



## **HELIX PROXY CONFIGURATION AND REGISTRY REFERENCE**

Helix™ Proxy Version 11.1

Revision Date: 20 September 2007

RealNetworks, Inc.  
2601 Elliott Avenue, Suite 1000  
Seattle, WA 98121  
U.S.A.

<http://www.real.com>  
<http://www.realnexus.com>

©2003-2007 RealNetworks, Inc. All rights reserved.

Information in this document is subject to change without notice. No part of this document may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without the express written permission of RealNetworks, Inc.

Printed in the United States of America.

Helix, the Helix Logo, Real, the Real "bubble" (logo), RealJukebox, RealOne, Real-rTV, RealArcade, RealAudio, RealDownload, RealNetworks, RealPix, RealPlayer, RealPresenter, RealProducer, RealProducer Plus, RealPoducer Pro, RealProxy, RealPublisher, RealSites, RealSystem, RealText, RealVideo, Rhapsody, SureStream, The Future is Real, TurboPlay, and Xing are trademarks or registered trademarks of RealNetworks, Inc.

Other product and corporate names may be trademarks or registered trademarks of their respective companies.

# CONTENTS

## PART I: OVERVIEW

INTRODUCTION	1
What is Helix? .....	1
Audience for this Guide .....	1
How This Manual Is Organized .....	1
Conventions Used in This Manual .....	3
Terminology.....	3
Typographical Conventions.....	3
Default Locations and Values.....	4
Additional RealNetworks Resources .....	4
1 NEW FEATURES	7
New Features in Helix Proxy Version 11.1 .....	7
StreamerCount Variable.....	7
SNMP Support.....	7
Reduced Media Start-Up Delay .....	7
Delayed Shutdown .....	8
Client Statistics Type 4 as Default.....	8
Bandwidth Detection for RealPlayer 11 .....	8
IPv6 Support .....	8
TCP Preference for Media Players .....	9
On-The-Fly Packetization of Non-Hinted MPEG Content.....	9
Capabilities Exchange .....	9
Rate Control .....	9
Support for Differentiated Services .....	9
Configurable RTSP Timeout Value.....	9
Logging Enhancements .....	10
Removed Features in Helix Proxy Version 11.1 .....	10
Progressive Networks Audio (PNA).....	10
Legacy Bandwidth Negotiation.....	10
Support for MPEG-1 and Vivo Video Formats .....	10
Upgrade Issues .....	11
Compatibility with Previous Versions .....	11
ProcessorCount Variable Obsolete .....	11

- Access Logging Templates ..... 11
- Media Player Session Templates..... 11
- Binding Syntax for IPv6 Machines..... 12
- 2 OVERVIEW ..... 13
  - Methods for Configuring Features ..... 13
  - Editing the Configuration File ..... 13
    - Helix Administrator ..... 14
    - XML-Based Syntax..... 14
    - XML Declaration Tag..... 15
    - Comment Tags ..... 15
    - List Tags ..... 15
    - Variable Tags ..... 16
  - File System Section ..... 17
  - Applying Configuration Changes..... 17
    - Using the SIGHUP Command on UNIX ..... 17

PART II: CONFIGURATION FILE SYNTAX

- 3 GENERAL CONFIGURATION ..... 21
  - Paths ..... 21
    - Var: PluginDirectory ..... 21
    - Var: SupportPluginDirectory ..... 22
    - Var: LicenseDirectory..... 22
    - Var: PidPath..... 22
    - Var: LogPath ..... 23
    - Var: ErrorLogPath ..... 23
  - Ports ..... 23
    - Ports Syntax ..... 23
    - Var: RTSPPort ..... 23
    - Var: HTTPPort ..... 24
    - Var: MMSPort..... 24
    - Var: MonitorPort..... 25
    - Var: AdminPort ..... 25
  - Monitor Password..... 25
  - IP Binding..... 26
    - IP Binding Syntax ..... 26
    - List: IPBindings ..... 27
    - Var: Address\_n..... 27
    - Var: UseIPv4Only ..... 29
  - Connection Control ..... 29
    - Connection Control Syntax..... 30
    - List: Proxy ..... 30

Var: MaxProxyConnections .....	30
Var: MaxProxyBandwidth .....	31
Var: MaxGatewayBandwidth .....	31
Var: KeepAliveInterval .....	31
UNIX Group and User Names .....	32
Var: Group .....	32
Var: User .....	33
Delayed Shutdown .....	33
List: ServerShutdown .....	34
Var: ClientDisconnectInterval .....	34
Var: ShutdownProceedTime .....	34
Var: NewClientConnectionsAllowed .....	35
Var: LogPlayerTermination .....	35
Automatic Bandwidth Detection .....	36
Bandwidth Detection Syntax .....	36
List: AutoBWDetection .....	36
Var: SendBWPKets .....	37
Helix Administrator .....	37
Helix Administrator Syntax .....	38
List: RealSystem Administrator Files .....	38
List: RealSystem Administrator HTML .....	39
List: RealSystem Administrator DOCS .....	39
List: RealSystem Administrator IMAGES .....	39
List: RealSystem Administrator MONITOR .....	40
<b>4 BASIC STREAMING FEATURES .....</b>	<b>41</b>
On-Demand Streaming .....	41
List: FSMount .....	41
Var: ShortName .....	42
Var: Mountpoint .....	42
Var: BasePath .....	43
HTTP Delivery .....	43
HTTP Delivery Syntax .....	43
List: HTTPDeliverable .....	43
Var: Path_n .....	44
Streamer Count .....	44
Transport Buffers .....	45
Var: TCPSendBufferSize .....	45
Var: UDPSendBufferSize .....	46
Protocol Options .....	46
Protocol Options Syntax .....	46
List: Protocols .....	46

- List: RTSP ..... 47
- Var: PreferClientTCP..... 47
- Caching ..... 48
  - Caching Syntax ..... 48
    - List: RNCache Local File System ..... 49
    - Var: ShortName ..... 49
    - Var: MountPoint ..... 49
    - Var: BasePath ..... 49
  - List: RNCache ..... 50
  - Var: Enabled ..... 50
  - Var: MaxCacheSizeMB..... 50
  - Var: CacheMountPoint..... 51
- Automatic Media Player Redirection..... 51
  - Media Player Redirection Syntax..... 51
    - List: RTSPRedirector..... 52
    - Var: Port..... 52
    - Var: RedirectToAddress ..... 52
    - Var: RedirectToPort..... 53
- Proxy Routing..... 53
  - Proxy Routing Syntax..... 54
    - List: ProxyRoutingTable ..... 54
    - List: Rule Number ..... 54
    - Var: Description ..... 55
    - Var: Rule..... 55
    - Var: UseParentProxy ..... 55
    - Var: ParentMEIPort ..... 56
    - Var: ParentRTSPPort ..... 56
    - Var: ParentName..... 57
- Redundant Proxies ..... 57
  - Redundant Proxy Syntax ..... 57
    - List: ProxyAlternates ..... 58
    - List: Alternates ..... 58
    - List: Alternate Name..... 58
    - Var: Host ..... 59
    - Var: Port..... 59
- Datatype Packetization ..... 60
  - Datatypes Packetization Syntax ..... 60
    - List: Datatypes ..... 60
    - Var: DefaultMaxPacketSize ..... 61
    - List: MIME\_Types..... 61
    - Var: ForcePacketization ..... 62
- Differentiated Services ..... 62

	Differentiated Services Syntax.....	63
	Differentiated Services Decimal Values.....	63
	List: DiffServ.....	64
	Var: Control.....	64
	Var: Media.....	64
5	RATE CONTROL.....	67
	Capabilities Exchange.....	67
	Capabilities Exchange Syntax.....	67
	List: CapExProfiles.....	68
	Var: ShortName.....	68
	Var: MountPoint.....	68
	Var: BasePath.....	69
	List: CapabilityExchange.....	69
	Var: ProfileCacheSize.....	69
	Var: MaxProfileSize.....	70
	List: DefaultProfiles.....	70
	List: Client_n.....	70
	Var: UserAgent.....	71
	Var: URI.....	71
	Var: IgnoreCapExHeaders.....	72
	Rate Control.....	72
	Rate Control Syntax.....	72
	List: MediaDelivery.....	74
	Var: ConfigHelixRAEnabled.....	75
	List: UserAgentProfiles.....	75
	List: MediaPlayerName.....	75
	Var: UserAgent.....	76
	Var: UseMediaDeliveryPipeline.....	76
	Var: InactivityTimeout.....	77
	List: Transport.....	77
	Var: RtcpRRratio.....	78
	Var: RtcpRSratio.....	78
	List: CongestionControl.....	79
	Var: Type.....	79
	Var: MaxBurst.....	80
	Var: PassThrough.....	80
	Var: InitialRate.....	81
	Var: ChannelRate.....	81
	Var: MaxRateScalar.....	82
	Var: UseClientRateReq.....	82
	List: Timeout.....	83

Var: InitialTimeout .....	83
Var: Interval .....	84
Var: MaxTimeouts .....	84
Var: Disable .....	84
List: TFRC .....	85
Var: RTTUseIIR .....	86
Var: TimeoutTransmission .....	86
Var: TimeoutScalar .....	87
Var: UseSlowStart .....	87
Var: StartRate .....	88
Var: SlowStartScalar .....	88
Var: Trace .....	89
List: BCC .....	91
Var: RTTUseIIR .....	91
Var: IncreaseCoefficient .....	92
Var: DecreaseCoefficient .....	92
Var: IncreaseExponent .....	93
Var: DecreaseExponent .....	93
Var: IncreaseThreshold .....	93
Var: DecreaseThreshold .....	94
Var: LowerLimit .....	94
Var: TargetRatio .....	94
Var: TimeoutTransmission .....	95
Var: Trace .....	95
List: TCP .....	97
Var: LimitRate .....	97
List: Session .....	98
List: RateManager .....	98
Var: BufferModel .....	98
Var: Type .....	99
Var: MultiStreamVerifier .....	99
Var: ClientBufferLowWatermark .....	99
Var: ClientBufferHighWatermark .....	100
Var: TargetTimeLowWatermark .....	101
Var: TargetTimeHighWatermark .....	101
Var: ResendTimeLimit .....	102
Var: ClientBufferPeriod .....	103
Var: Trace .....	103
List: InputSource .....	107
Var: InitialMediaRate .....	107
Var: MinPreroll .....	107
Var: NetworkAffinity .....	108

	List: StaticPushSource .....	108
	Var: AllowInitialExternalSubscription .....	108
6	MULTICASTING .....	111
	Multicasting Syntax .....	111
	List: Multicast .....	112
	Var: Enabled .....	112
	Var: AnnounceSAP .....	112
	Var: RTSPPort .....	113
	Var: TTL.....	113
	Var: Resend.....	113
	Var: AddressRange .....	114
	Var: DeliveryOnly.....	114
	List: ControlList.....	115
	List: Rule Number.....	115
	Var: Allow .....	115
7	PULL SPLITTING .....	117
	Pull Splitting Operation .....	117
	Server Compatibility .....	117
	File System Setup.....	118
	FSMount Syntax.....	118
	List: Broadcast Distribution List Name .....	119
	List: Splitter_DoubleURL .....	119
	Var: ShortName .....	119
	Var: MountPoint .....	120
	Var: Port .....	121
	Var: SplitterProtocol .....	121
	Broadcast Receiver Configuration.....	121
	Broadcast Receiver Syntax.....	122
	List: BroadcastReceiver .....	122
	List: Receivers.....	123
	List: Proxy Receiver Name .....	123
	Var: Protocol.....	123
	Var: SureStreamAware .....	124
	Var: PullSplitEnabled .....	124
	Var: UseTCPForPullBackchannel .....	125
	Var: PathPrefix.....	125
	Var: ResendSupported .....	126
	Var: FECLevel .....	126
	Var: UseRTSPInitiate.....	127
	Proxy List Splitting Variables.....	127
	Var: BroadcastDistributionMountPoint .....	128

- Var: BitsaveEnable..... 128
- Var: BitsaveMountPoint..... 129
- 8 ACCESS CONTROL AND AUTHENTICATION 131
  - Access Control ..... 131
    - Access Control Syntax ..... 131
    - List: AccessControl..... 132
    - List: RuleNumber ..... 132
    - Var: Access ..... 133
    - Var: From ..... 133
    - Var: To ..... 134
    - List: Ports ..... 134
    - Var: Port\_n ..... 135
  - Authentication Databases..... 135
    - Database Syntax ..... 135
    - List: Databases ..... 136
    - List: Database Name ..... 136
    - Var: PluginID ..... 136
    - Var: Path ..... 137
    - Var: Name ..... 137
    - Var: User..... 138
    - Var: Password ..... 138
    - Var: TableNamePrefix..... 138
  - Authentication Realms ..... 139
    - Realm Syntax ..... 139
    - List: AuthenticationRealms ..... 140
    - List: Realm Name ..... 140
    - Var: Realm ..... 140
    - List: Protocol Name ..... 141
    - Var: PluginID ..... 141
    - Var: DatabaseID ..... 142
    - Var: Provider ..... 142
    - Var: Group..... 143
  - Proxy Authentication ..... 143
    - Proxy Authentication Syntax..... 143
    - List: ProxyAuthentication ..... 144
    - Var: Enabled ..... 144
    - List: Authority ..... 145
    - Var: DatabaseID ..... 145
    - Var: Realm ..... 145
    - Var: AllowDuplicateIDs ..... 146
    - List: RuleList ..... 146

