially blank)	Files in this directory list the clips and permission types.
guids	(initially blank)	For player validation, files connect the clientID with a user name.
logs	reglog.txt accesslog.txt	See below for a description of these files.
redirect	(initially blank)	For player validation, files contain an URL to which to send the client if redirection is necessary.

The actual data storage text files do not exist when Helix Server is first installed. They are created when authentication is in use, and secure content is first requested. When Helix Server creates the file structure, it creates the ppvbasic.txt file. Helix Server looks for this file the second and subsequent times you start it. If the file does not exist, it recreates the directory structure.

Warning! Do not delete the ppvbasic.txt file! If you delete the ppvbasic.txt file, Helix Server will rewrite the directories and erase their prior content.

Users Directory

The files in this directory are named *username*, where *username* is the user name. This directory contains one file per registered user. The first line of each file has the following format and is different than subsequent lines in the file:

password;uuid;uuid_writeable

Field	Description
password	When user authentication is in use, this stores the password. Otherwise shows an asterisk (*). Passwords are encrypted. See "Using the Password Tool" on page 281.
uuid	In player validation, stores playerID. In user authentication, an asterisk (*) appears in this field.
uuid_writeable	A flag set and used by Helix Server: 0–playerID is in database 1–record created, but playerID is not yet registered

The second and subsequent lines of each file have the following format. For further detail on allowable values in each field, see table "Permission Types" on page 292:

url;url_type;permission_type;expires;debitted_time

Field	Description
url	URL of secure directory or clip.
url_type	Whether URL is directory or clip: 0–clip 1–directory
permission_type	Permission type associated with access.
expires	If *permission_type* is 1, this is the expiration date/time, in format MM/DD/YYYY:HH:MM:SS. Otherwise blank.
debitted_time	If *permission_type* is 2, this is time remaining in seconds. If *permission_type* is 3, this is the number of seconds of material the visitor has viewed. Otherwise, it is blank.

The example file, user1, has the following content, when player validation is in use:

```
*;00001d00-0901-11d1-8b06-00a024406d59;0
Secure/clip1.rm;0;0;*;*
Secure/directory;1;0;*;*
Secure/time.rm;0;2;*;300;*
Secure/time.rm;0;1;05/24/1970:06:12:32;300;*
```

> **Note:** If you manually edit the files, be sure that any blank or unused fields use an asterisk (*) as a placeholder. Do not use a space for a placeholder.

Guids Directory

The files in this directory are given the names of the unique client IDs from the registered clients, one per registered user. Each file contains only the name of the associated user name. For example, a file such as 00001d00-0901-11d1-8b06-00a024406d59 contains the name of the user, user1.

Logs Directory

This directory contains two files: reglog.txt and accesslog.txt.

reglog.txt

Each line of reglog.txt represents the result of an attempt to register a visitor. This file has the following format:

status;userid;uuid;IP;register_time;url_redirect

Field	Description
status	Result of user's attempt to connect: 0–Success 1–Failed (clientID not readable) 2–Failed (clientID already used) 3–Failed (RealAudio Player 3 or earlier) 4–No user (Must be entered previously in the database) 5–General failure
userid	Unique name of up to 50 characters.
uuid	clientID.
IP	IP address from which user is attempting to connect.

(Table Page 1 of 2)

Field	Description
request_time	Time of connection request.
url_redirect	If connection failed, URL to which user was redirected (see redirect.txt).

(Table Page 2 of 2)

accesslog.txt

Each line of accesslog.txt describes the result of an attempt to view a clip. This file is not created until authentication is enabled and the first user attempts to connect:

status;userid;uuid;ip;url;access_type;permission_on;start_time;end_time;total_time;why_disconnect

Field	Description
status	Result of user's attempt to connect: 0–access to clip granted 1–denied
userid	Unique name of up to 50 characters.
uuid	Stores player ID.
ip	IP address from which user is attempting to connect.
url	Secured clip user is attempted to access.
access_type	Permission type associated with access. See the table "Permission Types" on page 292 for values.
permission_on	Permission type associated with URL: 0–file (individual clip) 1–directory 2–none
start_time	Time/date clip started playing.
end_time	Time/date clip stopped playing.
total_time	Total time clip played.
why_disconnect	Reason for disconnection: 0–client disconnected voluntarily 1–server access expired

Redirect Directory

Used only in player validation, the `redirect` directory contains files named after URLs that are restricted from unauthorized users. Within each file is the alternate URL to which Helix Server sends the user if he or she tries to click the restricted URL. If no files are present in this directory, and the user attempts to click a URL to which he or she has not been given access, the user receives an error message.