	List: Rule Name .....	146
	Var: NoAuthenticateHost.....	147
9	ACCESS AND ERROR LOGS .....	149
	Basic Access Log.....	149
	Basic Logging Variables.....	149
	Var: LoggingStyle.....	149
	Var: StatsMask.....	150
	Basic Access Log Syntax .....	151
	List: Basic Access Log .....	151
	Var: Type .....	152
	Var: Enabled .....	152
	Var: Logging Style.....	152
	List: Outputs.....	153
	List: Access Log .....	153
	Var: LogRollSize .....	153
	Var: Filename .....	154
	Var: LogRollFrequency .....	154
	Var: Type .....	155
	Basic Error Log.....	155
	Basic Error Log Syntax .....	155
	List: Basic Error Log.....	155
	Var: Type .....	156
	Var: Enabled .....	156
	List: Outputs.....	156
	List: Error Log .....	157
	Var: LogRollSize .....	157
	Var: Filename .....	158
	Var: LogRollFrequency .....	158
	Var: Type .....	158
	Advanced Logging Templates .....	159
	Advanced Logging Syntax.....	160
	List: Name .....	160
	Var: Type .....	161
	Var: Enabled .....	161
	Var: Format.....	162
	Var: Interval .....	163
	Var: Description .....	163
	Var: WatchRoot .....	164
	List: Outputs.....	164
	List: OutputTypeN.....	165
	List Output Variables.....	166

- Simple Network Monitoring Protocol (SNMP) Configuration ..... 170
  - SNMP Syntax ..... 170
  - List: SNMP..... 171
  - Var: Activate ..... 171
  - Var: Master Address ..... 171
  - Var: SendTrap ..... 172
  - Var: SNMPTrapInterval..... 172
  - List: TrapThresholds..... 173
  - Var: GlobalCPUUtilization ..... 173
  - Var: ServerStart ..... 173
  - Var: MemoryUsageWaterMark\_n ..... 174
  - Var: OutBandwidthWaterMark\_n ..... 174
  - Var: MaxConnections ..... 175

PART III: REGISTRY PROPERTIES

- 10 REGISTRY PROPERTIES ..... 179
  - Viewing the Registry..... 179
  - Date and Time Values..... 180
    - Date (MM/DD/YY)..... 180
    - Time of day by local time zone (HH:MM:SS) ..... 180
    - Greenwich Mean Time (HH:MM:SS)..... 180
    - Local Time Zone Difference..... 181
    - Hour of the day by local time zone (HH) ..... 181
    - Minute of the day (MM)..... 181
    - Second of the day (SS) ..... 182
  - Report Formatting Values ..... 182
    - Double Quote ..... 182
    - New Line ..... 182
    - Tab ..... 183
  - Obsolete Registry Properties ..... 183
- 11 CLIENT PROPERTIES ..... 185
  - Client Properties..... 185
    - CBID ..... 186
    - ClientID..... 187
    - ConnId ..... 188
    - ControlBytesSent ..... 188
    - GUID..... 188
    - IPAddress ..... 189
    - IsCloaked ..... 189
    - IsRDT ..... 190
    - Language ..... 190

PlayerStartTime .....	192
Port .....	193
Protocol.....	193
RequestMethod .....	194
StartTime.....	194
User-Agent.....	194
Version .....	195
Client Session Properties .....	196
ASMSubscribes .....	197
ASMUnsubscribes .....	198
AvgBitrate .....	198
BytesSent .....	198
Duration .....	199
DurationSeconds .....	199
EstimatedPlayerBufferOverruns .....	199
EstimatedPlayerBufferUnderruns .....	200
FailedResends .....	200
FileSize.....	201
InterfaceAddress.....	201
IsMulticastUsed .....	201
IsRAStreamFound .....	202
IsREStreamFound.....	202
IsRIStreamFound.....	202
IsRVStreamFound.....	203
IsUDP .....	203
LogStats .....	204
PacketsLost.....	204
PacketsSent.....	204
PlayerRequestedURL.....	204
PlayTime .....	205
ProxyConnectionType .....	205
RRFrequency.....	206
RTT .....	206
SendingTime .....	206
SessionID .....	207
SessionStartTime.....	207
Status .....	207
SuccessfulResends.....	209
TotalBitrateAdaptations .....	209
TotalDownshifts.....	210
TotalUpshifts .....	210
URL.....	210

12	PROXY REGISTRY PROPERTIES	211
	Proxy Registry Values .....	211
	BandwidthOutput.....	211
	BandwidthOutputPerPlayer .....	211
	BandwidthUsage.....	212
	BytesInUse.....	212
	BytesMemoryUsage .....	212
	ClientCount.....	213
	CloakedClientCount.....	213
	FileObjectCount.....	213
	HTTPClientCount .....	214
	Last10Seconds.....	214
	LoadState .....	215
	MMSClientCount.....	215
	MulticastTransportCount.....	216
	PercentAggCPUUsage.....	216
	PercentCPUUsage .....	217
	RTSPAttemptedStreamed .....	217
	RTSPClientCount .....	217
	TotalCAs .....	218
	TCPTransportCount.....	218
	UDPTransportCount.....	218
	Uptime .....	218
	platform .....	219
	version.....	219

## **OVERVIEW**

The following chapters explain the basics of using the Helix Proxy configuration file.



# INTRODUCTION

Welcome to Helix™ Proxy Version 11.1, the most powerful caching proxy server available for streaming media. Helix Proxy teams with media servers and players to optimize bandwidth and improve the playback experience. This manual will help you take full advantage of Helix Proxy by serving as a reference for editing the Helix Proxy configuration file.

## 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 advanced system administrators who prefer to edit Helix Proxy's configuration file directly rather than using Helix Administrator.

## How This Manual Is Organized

This guide contains the following chapters:

### Chapter 1: New Features

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

### **Chapter 2: Overview**

This chapter contains information you need about configuration file basics, text editing guidelines, and XML syntax used by the proxy configuration file, `rmproxy.cfg`.

### **Chapter 3: General Configuration**

This chapter describes general features that you can set up through the configuration file, including information about configuring ports, paths, and passwords.

### **Chapter 4: Basic Streaming Features**

This chapter explains configuration variables for basic streaming features, as well as proxy-related activities such as proxy routing and redundant proxies.

### **Chapter 5: Rate Control**

Refer to this chapter for information about server-side rate control and capabilities exchange.

### **Chapter 6: Multicasting**

This chapter discusses tags used for multicasting, in which Helix Proxy relays a single, live stream to multiple clients, rather than a separate stream to each client.

### **Chapter 7: Pull Splitting**

This chapter discusses improvements to the proxy splitting feature, including simplified port allocation and enhanced gateway bandwidth utilization.

### **Chapter 8: Access Control and Authentication**

Learn about tags to validate users attempting to access Helix Proxy.

### **Chapter 9: Access and Error Logs**

Helix Proxy can report client behavior with a customizable degree of detail. Use this chapter to learn about lists and tags associated with this feature.

### **Chapter 10: Registry Properties**

This chapter describes the basics of the Helix Proxy registry, which records configuration information and holds information about system performance.

### **Chapter 11: Client Properties**

Using the client properties, you can create advanced logging templates that report on media player activity.

## Chapter 12: Proxy Registry Properties

This chapter describes the server registry properties, which record information about proxy server load and operation.

## 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 Proxy administrators, the term “you” refers to the administrator.

RealNetworks clients, such as RealPlayer, or Windows Media Player are referred to generically as “clients”. Where information applies specifically to the RealNetworks® RealPlayer, this is spelled out.

**Note:** Although most clients currently in use are computers running RealPlayer or RealPlayer, RealNetworks also makes a software development kit (SDK) that enables other companies to develop their own players with which they can use to receive the various types of streamed data.

“Clips,” “content,” “media files,” and “files” are used interchangeably to indicate the material that Helix Proxy streams.

### Typographical Conventions

The following table explains the typographic conventions used in this guide:

Notational Conventions	
Convention	Meaning
<code>syntax</code>	This font is used for syntax of configuration files, URLs, or command-line instructions.
<i>variables</i>	Italic text represents variables. Substitute values appropriate for your system.
<b>emphasis</b>	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 access log, where statistics generated by the ClientStats variable are enclosed in regular brackets.

(Table Page 2 of 2)

**Default Locations and Values**

In all of the examples given in this book, it's assumed that you've installed Helix Proxy in the default location for your operating system and that you're using default values for all settings. Of course, you can customize Helix Proxy 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 the following:

C:\Program Files\Real\Helix Proxy

**Additional RealNetworks Resources**

In addition to this guide, the following RealNetworks resources are available at: <http://service.real.com/help/library/index.html>

- *Helix Proxy Administration Guide*

Information about Helix Proxy, powerful software for streaming media content is covered in this basic reference. This guide explains how to set up and configure Helix Proxy using Helix Administrator, an HTML-based, graphical user interface.

- *Helix Server Administration Guide*

Information about Helix Server, powerful software for streaming media content is covered in this basic reference. This guide explains how to set up, configure, and run Helix Server to stream multimedia.

- *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.

- *Helix Server and Helix Proxy Troubleshooting Guide*

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



## NEW FEATURES

Helix Proxy Version 11.1 facilitates greater extensibility and interoperability with third-party solutions. This chapter discusses the features that have been added to this version of Helix Proxy.

### New Features in Helix Proxy Version 11.1

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

#### **StreamerCount Variable**

The new `StreamerCount` variable determines how many streaming threads to create. See “Streamer Count” on page 44.

#### **SNMP Support**

Using Simple Network Monitoring Protocol (SNMP) version 1, 2c, or 3, you can monitor Helix Proxy Version 11.1 from any SNMP-compliant management system. The SNMP feature allows you to monitor Helix Proxy performance and update server configuration remotely.

**For More Information:** See “Simple Network Monitoring Protocol (SNMP) Configuration” on page 170.

#### **Reduced Media Start-Up Delay**

Helix Proxy 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.

## Delayed Shutdown

The delayed shutdown feature gives media players time to report playback statistics when Helix Server shuts down or restarts. The section “Delayed Shutdown” on page 33 explains how to set up this feature.

## Client Statistics Type 4 as Default

For the basic logging feature, statistics type 4, rather than types 1 and 2, is now the default for the access log. The section “Basic Access Log” on page 149 explains the access log configuration variables. In addition, 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 implementing a low-latency broadcast.

**For More Information:** For a description of client statistics, see the client statistics section in the basic logging chapter of *Helix Proxy Administration Guide*.

## Bandwidth Detection for RealPlayer 11

Helix Proxy Version 11.1 implements a new method of 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. This feature is turned on by default. You can turn it off through the `SendBWpackets` variable described in the section “Automatic Bandwidth Detection” on page 36.

## IPv6 Support

Helix Proxy supports Internet Protocol version 6 addresses (IPv6) for most features. This allows you to run Helix Proxy on a dual-stack IPv4/IPv6 machine and take advantage of IPv6 addressing where possible. A new variable, `UseIPv4only`, allows you to force IPv4-only behavior on a dual-stack machine.

**For More Information:** The section “IP Binding” on page 26 explains the syntax for binding Helix Proxy to certain IPv4 or IPv6 addresses.

## 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 implementation information, refer to “Protocol Options” on page 46.

## On-The-Fly Packetization of Non-Hinted MPEG Content

Helix Proxy can stream several types of MPEG content regardless of whether or not the clip contains a hint track. Through the configuration file, you can turn off on-the-fly packetization. For more information, refer to “Datatype Packetization” on page 60.

## Capabilities Exchange

The capabilities exchange feature enables Helix Proxy to learn the streaming media features of a media player. For example, Helix Proxy can determine the media player’s buffering capacity. See “Capabilities Exchange” on page 67.

## Rate Control

Using rate control, Helix Proxy can vary the streaming rate of a multi-rate clip based on server-side modeling of a media player’s buffer. See “Rate Control” on page 72.

## Support for Differentiated Services

Helix Proxy can set the bits for precedence and quality of service in IP packets for many 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. See “Differentiated Services” on page 62.

## Configurable RTSP Timeout Value

You can now set the amount of time that can elapse before Helix Proxy closes an idle RTSP connection. Refer to “Var: KeepAliveInterval” on page 31 for information about setting this timeout value.

## Logging Enhancements

Helix Proxy 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 basic access and error logs are now predefined templates within the advanced logging feature described in Chapter 9, “Access and Error Logs”. However, you can continue to use the same error and access logs you used in previous releases.

**Note:** Within the configuration file, the basic access and error logs are defined along with the advanced logging templates, and the `LogPath` and `ErrorLogPath` variables that indicated the location of these basic log files are now obsolete.

## Removed Features in Helix Proxy Version 11.1

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

### Progressive Networks Audio (PNA)

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

### Legacy Bandwidth Negotiation

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

## Upgrade Issues

This section explains issues about upgrading to Helix Proxy Version 11.1.

### Compatibility with Previous Versions

Aside from the issues described in the following sections, there are no known compatibility problems between Helix Proxy Version 11.1 and Helix Proxy Versions 9 and 10 or RealSystem Proxy version 8. You can use an older configuration file with Helix Proxy 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.

### ProcessorCount Variable Obsolete

The new StreamerCount variable replaces ProcessorCount, which should be removed from the configuration file. For more information, refer to “Streamer Count” on page 44.

### Access Logging Templates

The logging feature introduced with version 10 includes templates that replicate the standard access and error logs used in previous versions of Helix Proxy. These templates are turned on by default, and use the default values for log files in prior releases. If you are upgrading from version 9 or earlier, and your current access log does not use the default logging style 5 or client statistics 1 and 2, you need to modify the templates after you upgrade to set the values used in your current log files.

**For More Information:** See “Basic Access Log” on page 149.

### Media Player Session Templates

The Client Stats template type, which was introduced with version 10, records information about media player connections. A Session template no longer logs media player information. If you are upgrading from version 9 or earlier, and used Session templates to create reports when media players connected and disconnected, those templates no longer function with Helix Proxy Version 11.1. You need to recreate your reports using Client Stats templates.

**For More Information:** See “Advanced Logging Templates” on page 159.

## **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 “IP Binding” on page 26 for an explanation of how to set bindings and a list of additional binding options.

## OVERVIEW

This chapter explains the structure of the Helix Proxy configuration file. You need to understand the basics of the configuration file's XML-based syntax before you change the configuration parameters described in this guide.

### Methods for Configuring Features

You can configure most Helix Proxy features either through the HTML-based Helix Administrator or by editing the configuration file manually. There are a few advanced features that you can change only by editing the configuration file, however:

- Modifying how Helix Administrator operates requires manual configuration. See the section “Helix Administrator” on page 37.
- Changing the default datatype packetization for MPEG files. Refer to the section “Datatype Packetization” on page 60.
- Configuring capabilities exchange and rate control, which are described in Chapter 5.

**Tip:** When configuring other features, RealNetworks recommends that you use Helix Administrator.

### Editing the Configuration File

The configuration file holds the Helix Proxy information in a series of XML-formatted lists and variables. The standard configuration file is `rmproxy.cfg`, which resides in the main Helix Proxy installation directory. But at startup, you can specify an alternate file, which might be a file that you have manually edited using `rmproxy.cfg` as a starting point. Helix Proxy reads the specified configuration file at startup, and then loads the configuration values into its registry.

The Helix Proxy installation directory also contains a backup copy of the configuration file named `default.cfg`. This is a mirror image of the default `rmproxy.cfg` file that was created during installation. You can restore your configuration file from the backup copy if you make changes that you want to undo, or if you accidentally delete the main copy.

**Tip:** If you have multiple proxies, you may want to name each configuration file differently to identify which proxy you're working with. To ensure proxy integrity, be sure to store each configuration file where only authorized users can change it.

## Helix Administrator

Helix Administrator—the HTML-based configuration tool for Helix Proxy—updates the configuration file automatically when you change settings through the Helix Administrator user interface, as described in the installation chapter of *Helix Proxy Administration Guide*. You can also change Helix Proxy 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 to the configuration file manually.

**Tip:** To become familiar with configuration information, make changes first through Helix Administrator, examining the file afterward to see how the changes were made. When you do edit the configuration file manually, log off of Helix Administrator to ensure that no editing conflicts occur.

## XML-Based Syntax

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:

- XML declaration tag
- optional comment tags
- list tags
- variable tags

Of these four types, only lists and variables make up the instructions to Helix Proxy. 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 Proxy looks for exact spellings and correct use of angle brackets. Helix Proxy 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, quit Helix Administrator before opening the configuration file in a text editor.

## XML Declaration Tag

The XML declaration tag indicates which version of XML is in use. Helix Proxy 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 place 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 it is recommended for clarity.

**Note:** Not all possible lists are predefined within the configuration file. Some lists are created only when you activate and define a feature through Helix Administrator. You can also add lists and sublists manually by using the syntax described in this guide.

## 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 `MonitorPassword`), or they can 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. You do this by adding a number to the end of each name, such as `Extension_01`, `Extension_02`, and so on.

**Tip:** If you've restarted Helix Proxy and it's not responding to a change you've made to a variable, make sure that the variable has a closing forward slash mark, and that there is no space between the variable name and the closing slash.

## File System Section

The file system section of the configuration file—comprising the FSMount list, sublists, and variables—contains many entries for a number of different features:

```
<List Name="FSMount">
    ...file system mount point information...
</List>
```

The FSMount list primarily controls the configuration of the plug-ins used to access files such as media clips, live broadcasts, HTML pages, and so on. In many cases, a feature may have a mount point or other component configured within the FSMount list. The lists and variables that define how the feature works are then defined elsewhere in the configuration file.

**For More Information:** Plug-ins are stored in a directory indicated by the PluginDirectory variable, which is described in the section Var: PluginDirectory on page 21.

## Applying Configuration Changes

Typically, you need to restart Helix Proxy after you modify the configuration file manually. Helix Proxy reads the configuration information only at startup. This enables you to modify the file while Helix Proxy is running. But it also necessitates a restart to load the configuration changes into memory.

**For More Information:** Refer to the installation chapter of *Helix Proxy Administration Guide* for start-up instructions.

## Using the SIGHUP Command on UNIX

If you change the Helix Proxy file manually on a UNIX operating system, you can use SIGHUP to upload the changes to Helix Proxy without breaking any open connections, as long as the changes do not require a full proxy restart. To have Helix Proxy re-read the configuration file, use the following SIGHUP command:

```
kill -HUP processID
```

Here, *processID* is the Helix Proxy process number, as shown in the Logs/rmpoxy.pid file.

**Tip:** Helix Administrator indicates when changes require a full restart. Use it as your guide to changes that you can and cannot upload with SIGHUP.



## CONFIGURATION FILE SYNTAX

The following chapters explain the syntax, lists, and variable values of the Helix Proxy configuration file.



## GENERAL CONFIGURATION

This chapter explains the configuration lists and variables used to set up general features of Helix Proxy. This includes the variables that configure ports used for streaming media requests, as well as variables that regulate how many media players can connect to Helix Proxy.

### Paths

The configuration file provides several paths to various files used to support Helix Proxy. The following example shows path variables, along with typical paths used in Windows systems:

```
<Var PluginDirectory="C:\Program Files\Real\Helix Proxy\Plugins"/>  
<Var SupportPluginDirectory="C:\Program Files\Real\Helix Proxy\Lib"/>  
<Var LicenseDirectory="C:\Program File\Real\Helix Proxy\License"/>
```

The paths in a configuration file running on a UNIX system are similar to those on a Windows system. However, UNIX systems uses an additional variable named `PidPath`.

```
<Var PluginDirectory="/usr/bin/Helix Proxy/Plugins"/>  
<Var SupportPluginDirectory="/usr/bin/Helix Proxy/Lib"/>  
<Var LicenseDirectory="/usr/bin/Helix Proxy/License"/>  
<Var PidPath="/usr/bin/Helix Proxy/Logs/rmserver.pid"/>
```

### Var: PluginDirectory

Shows where the plug-in files are stored.

#### Tag Syntax

```
<Var PluginDirectory="Directory Path" />
```

#### Possible Values

Directory path to the plug-in files.

### Example

```
<Var PluginDirectory="C:\Program Files\Real\Helix Proxy\Plugins"/>
```

## Var: SupportPluginDirectory

Shows the location of the Lib directory.

### Tag Syntax

```
<Var SupportPluginDirectory="Directory Path" />
```

### Possible Values

Directory path to the plug-in library files.

### Example

```
<Var SupportPluginDirectory="C:\Program Files\Real\Helix Proxy\Lib"/>
```

## Var: LicenseDirectory

Shows the location of the license files.

### Tag Syntax

```
<Var LicenseDirectory="Directory Path" />
```

### Possible Values

Directory path to the license files.

### Example

```
<Var LicenseDirectory="C:\Program File\Real\Helix Proxy\License"/>
```

## Var: PidPath

The PidPath variable indicates the location of the process id file.

### Tag Syntax

```
<Var PidPath="File Path" />
```

### Possible Values

PID path and PID file name.

**Example**

```
<Var PidPath="/usr/bin/Helix Proxy/Logs/rmserver.pid"/>
```

**Var: LogPath**

The LogPath variable formerly supplied the path and name of the access log file. This variable is present in configurate files through Helix Proxy version 9, but is no longer used. Access logs are described in Chapter 9: Access and Error Logs.

**Var: ErrorLogPath**

The ErrorLogPath variable formerly supplied the path and name of the error log file. This variable is present in configurate files through Helix Proxy version 9, but is no longer used. Error logs are described in Chapter 9: Access and Error Logs.

**Ports**

The Helix Proxy configuration file provides variables that set values of the ports used for protocols such as RTSP and MMS, 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 Proxy.

**For More Information:** For more information about ports, refer to the setup chapter in *Helix Proxy Administration Guide*.

**Ports Syntax**

```
<Var RTSPPort="554"/>
<Var HTTPPort="80"/>
<Var MMSPort="1755"/>
<Var MonitorPort="9090"/>
<Var AdminPort="9999"/>
```

**Var: RTSPPort**

The port where Helix Proxy listens for RTSP requests between Helix Proxy, RealPlayer, and QuickTime Player. At installation, the value is 554. This is an

internal port used by Helix Proxy to communicate with its file systems. External access to this port is restricted.

**Note:** To use a port lower than 1024 on a UNIX system, you must be logged on as super-user.

#### Tag Syntax

```
<Var RTSPPort="port"/>
```

#### Possible Values

Port number of the RTSP port.

#### Example

```
<Var RTSPPort="554"/>
```

### Var: HTTPPort

Provides HTTP-based communication between Helix Proxy and Helix Administrator. The default value of this variable is 80.

#### Tag Syntax

```
<Var HTTPPort="port"/>
```

#### Possible Values

Port number of the HTTP port.

#### Example

```
<Var HTTPPort="80"/>
```

### Var: MMSPort

Provides MMS-based communication between Helix Proxy and Windows Media Player. The default value for this variable is 1755.

#### Tag Syntax

```
<Var MMSPort="port"/>
```

#### Possible Values

Port number of the MMS port.

**Example**

```
<Var MMSPort="1755"/>
```

**Var: MonitorPort**

The port that monitors connections to Helix Proxy. It is used by Proxy Monitor. The default value for this variable is 9090.

**Tag Syntax**

```
<Var MonitorPort="port"/>
```

**Possible Values**

Port number of the monitor port.

**Example**

```
<Var MonitorPort="9090"/>
```

**Var: AdminPort**

Provides communication with Helix Administrator. The value is randomly generated during installation, and is required in the browser URL when connecting to Helix Administrator.

**Tag Syntax**

```
<Var AdminPort="port"/>
```

**Possible Values**

Port number of the administration port

**Example**

```
<Var AdminPort="12345"/>
```

**Monitor Password**

Configured during installation, the MonitorPassword variable is used internally to connect the Proxy Monitor to the Helix Administrator. It is also used as the default password, initially, to gain access to Helix Administrator.

**Note:** You can change usernames and passwords to control access to Helix Administrator by working with authentication, which Chapter 8 describes.

**Warning!** After installation, do not edit the Monitor Password value because it is used for an internal function.

### Tag Syntax

```
<Var MonitorPassword="password"/>
```

### Possible Values

Password defined during installation.

### Example

```
<Var MonitorPassword="letmein"/>
```

## IP Binding

The IPBindings list determines which IP addresses Helix Proxy uses. By default, Helix Proxy binds to all available IPv4 and IPv6 addresses. You can bind Helix Proxy to fewer addresses through the IPBindings list. This list uses variables numbered sequentially: Address\_1, Address\_2, and so on for each IP address that Helix Proxy uses.

**For More Information:** For more on IP binding, see the server set-up chapter of *Helix Proxy Administration Guide*.

### IP Binding Syntax

The following is sample syntax for the IPBindings list:

```
<List Name="IPBindings">  
  <Var Address_1="*" />  
</List>
```

**Note:** The IP bindings syntax does not exist in the default configuration file because Helix Proxy binds to all available IP addresses automatically. Helix Proxy creates this list if you set up IP bindings through Helix Administrator. You can also define this list manually.

## List: IPBindings

This list contains the IP addresses or DNS names reserved for use by Helix Proxy.

```
<List Name="IPBindings">
  ...IP addresses...
</List>
```

### Registry Value

config.IPBindings

## Var: Address\_n

Each Address\_n variable reserves an address for use by Helix Proxy. If you don't use any values for the variables in the IPBindings list, Helix Proxy binds to the localhost address and the first network interface it finds on the computer—network interface 0. It does not bind to any other addresses.

### Tag Syntax

```
<Var Address_n="address"/>
```

### Possible Values

You can use any of the following:

- IPv4 Addresses

You can provide one or more dotted decimal IPv4 addresses, such as 205.23.5.108. Helix Proxy reserves each address you specify for its use.

- IPv6 Addresses

If the Helix Proxy machine has an IPv6 stack, you can specify the IPv6 addresses in colon-delimited, hexadecimal form, such as 1080:0:0:0:8:800:200C:417A.

**Tip:** Helix Proxy supports the use of a double colon (“::”) to compress fields containing only zeroes. You could therefore represent the preceding address as 1080::8:800:200C:417A. The double colon can appear at the beginning, middle, or end of the address. Only one double colon can appear within an address.

- Localhost Addresses

The *localhost* address (also called the *loopback* address) enables a simulated network connection from Helix Proxy to a client installed on the same computer. 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.

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

- All Addresses

The shorthand values shown in the following table allow you to reserve all IPv4 or IPv6 computer addresses, including the localhost address, for use by Helix Proxy.