Because certain characters that appear in URLs are illegal in file names, Helix Server requires a substitution for these illegal symbols.

Substitutions

Character	Replacement Sequence
/	+2f
\	+2b
+	+5c

For example, the URL `Secure/TopSecret.rm` would be converted to `Secure+2fTopSecret.rm`. The URL within each file, however, is represented normally.

Using a Database

This section describes the structure of the database templates included with Helix Server. To set up the database, see "Setting Up Other Types of Data Storage" on page 411. The database templates include five tables:

- **Users table**—Together with the permissions table, contains the lists of who is registered and with what access.
- **Permissions table**—Linked to the users table, lists specific clips and directories and the permissions associated.
- **Register_log table**—Used if player validation is in use, it tracks the clientID.
- **Redirect table**—Used in player validation only.
- **Access_log table**—Used by the commerce feature.

Users Table

Gives the list of user names and passwords.

Users Table

Field	Description
userid	User name of up to 50 characters. Ties to permissions table.
password	In user authentication, this stores the password. Otherwise blank. Passwords are encrypted.
uuid	In player validation, stores clientID. In user authentication, an asterisk (*) appears in this field.
uuid_writeable	A flag set and used by Helix Server: 0–clientID is in the database 1–the record has been created but the clientID is not yet registered with Helix Server.

Permissions Table

Linked to the users table through the userid, this identifies the specific clips or directories and the type of access for each.

Permissions Table

Field	Description
userid	User name of up to 50 characters. Ties to Users table.
url	URL of secure directory or clip.
url_type	Whether URL is directory or clip: 0 clip 1 directory.
permission_type	Permission type associated with access.
expires	Permission expiration date and time, in format MM/DD/YYYY:HH:MM:SS. Used only if *permission_type* is 1 (dated). Otherwise blank.
debitted_time	If permission_type = 2 (countdown), this is the number of seconds remaining. If permission_type=3 (countup), this is the number of seconds of material the visitor has viewed. Otherwise, it is blank.

Register_Log Table

The register_log table is used only if player validation is in use (indicated by `UseGUIDValidation=True`).

Register_log Table

Field	Description
status	Result of user's attempt to connect: 0–Success 1–Failed (clientID not readable) 2–Failed (clientID already used) 3–Failed (RealAudio Player 3 or earlier) 4–No user (Must be entered previously in the database) 5–General failure
userid	Unique name of up to 50 characters.
uuid	Stores clientID.
ip	IP address from which user is attempting to connect.
request_time	Time of connection request.
url_redirect	If connection failed, URL to which user was redirected (see Redirect Table, above).

Redirect Table

The redirect table is only used in player validation.

Redirect Table

Field	Description
url	URL of any secure clip or directory.
url_redirect	URL to which users could be redirected to if they are not allowed access to that clip. New URL must **not** be a secure URL.

Access_Log Table

Used by the commerce feature to show which secure content has been accessed.

Access_Log Table

Field	Description
status	Result of user's attempt to connect: 0–access to clip granted 1–denied
userid	Unique name of up to 50 characters.
uuid	Stores player ID.
ip	IP address from which user is attempting to connect.
url	Secured clip user is attempted to access.
permission_type	Permission type associated with access. See the table "Permission Types" on page 292 for values.
permission_on	Permission type associated with url: 0–file (individual clip) 1–directory 2–none
start_time	Time/date clip started playing.
end_time	Time/date clip stopped playing.
total_time	Total time clip played.
why_disconnect	Reason for disconnection: 0–client disconnected voluntarily 1–server access expired

Setting Up Other Types of Data Storage

Support for two types of databases is included.

➤ To set up your Windows computer for ODBC compliance:

1. On the **Start** menu, point to **Settings**, and click **Control Panel**.
2. Double-click **32bit ODBC.**
3. On the **System DSN tab**, click **Add**.
4. Select your ODBC driver from the list of drivers and click **Finish**.

5. In the **ODBC SQL Server Setup** dialog box, type the data source name. Click **Select**.
6. Type or browse for the path to your database file and click **OK**.
7. Click **OK** to exit the ODBC Data Source Administrator.