**Syntax for Capturing All IP Addresses**

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

**Warning!** If you use an option to capture all IP addresses of a certain type, do not use any other variables for that type of address in the `IPBindings` list. For example, if you specify 0.0.0.0, do not include any other IPv4 addresses. If you do, Helix Proxy will not be able to start up.

- DNS Name

You can specify a DNS host name on the computer reserved for use by Helix Proxy. Helix Proxy will run more efficiently if you use IP addresses, however.

**Example**

```
<!-- IPv4 bindings -->
<Var Address_1="205.23.5.108"/>
<!-- IPv6 bindings -->
<Var Address_2="::"/>
```

**Var: UseIPv4Only**

If your Helix Proxy machine includes both IPv4 and IPv6 stacks, Helix Proxy takes advantage of IPv6 addressing when it can. You can force IPv4-only behavior by adding the `UseIPv4Only` variable to the configuration file. If you enable this variable, ensure that the configuration file lists only IPv4 addresses or DNS names where machine addresses are required.

**Note:** Add this variable to the end of the configuration file. Do not include it in any other lists, such as the IP bindings list.

**Tag Syntax**

```
<Var UseIPv4Only="Boolean"/>
```

**Possible Values**

- 0|False – Take advantage of IPv6 features when possible.
- 1|True – Use only IPv4 addressing and features.

**Example**

```
<Var UseIPv4Only="1"/>
```

**Connection Control**

Helix Proxy uses several methods for managing bandwidth of streaming media. Whether you implement just one method, or you use several together, you have the ability to moderate the amount of streaming media traffic on your network.

**Note:** The list discussed in this chapter accommodates variables for both bandwidth and pull-splitting features. This chapter covers bandwidth. To configure pull splitting, refer to Chapter 7.

**For More Information:** For more information about managing the amount of bandwidth Helix Proxy uses, refer to the setup chapter of *Helix Proxy Administration Guide*.

## Connection Control Syntax

The following are the variables used for connection control:

```
<List Name="Proxy">
  <Var MaxProxyConnections="0"/>
  <Var MaxProxyBandwidth="0"/>
  <Var MaxGatewayBandwidth="0"/>
  <Var KeepAliveInterval="60"/>
</List>
```

### List: Proxy

Use this list to contain variables for bandwidth control. For variables included in this list for pull splitting, see “Proxy List Splitting Variables” on page 127.

#### Registry Value

config.Proxy

### Var: MaxProxyConnections

Limits the number of clients that can be connected to Helix Proxy simultaneously. This variable must be set to less than or equal to the number of streams in your license.

**Tip:** If you establish values for MaxProxyConnections, MaxProxyBandwidth, and MaxGatewayBandwidth concurrently, Helix Proxy will limit access when the lowest threshold is reached.

#### Tag Syntax

```
<Var MaxProxyConnections="integer"/>
```

#### Possible Values

Integer in the range 1 to 32767, where the maximum is the number of streams in the license. If this variable is omitted or set to 0, Helix Proxy uses the capacity determined by its license.

**Example**

```
<Var MaxProxyConnections="25"/>
```

**Var: MaxProxyBandwidth**

Establishes a threshold on the amount of bandwidth Helix Proxy uses for connections, in kilobits per second (Kbps). Helix Proxy will make no new connections once this threshold has been crossed. This is not a per-connection setting. The default value is 0, which means no limit is enforced by the proxy.

**Tag Syntax**

```
<Var MaxProxyBandwidth="integer"/>
```

**Possible Values**

Integer that sets the maximum bandwidth in Kbps. When Helix Proxy has used this amount, it no longer establishes connections to clients.

**Example**

```
<Var MaxProxyBandwidth="64"/>
```

**Var: MaxGatewayBandwidth**

Restricts the bandwidth in use between Helix Proxy and Helix Server.

**Tag Syntax**

```
<Var MaxGatewayBandwidth="integer"/>
```

**Possible Values**

The maximum bandwidth in Kbps in use by the proxy in connections to the server. The default value is 0, which means no limit is enforced by the proxy.

**Example**

```
<Var MaxGatewayBandwidth="64"/>
```

**Var: KeepAliveInterval**

Determines the number of seconds to wait for client feedback before querying the client's status and closing the connection if no response is received.

### Tag Syntax

```
<Var KeepAliveInterval="integer"/>
```

### Possible Values

- $n$  – A positive integer representing the number of a seconds. The default is 80.
- 0 – Turn the timer off and do not time out the player connection.

### Example

```
<Var KeepAliveInterval="80"/>
```

## UNIX Group and User Names

By default, Helix Proxy 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 Proxy as root, so that it can capture port 554 for RTSP communications, then assume a different user and group identity.

**Note:** The user and group names must be predefined through the operating system, and must have write permission for Helix Proxy's Logs and Secure directories.

**For More Information:** For more on UNIX users and groups, refer to the setup chapter in *Helix Proxy Administration Guide*.

### Var: Group

The group name under which Helix Proxy runs. The group name must already exist on the computer on which Helix Proxy is running; otherwise, Helix Proxy will not start. If you do not specify a group name, this variable defaults to the group name of the user who first starts Helix Proxy.

### Tag Syntax

```
<Var Group="Group Name"/>
```

### Possible Values

- *Group Name* – The group name. The default value is %1.

**Example**

```
<Var Group="users"/>
```

**Var: User**

The user name under which Helix Proxy runs. The user name must exist on the computer on which Helix Proxy is running. Otherwise, Helix Proxy will not start. If you don't specify a user name during setup, the user name defaults to the user name of the user who first logs in and starts Helix Proxy.

**Tag Syntax**

```
<Var User="User Name"/>
```

**Possible Values**

- *User Name* – The user name. The default value is %-1.

**Example**

```
<Var User="canderson"/>
```

**Delayed Shutdown**

The delayed shutdown feature lets you initiate a Helix Proxy 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 Proxy terminates all streams and shuts down immediately upon request, receiving no playback statistics from media players.

**Note:** The delayed shutdown feature is not configured by default. To implement this feature, add the list and variables described in the following section or set up the feature through Helix Administrator.

**For More Information:** For more information about the delayed shutdown feature, including the proper way to initiate a delayed shutdown, refer to the setup chapter in *Helix Proxy Administration Guide*.

## List: ServerShutdown

This list defines the delayed shutdown feature:

```
<List Name="ServerShutdown">  
  ...list of delayed shutdown variables...  
</List>
```

### Registry Value

config.ServerShutdown

## Var: ClientDisconnectInterval

The ClientDisconnectInterval variable defines the time in seconds that elapses between the shutdown request and cessation of all media streams.

### Tag Syntax

```
<Var ClientDisconnectInterval="seconds"/>
```

### Possible Values

- $n$  – A value in seconds. A value of 30, for example, gives media players nearing the end of a clip 30 seconds to reach the end of the stream.
- 0 – Begin disconnecting streams immediately upon the shutdown request, but allow media players time to report playback statistics.

### Example

```
<Var ClientDisconnectInterval="30"/>
```

## Var: ShutdownProceedTime

The ShutdownProceedTime variable defines an interval that begins once the client disconnect interval expires. It sets the time in seconds during which Helix Proxy shuts down the media sessions and receives media player statistics.

### Tag Syntax

```
<Var ShutdownProceedTime="seconds"/>
```

### Possible Values

- $n$  – A value in seconds. A value of 30, for example, provides a 30-second window for shutting down streams and reporting playback statistics.

**Example**

```
<Var ShutdownProceedTime="30"/>
```

**Var: NewClientConnectionsAllowed**

The `NewClientConnectionsAllowed` variable allows Helix Proxy to accept new media requests once the shutdown request is given.

**Tag Syntax**

```
<Var NewClientConnectionsAllowed="Boolean"/>
```

**Possible Values**

- 0|False – Do not accept new stream requests once the shutdown or restart request has been given.
- 1|True – Accept new media requests until the `ClientDisconnectInterval` expires.

**Example**

```
<Var NewClientConnectionsAllowed="1"/>
```

**Var: LogPlayerTermination**

The `LogPlayerTermination` variable defines whether Helix Proxy records the following information about successful and unsuccessful media player terminations in its error log:

- 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:** This variable 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:** For more on the error log, refer to “Basic Error Log” on page 155.

### Tag Syntax

```
<Var LogPlayerTermination="Boolean"/>
```

### Possible Values

- 0|False – Aggregate termination statistics not written to the error log.
- 1|True – Aggregate termination statistics written to the error log.

### Example

```
<Var LogPlayerTermination="1"/>
```

## Automatic Bandwidth Detection

Helix Proxy uses a method of bandwidth detection that allows RealPlayer 11 and later to receive the optimal stream when connected to different networks. This eliminates the need for the user to change the bandwidth configuration manually. You can turn this feature off through the configuration file.

**For More Information:** For information on automatic bandwidth detection, refer to the overview chapter in *Helix Proxy Administration Guide*.

### Bandwidth Detection Syntax

The automatic bandwidth feature comprises a single list and variable. This list should not be contained within any other lists:

```
<List Name="AutoBWDetection">  
  <Var SendBWPackets="1">  
</List>
```

### List: AutoBWDetection

This list defines the automatic bandwidth detection feature:

```
<List Name="AutoBWDetection">  
  ...variable for turning automatic bandwidth detection on or off...  
</List>
```

### Registry Value

config.AutoBWDetection

### Var: SendBWPackets

The SendBWPackets variable determines if Helix Proxy uses its automatic bandwidth detection feature with RealPlayer 11 and later.

### Tag Syntax

```
<Var SendBWPackets="Boolean"/>
```

### Possible Values

- 0|False – Do not use automatic bandwidth detection. All RealPlayer users must manually reconfigure the optimal bandwidth setting in the preferences when changing network configurations.
- 1|True – Implement automatic bandwidth detection for RealPlayer 11 and later.

### Example

```
<Var SendBWPackets="1">
```

## Helix Administrator

Several lists and variables define the HTML-based Helix Administrator configuration tool. The administration file system accepts the initial URL for Helix Administrator and requests the HTML files from the local file system. Once the local file system delivers the HTML files, the administration file system looks up the Helix Proxy configuration values and displays them in the appropriate HTML pages in Helix Administrator.

**Tip:** The lists and variables that define Helix Administrator typically do not need to be edited after the installation. If you are copying a configuration file to use on a different installation of Helix Proxy, you may need to change the base paths.

**For More Information:** For information on using Helix Administrator, refer to the installation chapter in *Helix Proxy Administration Guide*.

## Helix Administrator Syntax

The following sample shows the default lists used by the Helix Administrator. Installation paths vary depending on the operating system and user choice. These lists fall within the FSMount section of the configuration file:

```
<!-- Admin File System -->
<List Name="RealSystem Administrator Files">
  <Var ShortName="pn-admin"/>
  <Var MountPoint="/admin"/>
  <Var BaseMountPoint="/admin/html"/>
  <Var Realm="naan.AdminRealm"/>
</List>
<!-- Local File System; Admin HTML -->
<List Name="RealSystem Administrator HTML">
  <Var ShortName="pn-local"/>
  <Var MountPoint="/admin/html"/>
  <Var BasePath="/usr/local/Real/HelixProxy/HelixAdministrator"/>
</List>
<!-- Local File System; Admin DOCS -->
<List Name="RealSystem Administrator DOCS">
  <Var ShortName="pn-local"/>
  <Var MountPoint="/admin/Docs"/>
  <Var BasePath="/usr/local/Real/HelixProxy/HelixAdministrator/Docs"/>
</List>
<!-- Local File System; Admin IMAGES -->
<List Name="RealSystem Administrator IMAGES">
  <Var ShortName="pn-local"/>
  <Var MountPoint="/admin/images"/>
  <Var BasePath="/usr/local/Real/HelixProxy/HelixAdministrator/images"/>
</List>
<!-- Local File System; Admin MONITOR -->
<List Name="RealSystem Administrator MONITOR">
  <Var ShortName="pn-local"/>
  <Var MountPoint="/admin/JavaMonitor"/>
  <Var BasePath="/usr/local/Real/HelixProxy/HelixAdministrator/JavaMonitor"/>
</List>
```

### List: RealSystem Administrator Files

The RealSystem Administrator Files list is the link between the local file system and the administration file system. Its registry value is `config.FSMount.RealSystemAdministratorFiles`. It uses the following variables:

- ShortName – Use a value of `pn-admin`.

- **MountPoint** – The default value is `/admin/`. If you change this mount point, you will need to type a new URL to access Helix Administrator.
- **BaseMountPoint** – Indicates the mount point used in the RealSystem Administrator HTML list. Use the same value used for MountPoint in the RealSystem Administrator HTML list.
- **Realm** – Authenticates Helix Administrator users. The realm ID consists of the server name, a period, and AdminRealm, as shown here:

```
<Var Realm="naan.AdminRealm"/>
```

### List: RealSystem Administrator HTML

The RealSystem Administrator HTML list defines the mount point and location of Helix Administrator files. Its registry value is `config.FSMount.RealSystemAdministratorHTML`. It uses the following variables:

- **ShortName** – Use a value of `pn-local`.
- **MountPoint** – The default value is `/admin/html/`.
- **BasePath** – Indicates the location of the HTML pages, style sheets, scripts, and other non-graphics files for Helix Administrator.

### List: RealSystem Administrator DOCS

The RealSystem Administrator DOCS list defines the mount point and location for on-line documentation. It has a registry values of `config.FSMount.RealSystemAdministratorDOCS`. It uses the following variables:

- **ShortName** – Use a value of `pn-local`.
- **MountPoint** – The default value is `/admin/Docs/`.
- **BasePath** – Indicates the location of online documentation.

### List: RealSystem Administrator IMAGES

The RealSystem Administrator IMAGES list defines the mount point and location for Helix Administrator graphics files. It has a registry values of `config.FSMount.RealSystemAdministratorIMAGES`. It uses the following variables:

- **ShortName** – Use a value of `pn-local`.
- **MountPoint** – The default value is `/admin/images/`.

- BasePath – Indicates the location of the graphics files for Helix Administrator.

### **List: RealSystem Administrator MONITOR**

The RealSystem Administrator MONITOR list defines the mount point and location for Proxy Monitor files. It has a registry value of `config.FSMount.RealSystemAdministratorJAVAMONITOR`. It uses the following variables:

- ShortName – Use a value of `pn-local`.
- MountPoint – The default value is `/admin/JavaMonitor/`.
- BasePath – Indicates the location of files for the Proxy Monitor.

## BASIC STREAMING FEATURES

This chapter describes the configuration variables for basic streaming features, as well as proxy-specific features such as proxy routing and redundant proxies.

### On-Demand Streaming

The on-demand streaming mount point section appears under the FSMount list. Use these tags to set up mount points for on-demand media. You can create as many mount points as you like, as long as each is unique. This sample shows the default mount point for on-demand content.

```
<List Name="FSMount">
  ...additional features found in FSMount section...
  <List Name="RealSystem Content">
    <Var ShortName="pn-local"/>
    <Var MountPoint="/" />
    <Var BasePath="/home/user/Content"/>
  </List>
  ...additional features found in FSMount section...
</List>
```

#### List: FSMount

The FSMount list defines the names of all the configurable file system plug-ins in use. The plug-ins themselves are stored in a directory indicated by the PluginDirectory variable. All requests of Helix Proxy are processed by plug-ins.

**Note:** For more information about the PluginDirectory variable see “Var: PluginDirectory” on page 21.

## Var: ShortName

Each list within FSMount gives a short name for the plug-in. The short name is also stored within the plug-in file itself, and Helix Proxy uses this to identify the correct file to use.

### Tag Syntax

```
<Var ShortName="name"/>
```

### Possible Values

- `pn-local` – For content stored on a local drive.
- `pn-network` – For content stored on any sort of external data store such as an NFS or a SAN.
- `name` – For content stored in a different medium, such as a database. In this case, the ShortName value is defined by the plug-in used to retrieve the content from the medium.

### Example

```
<Var ShortName="pn-local"/>
```

## Var: Mountpoint

A Mountpoint variable defines a virtual path that invokes a Helix Proxy plug-in. This variable is used in several file systems. Its value depends on the list in which it is contained.

### Tag Syntax

```
<Var MountPoint="string"/>
```

### Possible Values

Each MountPoint must be unique and cannot contain spaces or any of the following special characters:

```
=!@#%&*()[{}];'"<>,?|~\
```

Possible values are:

- `/` – Default mount point for content.
- `/Name` – Mount point for any other content directory.

**Examples**

```
<Var MountPoint="/" />
<Var MountPoint="/videos" />
```

**Var: BasePath**

BasePath is the physical location of the particular feature in the Helix Proxy installation.

**Tag Syntax**

```
<Var BasePath="string" />
```

**Possible Values**

Use the full path as specified on the operating system.

**Example**

```
<Var BasePath="/home/export/san/realcontent" />
```

**HTTP Delivery**

HTTP is typically used for Web pages. With Helix Proxy, HTTP is used to display Helix Administrator pages and HTML-based documentation. Although Helix Administrator can stream media through HTTP, Helix Proxy is not an HTTP protocol proxy and thus does not handle any streaming media requests made through HTTP between clients and an origin Helix Server.

**HTTP Delivery Syntax**

The following is the HTTP delivery syntax:

```
<List Name="HTTPDeliverable">
  <Var Path_1="/admin" />
</List>
```

**List: HTTPDeliverable**

This list indicates the virtual paths that contain content can be streamed over HTTP.

```
<List Name="HTTPODeliverable">  
  ...path to stream over HTTP...  
</List>
```

**Note:** Be sure that admin path for Helix Administrator is in this list.

### Registry Value

config.HTTPODeliverable

### Var: Path\_n

Each Path variable gives the name of a mount point, directory, or virtual directory whose content can be streamed over HTTP.

### Tag Syntax

```
<Var Path_n="pathname"/>
```

### Possible Values

- admin—Refers to Helix Administrator page, which are served over HTTP.

### Example

```
<Var Path_0="/admin"/>
```

## Streamer Count

The top-level StreamerCount variable determines how many streaming threads to create. Typically, Helix Proxy sets this number automatically depending on the number of CPUs on the machine.

**Note:** This variable replaces the ProcessorCount variable used in earlier releases. If your configuration file contains the ProcessorCount variable, RealNetworks recommends that you delete the variable from the file.

### Tag Syntax

```
<Var StreamerCount="Integer"/>
```

### Possible Values

- $n$  – An integer representing the number of streaming threads to run. This is typically one thread for each CPU, but may be different.
- 0 – Set the streamer count automatically. This is the default.

**Note:** RealNetworks recommends that you keep the value as the default unless instructed to change it by RealNetworks support personnel.

### Example

```
<Var StreamerCount="0"/>
```

## Transport Buffers

Two variables—`TCPBufferSize` and `UDPBufferSize`—set the buffer size for each live or on-demand stream delivered using the TCP or UDP transport, respectively. You may need to increase the buffer sizes if media players frequently experience dropped packets. This problem is most noticeable shortly after a high-bandwidth, live broadcast begins to stream.

**Tip:** Increase a buffer size incrementally until media players no longer experience the problem. Keep in mind that increasing a transport buffer size increases overall Helix Proxy memory use.

### Var: TCPBufferSize

Sets the Helix Proxy buffer size for each stream that uses the TCP transport.

### Tag Syntax

```
<Var TCPBufferSize="bytes"/>
```

### Possible Values

An integer that sets the TCP buffer size in bytes. The default is 32768 (32 Kilobytes).

### Example

```
<Var TCPBufferSize="65536"/>
```

## Var: UDPSendBufferSize

Sets the Helix Proxy buffer size for each stream that uses the UDP transport.

### Tag Syntax

```
<Var UDPSendBufferSize="bytes"/>
```

### Possible Values

An integer that sets the UDP buffer size in bytes. The default is 16384 (16 Kilobytes).

### Example

```
<Var UDPSendBufferSize="32768"/>
```

## Protocol Options

The Protocols list is not included in the configuration file by default. You can add this list along with the PreferClientTCP variable to choose TCP over UDP delivery for media players that list TCP as a preference.

### Protocol Options Syntax

The following sample shows the protocol options syntax:

```
<List Name="Protocols">  
  <List Name="RTSP">  
    <Var PreferClientTCP="1"/>  
  </List>  
</List>
```

### List: Protocols

The main list for specifying protocol options:

```
<List Name="Protocols">  
  ...protocols with available options...  
</List>
```

### Registry Value

```
config.Protocols
```

## List: RTSP

Within the Protocols list, there is a separate list for each protocol that supports protocol options. Currently, RTSP is the only protocol with available options:

```
<List Name="RTSP">
  ...RTSP options...
</List>
```

### Registry Value

```
config.Protocols.RTSP
```

## Var: PreferClientTCP

When enabled by your Helix Proxy license and set to 1, the PreferClientTCP option sets TCP as the preferred transport method for media players that indicate TCP support in their RTSP SETUP request. This variable has no effect on sessions streamed through other application-level protocols, such as MMS or HTTP.

**Note:** Carefully consider the ramifications of using this variable. Using TCP can increase data reliability for delivering short clips over broadband connections. Because the TCP transport adds significant overhead to the streaming session bandwidth, however, using it is not advisable when streaming over narrowband connections or when delivering long clips or broadcasts.

### Tag Syntax

```
<Var PreferClientTCP="Boolean"/>
```

### Possible Values

- 0 – No preferred transport. Use the first protocol listed in the RTSP SETUP request, whether UDP or TCP. This is the default.
- 1 – TCP preferred. Stream over TCP as long as TCP is listed as one of the player's supported transport options in the RTSP SETUP request. If TCP is not supported, use UDP.

### Example

```
<Var PreferClientTCP="1"/>
```

## Caching

Caching of on-demand clips is the default behavior, and is turned on automatically in Helix Proxy. The first time a media player requests a media clip from Helix Server, Helix Proxy acquires and stores that clip.

**For More Information:** For more on caching, refer to the setup chapter in *Helix Proxy Administration Guide*.

### Caching Syntax

The caching configuration syntax appears within the FSMount list:

```
<List Name="RealSystem Media Import Filesystem">
  <Var ShortName="pn-mii-mgr"/>
  <Var MountPoint="/miicache"/>
  <Var CacheShortName="rn-cache"/>
</List>
<List Name="RealSystem Streaming Import Filesystem">
  <Var ShortName="pn-sii-mgr"/>
  <Var MountPoint="/siicache"/>
  <Var CacheShortName="rn-cache"/>
</List>
<List Name="RNCache Local File System">
  <Var ShortName="pn-local"/>
  <Var MountPoint="/fsforcache"/>
  <Var BasePath="C:\Program Files\Real\RealServer\Cache"/>
</List>
...
</List>
<List Name="RNCache">
  <Var Enabled="1"/>
  <Var MaxCacheSizeMB="1000"/>
  <Var CacheMountPoint="/fsforcache"/>
</List>
```

The RealSystem Media Import Filesystem and RealSystem Streaming Import Filesystem lists define the basic operation of the caching file system. Their values should not be changed unless you are creating new plug-ins that control the caching operation. The remaining lists and variables let you change the features of the caching system.

**List: RNCache Local File System**

This list defines the basic operation of the caching file system:

```
<List Name="RNCache Local File System">
  ...file system features...
</List/>
```

**Registry Value**

```
config.FSMount.RNCache Local File System
```

**Var: ShortName**

Indicates the name of the plug-in manager.

**Tag Syntax**

```
<Var ShortName="pn-local"/>
```

**Possible Values**

- pn-local

**Example**

```
<Var ShortName="pn-local"/>
```

**Var: MountPoint**

Indicates the name of the mount point, which is a virtual path used internally to invoke the feature.

**Tag Syntax**

```
<Var MountPoint="string"/>
```

**Possible Values**

- /fsforcache/

**Example**

```
<Var MountPoint="/fsforcache"/>
```

**Var: BasePath**

This variable defines the location of cached files.

### Tag Syntax

```
<Var BasePath="string" />
```

### Possible Values

Use a path that points to the cache defined by this list.

### Example

```
<Var BasePath="C:\Program Files\Real\Helix Proxy\Cache" />
```

## List: RNCache

Use RNCache to define characteristics of the cache. Set up these lists on both the parent and child Helix Proxies.

```
<List Name="RNCache">  
  ...cache information...  
</List/>
```

### Registry Value

config.RNCache

## Var: Enabled

Use Enabled to turn on the use of cache.

### Tag Syntax

```
<Enabled="Boolean" />
```

### Possible Values

- 0 – Cache disabled.
- 1 – Cache enabled. This is the default.

### Example

```
<Var Enabled="1" />
```

## Var: MaxCacheSizeMB

The MaxCacheSizeMB variable sets the maximum size of the cache.

**Tag Syntax**

```
<Var MaxCacheSizeMB="string"/>
```

**Possible Values**

Use an integer to set the size in Megabytes.

**Example**

```
<Var MaxCacheSizeMB="1000"/>
```

**Var: CacheMountPoint**

Indicates a mount point used as an internal resource.

**Tag Syntax**

```
<Var CacheMountPoint="string"/>
```

**Possible Values**

- /fsforcache/

**Example**

```
<Var CacheMountPoint="/fsforcache"/>
```

**Automatic Media Player Redirection**

You can configure a Helix Proxy on a local network to intercept media requests to external servers. This allows you to redirect external media requests to the proxy without configuring media players to use the proxy. However, it does require the use of software or hardware, such as a Layer-4 switch, that routes TCP traffic by destination port.

**For More Information:** For details about network requirements, refer to the media player configuration chapter in *Helix Proxy Administration Guide*.

**Media Player Redirection Syntax**

The redirection method uses the Helix Proxy plug-in named `redirect.so.6.0` on UNIX or `redi3260.dll` on Windows. This plug-in ships with Helix Proxy, but is

disabled by default. To configure it, you add the following list and variables to the configuration file:

```
<List Name="RTSPRedirector">  
  <Var Port="554"/>  
  <Var RedirectToAddress="127.18.32.0"/>  
  <Var RedirectToPort="1091"/>  
</List>
```

## List: RTSPRedirector

This list configures the RTSP redirection plug-in. You must add it to the configuration file manually.

### Registry Value

config.RTSPRedirector

## Var: Port

The port that the redirector plug-in uses to listen for media player requests redirected by the network switching mechanism.

### Tag Syntax

```
<Var Port="value"/>
```

### Possible Values

This is commonly port 554. However, it can be any open port. If port 554 is not used, the switch must overwrite TCP packets in media requests to use the selected port number. If port 554 is used, Helix Proxy cannot use that port as its RTSP listen port.