You must now tell Helix Server where to find your database.

➤ To set up the supplied database application on UNIX:

1. At a command line, start the database by typing the following:
   `./msql2d &`
2. Create the database by typing the following:
   `./msqladmin create` *`databasename`*
3. Note that whatever you type for *`databasename`* will need to match the database cited in the Databases list.
4. Create the tables using the database text file by typing the following:
   `.msql -h` *`localhost databasename`* `< ppvdemo.db`
   Be sure to include the less-than sign (<).

GLOSSARY

A

access control
A Helix Server feature that allows or denies connections based on the requesting client's IP address.

alias
A replacement string that allows you to shorten the URL used to request a streaming media clip or broadcast.

ASX file
A text file that uses the file extension `.asx`. It launches Windows Media Player and gives it the URL to a streaming clip or presentation.

ASXgen
A Helix Server feature that launches Windows Media Player through an HTTP link in a Web page, eliminating the need to use an ASX file.

authentication
A Helix Server feature that allows or denies media requests based on the viewer's user name (or global ID) and password.

B

back-channel
A control connection to Helix Server that a media player maintains during a multicast. The connection allows the player to send commands such as **Stop**, as well as report quality of service.

bandwidth
The upper limit on the amount of data, typically expressed as Kilobits per second (Kbps), that can pass through a network connection.

binary tag
An XML tag that comprises opening and closing tags, such as `<List>` and `</List>`.

bit
The smallest unit of measure of data in a computer. A bit has a binary value, either `0` or `1`.

bit mask
A code that indicates a range of IPv4 addresses or IPv6 addresses.

bit rate
A measure of bandwidth, expressed as the number of bits transmitted per second. A 28.8 Kbps modem, for example, can transmit or receive around 29,000 bits per second.

broadcast
To deliver a presentation, whether live or prerecorded, in which all viewers join the presentation in progress. Broadcast streams can be delivered by unicast or multicast. They can also employ splitting. Contrast to *on-demand*.

buffering
The receiving and storing of data before it is played back. A clip's initial buffering is called *preroll*. After this preroll, excessive buffering may stall the presentation.

byte
A common measurement of data. One byte consists of 8 bits.

C

cable modems
Devices that allow rapid transmission and reception of data over television cable. They are digital devices, unlike dial-up modems, which transmit analog data.

cache
1. To store a local copy of a clip that resides on a different server.
2. The pool of stored, local clips.

CBR
Constant Bit Rate. A type of RealVideo encoding in which all parts of the video play back at the same bit rate. Contrast to *VBR*.

client
A software application that receives data from a server. A Web browser is a client of a Web server. RealPlayer is a client of Helix Server.

clip
A media file within a presentation. Clips typically have an internal timeline, as with RealAudio and RealVideo. Other clip types, such as RealText and SMIL, have a timeline set through markup.

codec
Coder/decoder. Codecs convert data between uncompressed and compressed formats, reducing the bandwidth a clip consumes. All audio and video files encoded in a streaming format are compressed using a codec.

commerce rule
Rules that determine whether authentication is required for certain on-demand clips or live broadcasts.

D

download
To send a file over a network with a nonstreaming protocol such as HTTP. Contrast to *stream*.

DSL
Digital Subscriber Line. A technology for transmitting digital data over a regular telephone line much faster than through dial-up modems.

duress stream
A low-bandwidth SureStream audio or video stream that Helix Server uses if a connection's available bandwidth drops greatly.

E

encoder
Software that generates a live or simulated live stream. For RealMedia, an encoder can be RealProducer or the simulated live transfer agent (SLTA).

encoding
Converting a file into a compressed, streaming format. For example, you can encode `.wav` files as RealAudio clips.

F

Flash
A software application and an animation format created by Macromedia. RealPlayer can play Flash animations and stream them in parallel with other clips, such as RealAudio clips.

firewall
A hardware device or software program that monitors and controls connections between computers and the Internet.

Flash Player file
A compressed Flash file format (file extension `.swf`) suitable for streaming. To stream Flash, you export the Flash Player file and tune it so that it plays well in RealPlayer.

H

Helix Administrator
The browser-based application that you use to configure and run Helix Server.

Helix Proxy
RealNetworks software used to stream multimedia presentations to media players. The proxy re-serves streams the originate from Helix Server.