### Example

```
<Var Port="554"/>
```

## Var: RedirectToAddress

Helix Proxy directs media players to connect to it using this address.

### Tag Syntax

```
<Var RedirectToAddress="IPaddress"/>
```

### Possible Values

The IPv4 or IPv6 address of Helix Proxy on your local network.

### Example

```
<Var RedirectToAddress="127.18.32.0"/>
```

## Var: RedirectToPort

The port Helix Proxy uses to listen for RTSP requests. The proxy directs media players to connect to it using this port value.

### Tag Syntax

```
<Var RedirectToPort="value"/>
```

### Possible Values

The same value as the RTSP listen port. See “Var: HTTPPort” on page 24.

**Note:** If the network switching mechanism **cannot** distinguish between requests sent to external servers on port 554 and requests sent to Helix Proxy on port 554, Helix Proxy **cannot** use the default 554 as its RTSP listen port.

### Example

```
<Var RedirectToPort="1091"/>
```

## Proxy Routing

For networks that handle Internet-bound requests with strict rules, you can configure Helix Proxy to route requests for material through other Helix Proxies, which can be useful when setting up content delivery for multiple subnets.

**Note:** RealNetworks recommends using the same version of Helix Proxy for all proxies in any routing configuration.

**For More Information:** For details, refer to the proxy routing chapter in *Helix Proxy Administration Guide*.

## Proxy Routing Syntax

The following is an example of the proxy routing syntax:

```
<List Name="ProxyRoutingTable">
  <List Name="100">
    <Var Rule="*.tokyo.example.com"/>
    <Var Description="Tokyo"/>
    <Var UseParentProxy="0"/>
    <Var ParentRTSPPort="554"/>
    <Var ParentMEIPort="7878"/>
    <Var ParentName=""/>
  </List>
  <List Name="200">
    <Var Rule="*/>
    <Var Description="All Other Locations"/>
    <Var UseParentProxy="1"/>
    <Var ParentRTSPPort="554"/>
    <Var ParentMEIPort="7878"/>
    <Var ParentName="helixproxy.example.com"/>
  </List>
</List>
```

### List: ProxyRoutingTable

Use this list to accommodate rules by which Helix Proxy routes its traffic to other Helix Proxies. Each rule is defined by a nested List: *Rule Number*.

#### Registry Value

config.ProxyRoutingTable

### List: Rule Number

Each List: *Rule Number* defines a single rule for routing proxy traffic to another proxy. The rule number, an integer, determines the order in which Helix Proxy analyzes and uses rules, with lower values examined first.

#### Registry Value

config.ProxyRoutingTable.*Rule Number*

#### Example

```
<List Name="100"> ... </List>
```

**Var: Description**

Descriptive name of the rule.

**Tag Syntax**

```
<Var Description="string"/>
```

**Possible Values**

Use a statement to differentiate this rule from other rules.

**Example**

```
<Var Description="All Other Locations"/>
```

**Var: Rule**

For a given rule, this variable defines the URL of the requested content that Helix Proxy will either handle itself by sending requests to an origin Helix Server, or by routing requests to a parent proxy.

**Tag Syntax**

```
<Var Rule="string"/>
```

**Possible Values**

The following are samples of possible values you could use. You can use an asterisk (\*) as a wildcard:

- \*.example.com
- /sports/
- example\_file.rm
- \*.example.com/sports/\*\_file.rm

**Example**

```
<Var Rule="*.real.com"/>
```

**Var: UseParentProxy**

For a given rule, use this variable to enable or disable proxy routing.

### Tag Syntax

```
<Var Description="Boolean" />
```

### Possible Values

- 0|False – Do not route matching requests received by this proxy to a parent proxy.
- 1|True – Route matching requests received by this proxy to a parent proxy.

### Example

```
<Var UseParentProxy="0" />
```

## Var: ParentMEIPort

Use this variable to define the caching port number of a parent proxy. From Helix Server, Helix Proxy uses the RTSP protocol to transfer data to its cache. However, in a given routing configuration, you could use any supported version of Helix Server or RealSystem Server (which uses MEI) as origin servers. As a result, if proxy routing is enabled in a given rule, this variable is required.

### Tag Syntax

```
<Var ParentMEIPort="value" />
```

### Possible Values

Match the parent RealSystem Proxy value for MEI Port, usually 7878.

### Example

```
<Var ParentMEIPort="7878" />
```

## Var: ParentRTSPPort

Use this variable to indicate the RTSP port number of the parent Helix Proxy. If proxy routing is enabled in a given rule, this variable is required.

### Tag Syntax

```
<Var ParentRTSPPort="value" />
```

### Possible Values

Match the parent Helix Proxy value for RTSP Port, sometimes 554.

**Example**

```
<Var ParentRTSPPort="554"/>
```

**Var: ParentName**

Use this variable to indicate the host name or the IP address of the parent Helix Proxy where client requests should be directed. If proxy routing is enabled in a given rule, this variable is required.

**Possible Values**

- DNS host name of Helix Proxy.
- IPv4 address of the Helix Proxy.
- IPv6 address of Helix Proxy.

**Example**

```
<Var ParentName="172.31.255.240"/>
```

**Redundant Proxies**

By default, when an RTSP connection between RealPlayer and Helix Proxy becomes unavailable, RealPlayer attempts to re-establish the connection to the same Helix Proxy. However, if you've designated alternate proxies, the player attempts to connect to one of the alternate proxies for delivery of the stream.

**Note:** The RealPlayer reconnect feature is supported for earlier versions of Helix Proxy, but use of ProxyAlternates is only available in Helix Proxy version 9 and later.

**For More Information:** For more on redundant proxies, refer to the setup chapter in *Helix Proxy Administration Guide*.

**Redundant Proxy Syntax**

The following example demonstrates how to specify three alternates for a Helix Proxy:

```
<List Name="ProxyAlternates">
  <List Name="Alternates">
    <List Name="Alternate_1">
      <Var Host="172.31.255.254"/>
```

```
        <Var Port="554"/>
    </List>
    <List Name="Alternate_2">
        <Var Host="main.rbn.com"/>
        <Var Port="554"/>
    </List>
    <List Name="Alternate_3">
        <Var Host="home.rbn.com"/>
        <Var Port="554"/>
    </List>
</List>
</List>
```

### List: ProxyAlternates

This is the primary list tag that is used to identify the presence of alternate proxies in the configuration file.

```
<List Name="ProxyAlternates">
    ...list of designated alternate proxies...
</List>
```

#### Registry Value

config.ProxyAlternates

### List: Alternates

The Alternates list contains a list of alternate proxies, where each alternate is itself a list.

```
<List Name="Alternates">
    ...designated alternate proxy lists...
</List>
```

#### Registry Value

config.ProxyAlternates.Alternates

### List: Alternate Name

This list defines a Helix Proxy alternate. Specify any name for the list, as long as it is unique. When a Helix Proxy becomes unavailable, RealPlayer attempts to connect to any designated alternate proxy in the list, beginning with the first alternate.

```
<List Name="Alternate Name">
  ...variables that define the alternate proxy...
</List>
```

### Registry Value

```
config.ProxyAlternates.Alternates.string
```

## Var: Host

This variable specifies the host name of the designated alternate proxy.

### Tag Syntax

```
<Var Dest="Destination" />
```

### Possible Values

- DNS host name of Helix Proxy.
- IPv4 address of the Helix Proxy.
- IPv6 address of Helix Proxy.

### Example

```
<Var Dest="205.23.5.108" />
```

## Var: Port

This variable specifies the RTSP port number through which to connect to the designated alternate proxy.

### Tag Syntax

```
<Var Port="integer" />
```

### Possible Values

- port – Data channel for RTSP connections. The default is 554.

### Example

```
<Var Port="554" />
```

## Datatype Packetization

Helix Proxy generally can stream MPEG-4 clips encoded with any codec, as long as the clips include a hint track. However, Helix Proxy can stream clips encoded with certain codecs whether or not the clips contain hint tracks. For these codecs, Helix Proxy refers to the hint track if it is present, but still streams the media packets if the track is absent. Through the configuration file, you can turn off reading of a hint track entirely.

**Tip:** Unless the presence of hint tracks causes streaming problems, it's generally better to leave the default values, which cause Helix Proxy to use the hint track if it is present. Streaming without reference to the hint track is less efficient.

### Datatypes Packetization Syntax

The following sample shows the syntax used to determine which MPEG codecs use on-the-fly packetization at all times:

```
<List Name="Datatypes">
  <Var DefaultMaxPacketSize="1400"/>
  <List Name="MIME type">
    <Var ForcePacketization="1"/>
  </List>
  ...other datatypes for forced packetization...
</List>
```

### List: Datatypes

The Datatypes list indicates the MPEG codecs for which forced packetization can be used:

```
<List Name="Datatypes">
  ... datatypes for forced packetization...
</List>
```

#### Registry Value

config.Datatypes

## Var: DefaultMaxPacketSize

The DefaultMaxPacketSize variable defines the maximum size in bytes for each media packet that uses forced packetization. Otherwise, the clip's hint track determines the packet size..

### Tag Syntax

```
<Var DefaultMaxPacketSize="bytes"/>
```

### Possible Values

Size in bytes.

### Example

```
<Var DefaultMaxPacketSize="1400"/>
```

## List: MIME\_Types

The MIME types list specifies the MIME type of the media stream produced by the codec for which forced packetization can be used:

```
<List Name="MIME type">
  ...packetization variable...
</List>
```

### Possible Values

The list name uses the MIME type for the MPEG-4 datatype. Possible values include the following:

- video/mp4v-es
- video/h263-2000
- audio/mp4a-latm
- audio/amr
- audio/amr-wb

### Registry Value

config.Datatypes

### Example

```
<List Name="video/mp4v-es">  
  <Var ForcePacketization="1"/>  
</List>
```

## Var: ForcePacketization

The ForcePacketization variable determines if any hint track is ignored and packetization is forced instead.

### Tag Syntax

```
<Var ForcePacketization="Boolean"/>
```

### Possible Values

- 0|False – Use on-the-fly packetization only if hint track is missing.
- 1|True – Force packetization in all cases.

### Example

```
<Var ForcePacketization="False"/>
```

## Differentiated Services

The differentiated services feature allows you to assign priorities to the IP packets that carry streaming media data. 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 Proxy 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>. See also the setup chapter of *Helix Proxy Administration Guide*.

## Differentiated Services Syntax

Differentiated services operates by assigning bit values to the DSFIELD field in IP headers. The Helix Proxy configuration file holds a decimal value that corresponds to the various, possible binary values of DSFIELD, as illustrated in the following sample:

```
<List Name="DiffServ">
  <Var Control="12"/>
  <Var Media="16"/>
</List>
```

## Differentiated Services Decimal Values

The following table lists the decimal values set in the configuration file to define differentiated services. If a protocol is set to the CRITIC/ECP precedence, for example, the base value is 160. To that you add 16 to set Delay to Low, 8 to set Throughput to High, and 4 to set Reliability to Minimum. For instance, the decimal value used for the CRITIC/ECP precedence if Throughput is High while the other two values are set to Normal is 168.

**Differentiated Services Decimal Value Settings**

Precedence	Base Value	Delay	Throughput	Reliability
Network Control	224	Normal (+0) Low (+16)	Normal (+0) High (+8)	Normal (+0) Minimum (+4)
Internetwork Control	192	Normal (+0) Low (+16)	Normal (+0) High (+8)	Normal (+0) Minimum (+4)
CRITIC/ECP	160	Normal (+0) Low (+16)	Normal (+0) High (+8)	Normal (+0) Minimum (+4)
Flash Override	128	Normal (+0) Low (+16)	Normal (+0) High (+8)	Normal (+0) Minimum (+4)
Flash	96	Normal (+0) Low (+16)	Normal (+0) High (+8)	Normal (+0) Minimum (+4)
Immediate	64	Normal (+0) Low (+16)	Normal (+0) High (+8)	Normal (+0) Minimum (+4)
Priority	32	Normal (+0) Low (+16)	Normal (+0) High (+8)	Normal (+0) Minimum (+4)
Routine	0	Normal (+0) Low (+16)	Normal (+0) High (+8)	Normal (+0) Minimum (+4)

## List: DiffServ

This list defines the differentiated services variables:

```
<List Name="DiffServ">  
    ...differentiated services variable values defined here...  
</List>
```

### Registry Value

config.DiffServ

## Var: Control

Sets the differentiated services binary value to use in IP headers for RTSP control protocol packets.

### Tag Syntax

```
<Var Control="Integer"/>
```

### Possible Values

A decimal representation of a binary value as described in “Differentiated Services Decimal Values” on page 63.

### Example

```
<Var Control="168"/>
```

## Var: Media

Sets the differentiated services binary value to use in IP headers for packets of the RDT and RTP formats.

### Tag Syntax

```
<Var Control="Integer"/>
```

### Possible Values

A decimal representation of a binary value as described in “Differentiated Services Decimal Values” on page 63.

### Example

```
<Var Media="76"/>
```





## RATE CONTROL

This chapter covers methods for adjusting the streaming rate for media players based on the player's capabilities.

## Capabilities Exchange

The capabilities exchange feature enables Helix Proxy to learn the streaming media features of a media player. For example, Helix Proxy can determine the media player's buffering capacity. This, in turn, helps it to use the rate control feature more effectively.

**For More Information:** For more on capabilities exchange, refer to the setup chapter in *Helix Proxy Administration Guide*.