Helix Server
RealNetworks server software used to stream multimedia presentations to media players.

hint track
Information encoded into clips of many formats, including QuickTime and MPEG, that enables Helix Server to stream the clips.

HTTP
Hypertext Transport Protocol. The protocol used by Web servers to communicate with Web browsers. In contrast, Helix Server streams clips to RealPlayer with RTSP.

I

IETF
Internet Engineering Task Force. A standards body that proposes and ratifies Internet standards, including protocols such as RTSP and RTP. The IETF maintains a Web site at **http://www.ietf.org/**.

IPv4 address
An address expressed in dotted decimal form (as in `123.45.123.45`) that identifies a computer on a TCP/IP network.

IPv6 address
The next generation of IP addresses expressed in colon-delimited, hexadecimal form (as in `1080:0:0:0:8:800:200C:417A`) that identifies a computer on a TCP/IP network.

ISDN
Integrated Services Digital Network. Technology that makes digital data connections at 64 or 112 Kbps possible over telephone lines.

ISP
Internet Service Provider. A company that provides access to the Internet. Many ISPs have Helix Server available to stream media clips.

K

kilobit (Kb)
A common unit of data measurement equal to 1024 bits. A kilobit is usually referred to in the context of bit rate per unit of time, such as Kilobits per second (Kbps).

kilobyte (KB)
A common unit of data measurement equal to 1024 bytes or 8 Kilobits.

L

LAN
Local Area Network. A computer network confined to a local area, such as a single building. LANs vary in speed, with bandwidth shared among all networked devices.

lossy
A compression scheme that lowers clip size by discarding nonessential data from the source file. Both RealAudio and RealVideo are lossy.

M

master agent
An executable program that acts as an intermediary between an SNMP management system and the Helix Server SNMP plug-in. It can run as an application, a Windows service, or a UNIX background process.

metafile
Another name for a, ASX file, Ram file, or SDP file.

MMS
A proprietary control protocol used for streaming Windows Media clips and broadcasts to Windows Media Player.

mount point
A portion of a streaming media URL that looks like a directory listing (as in `/ramgen/`) but invokes a Helix Server feature (such as Ramgen).

MPEG
A set of standards-based audio and video compression schemes that includes MPEG-4.

multicast
Delivering a broadcast so that all media players connect to a single stream instead of receiving a separate stream from the server. Helix Server supports *back-channel* and *scalable* multicasts. Contrast to *unicast*.

O

on-demand
A type of streaming in which a clip plays from start to finish when a user clicks a link. Most clips are streamed this way. Contrast to *broadcast*.

P

permissions
Authorizations within the authentication feature that attach to commerce rules to govern which users can view which protected clips or broadcasts.

PNA
A discontinued, proprietary protocol that earlier versions of Helix Server used for backward compatibility with RealPlayer 3 through 5.

port
A connection to a server, designated by a number such as 8080. Helix Server uses different ports for the RTSP, HTTP, and MMS protocols.

preroll
Buffering that occurs just before a clip plays back. Preroll should be no more than 15 seconds.

proxy
A software component that contacts Helix Server to fulfill media requests for media players located behind a firewall. The RealNetworks proxy is Helix Proxy.

pull splitting
In pull splitting, a receiver initiates the splitting session by contacting a transmitter and requesting the stream.

push splitting
In push splitting, a transmitter initiates the splitting session by delivering the broadcast stream to one or more receivers.

Q

QuickTime
A video file format developed by Apple Computer, Inc. and streamed by Helix Server. A QuickTime clip can use a variety of codecs, such as the proprietary Sorenson codec or the standards-based MP3 codec.

R

Ram file
A text file that uses the file extension `.ram` or `.rpm`. It launches RealPlayer and gives it the URL to a streaming clip or presentation.

Ramgen
A Helix Server feature that launches RealPlayer through an HTTP link in a Web page, eliminating the need to use a Ram file.

RDF
Resource Description Framework. A mechanism that media players can use to describe their streaming capabilities to Helix Server. An XML-based RDF file uses the file extension `.rdf`.

RDT
RealNetworks Data Transport. The proprietary data package Helix Server uses (along with RTSP) when communicating with RealPlayer. Contrast to *RTP*.

RealAudio
A clip type for streaming audio over a network. RealAudio clips use the `.rm` extension.

realm
Used with the authentication feature, the realm indicates the database that stores a user's name and password.