**Tip:** Helix Proxy needs to contact HTTP servers to obtain media player RDF files. To enable this, configure your firewall to allow outgoing TCP communication to port 80 on external HTTP servers.

## Capabilities Exchange Syntax

Within the FSMount section, the capabilities exchange feature defines a mount point that defines the directory that holds the media player RDF files:

```
<List Name="CapExProfiles">  
  <Var ShortName="pn-local"/>  
  <Var MountPoint="/profiles"/>  
  <Var BasePath="/usr/local/Real/HelixProxy/ClientProfiles"/>  
</List>
```

The capabilities exchange profiles for each media player are defined in a top-level list elsewhere in the configuration file. The following provides an example of the capabilities exchange syntax defined for a single media player:

```
<List Name="CapabilityExchange">
  <Var ProfileCacheSize="100"/>
  <Var MaxProfileSize="64"/>
  <List Name="DefaultProfiles">
    <List Name="Client1">
      <Var UserAgent="RealOnePlayer/3.2.23_epoc_series60_thumb"/>
      <Var URI="file://profiles/Nokia3650.rdf"/>
      <Var IgnoreCapExHeaders="0"/>
    </List>
    ...other client profiles...
  </List>
</List>
```

### List: CapExProfiles

This list falls within the FSMount section and defines the mount point used to access capabilities exchange profiles:

```
<List Name="CapExProfiles">
  ...capabilities exchange mount point variables...
</List>
```

#### Registry Values

config.FSMount.CapExProfiles

### Var: ShortName

Short name for the capabilities exchange file system.

#### Tag Syntax

```
<Var ShortName="Name"/>
```

#### Possible Values

- pn-local – Standard plug-in for local file access.

#### Example

```
<Var ShortName="pn-local"/>
```

### Var: MountPoint

The MountPoint variable is used in URLs to allow Helix Proxy to access the RDF profiles.

**Tag Syntax**

```
<Var MountPoint="Mount Point"/>
```

**Possible Values**

Any mount point name. The default is `/profiles/`.

**Example**

```
<Var MountPoint="/profiles/" />
```

**Var: BasePath**

The `BasePath` variable provides the path to the directory that holds the RDF files.

**Tag Syntax**

```
<Var BasePath="path"/>
```

**Possible Values**

Any path. The default is the `ClientProfiles` directory under the main Helix Proxy installation directory.

**Example**

```
<Var BasePath="/usr/local/Real/HelixProxy/ClientProfiles"/>
```

**List: CapabilityExchange**

This top-level list defines the capabilities exchange feature:

```
<List Name="CapabilityExchange">
  ...capabilities exchange variables...
</List>
```

**Registry Values**

```
config.CapabilityExchange
```

**Var: ProfileCacheSize**

Sets the number of RDF file profiles that Helix Proxy caches in its registry.

### Tag Syntax

```
<Var ProfileCacheSize="integer"/>
```

### Possible Values

Number of profiles to cache.

### Example

```
<Var ProfileCacheSize="100"/>
```

## Var: MaxProfileSize

Sets the maximum size of each RDF file.

### Tag Syntax

```
<Var MaxProfileSize="integer"/>
```

### Possible Values

A size in Kilobytes. The default is 128.

### Example

```
<Var MaxProfileSize="64"/>
```

## List: DefaultProfiles

This list holds other lists that define the capabilities exchange characteristics for each media player:

```
<List Name="DefaultProfiles">  
  ...media player capabilities defined here...  
</List>
```

### Registry Value

```
config.MediaDelivery.UserAgentProfiles
```

## List: Client\_n

Within the DefaultProfiles list, a separate list holds the profile information for each media player. Client lists are identified in numeric order (Client1, Client2, and so on), but each list can have any user-defined name that identifies the client. The final list is named Default, and applies to any media player that does not have client capabilities defined for it.

## Registry Value

config.MediaDelivery.UserAgentProfiles.*Client*

## Example

```

<List Name="Client1">
  <Var UserAgent="RealOnePlayer/3.2.23_epoc_series60_thumb"/>
  <Var URI="file://profiles/Nokia3650.rdf"/>
  <Var IgnoreCapExHeaders="0"/>
</List>

```

## Var: UserAgent

The UserAgent variable uniquely identifies the media player. This string must also match the UserAgent variable used with the rate control feature.

**For More Information:** For information on the variable used with rate control, see “Var: UserAgent” on page 79.

## Tag Syntax

```
<Var UserAgent="ID"/>
```

## Possible Values

String ID reported by the media player to Helix Proxy, typically in an RTSP DESCRIBE message.

## Example

```
<Var UserAgent="RealOnePlayer/3.2.23_epoc_series60_thumb"/>
```

## Var: URI

Specifies the location of the RDF file for the media player.

## Tag Syntax

```
<Var URI="URI"/>
```

## Possible Values

Local URI for the RDF file. Files are typically accessed through the profiles/ mount point, which corresponds by default to the ClientProfiles directory under the main Helix Proxy installation directory.

### Example

```
<Var URI="file://profiles/Nokia3650.rdf"/>
```

## Var: IgnoreCapExHeaders

Determines whether to use the values from the local, cached RDF file, or values from the URI sent by the media player at the start of the session.

### Tag Syntax

```
<Var IgnoreCapExHeaders="Boolean"/>
```

### Possible Values

- 0 – Always use the locally stored RDF file values.
- 1 – Use the RDF file URI sent at the session start-up.

### Example

```
<Var IgnoreCapExHeaders="1"/>
```

## Rate Control

The server-side rate control feature allows Helix Proxy to adjust the bit rate of a prerecorded clip streamed to media players based on periodic status reports from the player. These reports allow Helix Proxy to compensate for fluctuating bandwidth that may occur due to network congestion. Rate control does not function with live broadcasts.

**Note:** You can set a generic rate control profile through Helix Administrator, as described in the server set-up chapter of *Helix Proxy Administration Guide*. To set profiles for specific media players, though, you must edit the configuration file manually.

### Rate Control Syntax

The variable values selected for each media player result from extensive testing. Many variables are interrelated, meaning that a change in one variable value may require a change in other values. **RealNetworks recommends that you do not change any values in the existing lists, or add any new lists, without first consulting a RealNetworks technical representative.**

The following provides an example of the rate control syntax defined for a single media player:

```

<List Name="MediaDelivery">
  <List Name="UserAgentProfiles">
    ...additional media player profiles...
  <List Name="PlayerName">
    <Var UserAgent="UserAgentString"/>
    <Var UseMediaDeliveryPipeline="1"/>
    <List Name="Transport">
      <Var RtcpRRratio="200"/>
      <Var RtcpRSratio="100"/>
      <List Name="CongestionControl">
        <Var Type="TFRC"/>
        <Var MaxBurst="3"/>
        <Var PassThrough="0"/>
        <Var InitialRate="10000"/>
        <Var ChannelRate="132000"/>
        <Var MaxRateScalar="200"/>
        <List Name="Timeout">
          <Var InitialTimeout="3000"/>
          <Var Interval="50"/>
          <Var MaxTimeouts="128"/>
          <Var Disable="0"/>
        </List>
      <List Name="TFRC">
        <Var RTTUseIIR="0"/>
        <Var TimeoutTransmission="0"/>
        <Var TimeoutScalar="50"/>
        <Var UseSlowStart="1"/>
        <Var StartRate="16000"/>
        <Var SlowStartScalar="300"/>
        <Var Trace="0"/>
      </List>
    <List Name="BCC">
      <Var RTTUseIIR="0"/>
      <Var IncreaseCoefficient="512000"/>
      <Var DecreaseCoefficient="8"/>
      <Var IncreaseExponent="0.5"/>
      <Var DecreaseExponent="0.5"/>
      <Var IncreaseThreshold="0"/>
      <Var DecreaseThreshold="5"/>
      <Var LowerLimit="2048"/>
      <Var TargetRatio="4"/>
    </List>
  </List>
</List>

```

```

        <Var TimeoutTransmission="0"/>
        <Var Trace="0"/>
    </List>
    <List Name="TCP">
        <Var LimitRate="1"/>
    </List>
</List>
</List>
<List Name="Session">
    <List Name="RateManager">
        <Var BufferModel="ANNEXG"/>
        <Var Type="ANNEXG"/>
        <Var MultiStreamVerifier="AGGREGATE"/>
        <Var ClientBufferLowWatermark="20"/>
        <Var ClientBufferHighWatermark="120"/>
        <Var TargetTimeLowWatermark="125"/>
        <Var TargetTimeHighWatermark="250"/>
        <Var ResendTimeLimit="1000"/>
        <Var ClientBufferPeriod="7000"/>
        <Var Trace="0"/>
    </List>
</List>
<List Name="InputSource">
    <Var InitialMediaRate="32000"/>
    <Var MinPreroll="3000"/>
    <Var NetworkAffinity="5000"/>
    <List Name="StaticPushSource">
        <Var AllowInitialExternalSubscription="0"/>
    </List>
</List>
</List>
</List>
</List>

```

## List: MediaDelivery

This is the master list that defines rate control.

```

<List Name="MediaDelivery">
    ...all rate control variables defined here...
</List>

```

### Registry Value

```
config.MediaDelivery
```

## Var: ConfigHelixRAEnabled

The variable ConfigHelixRAEnabled turns server-side rate control off for newer, Helix-based players. It affects these players only when they use the proprietary RDT packet format. If rate control is disabled, these clients can still use client-side stream shifting with SureStream clips.

### Tag Syntax

```
<Var ConfigHelixRAEnabled="Boolean"/>
```

### Possible Values

- 0|false – Helix Proxy does not use server-side rate control. The player may still use client-side rate control with SureStream clips.
- 1|true – Helix Proxy uses server-side rate control with multirate clips. This is the default.

### Example

```
<Var ConfigHelixRAEnabled="0"/>
```

## List: UserAgentProfiles

This list encapsulates the lists that define rate control for individual media players.

```
<List Name="UserAgentProfiles">
  ...media player rate control lists defined here...
</List>
```

### Registry Value

```
config.MediaDelivery.UserAgentProfiles
```

## List: MediaPlayerName

This list provides a user-defined name for the media player.

```
<List Name="MediaPlayerName">
  ...rate control for this media player defined here...
</List>
```

### Registry Value

```
config.MediaDelivery.UserAgentProfiles.MediaPlayerName
```

## Var: UserAgent

The UserAgent variable uniquely identifies the media player. This string must also match the UserAgent variable used with capabilities exchange.

**For More Information:** For information on the variable used with capabilities exchange, see “Var: UserAgent” on page 71.

### Tag Syntax

```
<Var UserAgent="ID"/>
```

### Possible Values

String ID reported by the media player to Helix Proxy, typically in an RTSP DESCRIBE message.

### Example

```
<Var UserAgent="RealMedia Player/mc"/>
```

## Var: UseMediaDeliveryPipeline

This variable determines whether Helix Proxy uses its server-side rate control feature for a media player that does not signal in its session headers that it supports rate control. If a client signals that it supports rate control, Helix Proxy uses rate control regardless of this variable’s setting. The following table summarizes the possible cases.

**Conditions for Using Rate Control**

Client Signals Rate Control Support	UseMediaDeliveryPipeline=0	UseMediaDeliveryPipeline=1
Yes	Use rate control.	Use rate control.
No	Do not use rate control.	Use rate control.

### Tag Syntax

```
<Var UseMediaDeliveryPipeline="Boolean"/>
```

### Possible Values

- 0 – Rate control disabled if the client does not signal support for rate control. The media player therefore receives a stream at a constant bit rate.

RealNetworks media players can use client-side stream adaptation if SureStream clips are used. This is the default.

- 1 – Rate control always enabled. Helix Proxy can use server-side rate control to adjust the streaming rate for a multi-rate clip.

**Note:** Helix Proxy can still maintain a buffer model for a client that does not support rate control, but the model will be less accurate than if the client supported rate control.

#### Example

```
<Var UseMediaDeliveryPipeline="1"/>
```

### Var: InactivityTimeout

The InactivityTimeout variable determines how long Helix Server keeps a connection open when there is no activity from the client. Once this time elapses, Helix Server tears down the connection.

#### Tag Syntax

```
<Var InactivityTimeout="Integer"/>
```

#### Possible Values

An integer value that represents the number of seconds. The default is 0, which means no teardown occurs.

#### Example

```
<Var InactivityTimeout="1000"/>
```

### List: Transport

The variables in the <Transport> list affect the bandwidth parameters used to vary the transmission rate for a streaming session.

```
<List Name="Transport">
  ...transport variables...
</List>
```

#### Registry Value

```
config.MediaDelivery.UserAgentProfiles.MediaPlayerName.Transport
```

## Var: RtcpRRratio

If the media player implements the b=RR method in its SDP to determine the frequency of RTCP receiver reports, the variable RtcpRRratio indicates the percentage of the initial media rate for a given stream used by the player for RTCP receiver report transmission. Not all media players honor this attribute, however.

**Note:** The bandwidth used for receiver report transmission does not change if the media rate shifts up or down.

### Tag Syntax

```
<Var RtcpRRratio="Integer"/>
```

### Possible Values

The value is a percentage scaled by 100. The default value of 200 therefore means that 2 percent of bandwidth is reserved for receiver report transmission. Choose a value that, given the initial media rate, transmits the RTCP receiver reports at an interval that approximates the expected round-trip time (RTT) of the link.