RealPlayer
The RealNetworks desktop media player that combines streaming and digital download technologies.

RealPix
A clip type (file extension `.rp`) for streaming still images over a network. RealPix uses a markup language for creating special effects such as fades and zooms.

RealProducer
The primary RealNetworks tool for encoding RealAudio and RealVideo clips.

RealText
A clip type (file extension `.rt`) for streaming text over a network. It uses a markup language for formatting text.

real-time
Delivered as it occurs. For example, a live event is streamed across a network in a real-time broadcast.

RealVideo
A clip type for streaming video over a network. RealVideo clips use the extension `.rm`.

rebuffering
An undesirable state in which a media player must pause a presentation to wait for streaming data to arrive. Rebuffering can result from network conditions, or a poorly produced presentation.

receiver
A Helix Server that receives a stream from a transmitter and broadcasts it to media players.

relay
A Helix Server that functions as both a receiver and a transmitter. It receives a stream from another source, and transmits that stream to another receiver.

RTCP
Real-Time Control Protocol. A control protocol used for monitoring and control of RTP sessions.

RTP
Real-Time Transport Protocol. The open, standards-based data package Helix Server uses (along with RTSP) to communicate with RTP-based clients. Contrast to *RDT*.

RTSP
Real-Time Streaming Protocol. An open, standards-based control protocol that Helix Server uses to stream clips to RealPlayer or any RTP-based client. Contrast to *HTTP*.

S

SDP file
A text file that uses the file extension `.sdp`. It provides media players (typically those playing the MPEG format) with information about the clip or broadcast.

server
1. A software application, such as a Web server or Helix Server, that sends requested data over a network.

2. A computer that runs server software.

Session Announcement Protocol (SAP)
A standard Internet protocol used to publicize broadcasts that are multicast.

Shockwave Flash
See *Flash Player file*.

simulated live broadcast
Delivering a pre-recorded clip (or series of clips) using SLTA so that the presentation appears to be a live broadcast.

SLTA
The Simulated Live Transfer Agent, a Helix Server utility used to deliver pre-recorded clips as a simulated live broadcast.

SMIL
Synchronized Multimedia Integration Language. A markup language for specifying how and when each clip plays within a presentation. SMIL files use the extension `.smil`.

SNMP
Simple Network Monitoring Protocol. A standard protocol that allows one application to monitor activity on another application.

splitting
Sending a live or simulated live stream from a transmitter to a receiver.

stream
1. To send a media clip over a network so that it begins playing back as quickly as possible.
2. A flow of a single type of data, measured in Kilobits per second (Kbps). A RealVideo clip's soundtrack is one stream, for example.

SureStream
A RealNetworks technology that enables a RealAudio or RealVideo clip to stream at multiple bit rates.

T

TCP
Transmission Control Protocol. An Internet transport protocol that provides a bi-directional channel, allowing Helix Server and media players to communicate with each other. Contrast to *UDP*.

transmitter
A Helix Server on which a stream originates. The transmitter sends the stream to a receiver, and can also broadcast the stream directly to media players.

TurboPlay
A RealPlayer feature that minimizes the amount of initial buffering when a clip begins to play.

U

UDP
User Datagram Protocol. An Internet transport protocol that allows Helix Server to send data to media players more efficiently than when using TCP.

unary tag
An XML tag that includes a closing slash, as in `<Var.../>`.

unicast
Delivering a separate broadcast stream to each media player. This is the default broadcasting method. Contrast to *multicast*.

URL
Uniform Resource Locator. A location description that enables a Web browser or RealPlayer to receive a clip stored on a Web server or Helix Server.

V

VBR
Variable Bit Rate. A type of RealVideo or RealAudio encoding that enables RealPlayer to play different parts of the video at different bit rates, even though the video is streamed at a constant rate. Contrast to *CBR*.

W

W3C
World Wide Web Consortium. A standards body that proposes and ratifies Internet software standards, including markup languages such as SMIL. The W3C maintains a Web site at **http://www.w3.org/**.

Windows Media
A proprietary audio and video format developed by Microsoft, Inc. Helix Server streams the Windows Media format to Windows Media Player.

X

XML
Extensible Markup Language. The Helix Server configuration file is based on XML, which allows one to develop flexible, standardized languages for any purpose.

INDEX

X