**Tip:** Helix Proxy records as client properties the average receiver report frequency and round-trip time. You can use these values to determine actual client performance and set RtcpRRratio more accurately. For more information, refer to “RRFrequency” on page 206 and “RTT” on page 206.

### Example

```
<Var RtcpRRratio="300"/>
```

## Var: RtcpRSratio

The variable RtcpRSratio indicates the percentage of the initial media rate for a given stream used by the player for RTCP sender report transmission. Not all media players honor this attribute, however.

### Tag Syntax

```
<Var RtcpRSratio="Integer"/>
```

### Possible Values

The value is a percentage scaled by 100. Therefore, the default value of 100 means that 1 percent of bandwidth reserved for sender report transmission.

### Example

```
<Var RtcpRSratio="100"/>
```

## List: CongestionControl

This list defines the Helix Proxy congestion control operation for a media player. Variables and sublists within this list determine if the TFRC or BCC congestion control mechanism is used:

```
<List Name="CongestionControl">
  <Var Type="TFRC"/>
  <Var MaxBurst="3"/>
  <Var PassThrough="0"/>
  <Var InitialRate="10000"/>
  <Var ChannelRate="132000"/>
  <Var MaxRateScalar="200"/>
  ...additional congestion control sublists...
</List>
```

### Registry Value

```
config.MediaDelivery.UserAgentProfiles.MediaPlayerName.CongestionControl
```

## Var: Type

Determines the type of congestion control mechanism used for this media player.

### Tag Syntax

```
<Var Type="TFRC|BCC"/>
```

### Possible Values

- BCC

Binomial Congestion Control (BCC) insures that correct congestion control response is maintained while not causing the RTT to increase to excessive levels. BCC should only be used on networks with large buffers along the link path. The section “List: BCC” on page 91 explains the BCC variables.

**Note:** Do not choose BCC without first performing diagnostics on the network to determine if this type of congestion control is appropriate.

- TFRC

TCP Friendly Rate Control (TFRC) is more commonly used than Binomial Congestion Control. It poses a problem, however, for some networks that contain network buffers that exceed 1 MB in size. On such networks, TFRC may cause large round trip times, negatively affecting throughput. The section “List: TFRC” on page 85 explains the TFRC variables.

#### Example

```
<Var Type="TFRC"/>
```

### Var: MaxBurst

Sets the number of packets to send in a single scheduling window.

#### Tag Syntax

```
<Var MaxBurst="Integer"/>
```

#### Possible Values

An integer in the range 0 to 1024. With the default value of 3, for example, the scheduling window for a 20 Kbps clip is approximately 6.6 seconds if the packet size is 1 Kb.

#### Example

```
<Var MaxBurst="2"/>
```

### Var: PassThrough

Determines if congestion control is used.

#### Tag Syntax

```
<Var PassThrough="Boolean"/>
```

#### Possible Values

- 0 – Use congestion control. This is the default.

- 1 – Disable congestion control.

#### Example

```
<Var PassThrough="0"/>
```

### Var: InitialRate

Sets the initial rate in bits per second that Helix Proxy streams media. This is used only if the average bit rate is not specified in the clip's file header. When the InitialRate value is used, it is observed only for session startup. After that, the rate may change based on the Helix Proxy congestion control model.

#### Tag Syntax

```
<Var InitialRate="Integer"/>
```

#### Possible Values

Any positive integer indicating a rate in bits per second.

#### Example

```
<Var InitialRate="10000"/>
```

### Var: ChannelRate

The top streaming bit rate for the network channel in bits per second. You can set this arbitrarily high, but Helix Proxy may periodically overload the channel rate causing packet loss.

#### Tag Syntax

```
<Var ChannelRate="Integer"/>
```

#### Possible Values

Any positive integer indicating a maximum rate in bits per second. The default is 10000000 (10 million) bits per second.

#### Example

```
<Var ChannelRate="15000000"/>
```

## Var: MaxRateScalar

MaxRateScalar sets the maximum streaming rate increase based on the clip's current streaming rate. This value determines the ability of Helix Proxy to upshift to a faster rate in a multi-rate clip. Suppose that a clip begins to stream at 12 Kbps, and that MaxRateScalar is set to 200. For a clip encoded with 12, 24, and 36 Kbps streams, the media player can upshift through all values, because no rate exceeds the preceding rate by more than 200 percent. However, if the clip contains just 12 and 36 Kbps streams, the player cannot upshift from the 12 Kbps stream.

**Note:** The MaxRateScalar variable sets a maximum rate based on the current stream speed. The ChannelRate variable sets a maximum rate based on the network's capacity. The lesser of the two sets the actual maximum rate.

### Tag Syntax

```
<Var MaxRateScalar="Percentage"/>
```

### Possible Values

An integer greater than 100 that expresses a percentage of the stream rate. A value of 300, for example, sets the maximum transmission rate at 300 percent of the stream's speed. The default value is 500.

### Example

```
<Var MaxRateScalar="300"/>
```

## Var: UseClientRateReq

This variable can be used only with RealNetworks media players and generally should not be changed. It instructs Helix Proxy to respect the RTSP extension SetDeliveryBandwidth. If Helix Proxy respects these legacy requests, the other congestion control variables are ignored.

### Tag Syntax

```
<Var UseClientRateReq="Boolean"/>
```

### Possible Values

- 0 – Ignore SetDeliveryBandwidth and use congestion control. This is the default.

- 1 – Listen to SetDeliveryBandwidth and disable congestion control.

### Example

```
<Var UseClientRateReq="1"/>
```

### List: Timeout

The variables in the <Timeout> list affect how long Helix Proxy waits for media player receiver reports. The settings in this list define the point at which Helix Proxy determines that a media player has timed out. The actions that occur on a timeout depend on the settings of timeout variables in the TFRC or BCC list, depending on which congestion control mechanism is used for rate control.

```
<List Name="Timeout">
  <Var InitialTimeout="3000"/>
  <Var Interval="50"/>
  <Var MaxTimeouts="128"/>
  <Var Disable="0"/>
</List>
```

### Registry Value

```
config.MediaDelivery.UserAgentProfiles.MediaPlayerName.CongestionControl.Timeout
```

### Var: InitialTimeout

Sets the time in milliseconds that Helix Proxy waits for the first receiver report after it begins to stream a clip. If this time elapses without a receiver report, Helix Proxy disables congestion control for this session and continues to stream the clip at a constant rate.

### Tag Syntax

```
<Var InitialTimeout="Integer" />
```

### Possible Values

A positive integer representing the number of milliseconds.

### Example

```
<Var InitialTimeout="3000"/>
```

## Var: Interval

The Interval variable determines how many feedback intervals (intervals between RTCP receiver reports) can elapse without a receiver report before Helix Proxy declares a timeout.

### Tag Syntax

```
<Var Interval="Integer"/>
```

### Possible Values

Any positive integer that indicates the number of RTTs to wait for a receiver report before declaring a timeout. The default is 8.

### Example

```
<Var Interval="6"/>
```

## Var: MaxTimeouts

Sets the number of consecutive timeouts that can occur before Helix Proxy acts on the timeout condition. The action depends on the congestion control mechanism (TFRC or BCC) that is used.

**For More Information:** See “List: TFRC” on page 85

### Tag Syntax

```
<Var MaxTimeouts="Integer"/>
```

### Possible Values

Integer representing the maximum number of consecutive timeouts. The default is 2.

### Example

```
<Var MaxTimeouts="4"/>
```

## Var: Disable

Determines whether to disable congestion control if timeout events occur.

### Tag Syntax

```
<Var Disable="Boolean"/>
```

### Possible Values

- 0 – Do not allow congestion control to stop based on the occurrence of timeouts.
- 1 – Disable congestion control when the timeout limits have been reached. This is the default. How rate control recovers from these timeouts depends on whether the TFRC or BCC mechanism is used.

**For More Information:** For timeout actions taken when TFRC is used, refer to “Var: TimeoutTransmission” on page 86. The section “Var: TimeoutTransmission” on page 95 defines the timeout actions taken when BCC is used.

### Example

```
<Var Disable="0"/>
```

### List: TFRC

Variables in the <TFRC> list affect congestion control when TCP Friendly Rate Control is used. The timeout variables determine how streaming rate recovery occurs if Helix Proxy disables congestion control due to the occurrence of timeouts.

```
<List Name="TFRC">
  <Var RTTUseIIR="0"/>
  <Var TimeoutTransmission="0"/>
  <Var TimeoutScalar="50"/>
  <Var UseSlowStart="1"/>
  <Var StartRate="16000"/>
  <Var SlowStartScalar="300"/>
  <Var Trace="1"/>
</List>
```

**Note:** The Type variable determines if the TFRC or BCC congestion control mechanism is used. See “Var: Type” on page 79.

**For More Information:** For more on TFRC, refer to IETF RFC 3448 at <http://www.ietf.org/rfc/rfc3448.txt>.

### Registry Value

```
config.MediaDelivery.UserAgentProfiles.MediaPlayerName.CongestionControl.TFRC
```

## Var: RTTUseIIR

Determines whether to use the Infinite Impulse Response (IIR) filter for round-trip time (RTT) calculations. If the media player uses the b=RR method in SDP, RTTUseIIR should be set to 0. Setting a value of 1 trades smoothness for responsiveness.

**Tip:** The value for this variable should typically not be changed.

### Tag Syntax

```
<Var RTTUseIIR="Boolean"/>
```

### Possible Values

- 0 – Do not use the IIR filter in RTT calculations.
- 1 – Use the IIR filter.

### Example

```
<Var RTTUseIIR="0"/>
```

## Var: TimeoutTransmission

The variables for List: Timeout (see page 83) determine when Helix Proxy declares that a media player has timed out. When TFRC is used, the TimeoutTransmission variable determines the actions that Helix Proxy takes when a timeout occurs. Helix Proxy may stop sending packets or scale down its transmission rate.

### Tag Syntax

```
<Var TimeoutTransmission="Boolean"/>
```

### Possible Values

- 0 – On each timeout event, Helix Proxy scales down its transmission rate until it receives a receiver report. In this case, the other variables in this list determine how much the streaming rate scales down on each timeout, and how quickly the rate scales up once the media player begins to return receiver reports.
- 1 – Helix Proxy stops sending packets when a timeout occurs. This is the default. The streaming session remains active as Helix Proxy waits for the media player to terminate the session or issue receiver reports. If

streaming resumes, Helix Proxy streams media at the same rate used before the timeout occurred. This resumption of data streaming, which usually occurs when the media player's buffer empties, typically results in media player rebuffering.

#### Example

```
<Var TimeoutTransmission="0"/>
```

### Var: TimeoutScalar

This variable sets the rate by which Helix Proxy scales down the streaming rate on each timeout event as long as `TimeoutTransmission` is set to 0. Once the media player returns a receiver report, the rate downscaling stops and the media rate begins to scale up based on the settings of additional variables in this list.

#### Tag Syntax

```
<Var TimeoutScalar="Integer"/>
```

#### Possible Values

An integer representing the percentage by which to scale down the streaming rate on each timeout event. The default is 50, meaning a 50 percent reduction.

#### Example

```
<Var TimeoutScalar="30"/>
```

### Var: UseSlowStart

`UseSlowStart` determines whether to scale the streaming rate up slowly or all at once during session start-up, as well as after timeouts have stopped (if `TimeoutScalar` is used).

#### Tag Syntax

```
<Var UseSlowStart="Boolean"/>
```

#### Possible Values

- 0 – Slow start disabled. On session start-up, Helix Proxy streams data at the average bit rate set in the clip's file header, or that determined by the `InitialRate` variable. After a timeout, Helix Proxy resumes the streaming

rate used before the timeouts occurred once it receives a receiver report from the media player. It immediately reinstates normal operation of congestion control based on the TFRC mechanism.

**For More Information:** See “Var: InitialRate” on page 81.

- 1 – Slow start enabled. This is the default. On session start-up and once a media player issues a receiver report following a timeout, Helix Proxy sets the streaming rate to the StartRate value, then increases the rate by the SlowStartScalar percentage each time another receiver report is received. Normal operation of congestion control based on the TFRC mechanism occurs only after data loss is detected.

#### Example

```
<Var UseSlowStart="1"/>
```

### Var: StartRate

If slow start is enabled, this variable sets the streaming rate in bits per second that Helix Proxy uses once it receives the first receiver report. After that, the rate scales up by the SlowStartScalar percentage on each subsequent receiver report.

#### Tag Syntax

```
<Var StartRate="Integer"/>
```

#### Possible Values

A positive integer representing bits per second.

#### Example

```
<Var StartRate="16000"/>
```

### Var: SlowStartScalar

If slow start is enabled, this variable sets the rate by which Helix Proxy scales up the streaming rate from the StartRate value on each receiver report after the first. Once the receiver reports denote data loss, normal operation of congestion control based on the TFRC mechanism resumes.

### Tag Syntax

```
<Var SlowStartScalar="Integer"/>
```

### Possible Values

An integer divisible by 100 in the range from 100 to 1000. A value of 300, for example, means to scale the media rate up by approximately a third of its value each time a receiver report is received. The default is 200.

### Example

```
<Var SlowStartScalar="300"/>
```

## Var: Trace

When set to 1, this variable causes Helix Proxy to record diagnostic statistics about each media player session that uses TFRC.

### Tag Syntax

```
<Var Trace="Boolean"/>
```

### Possible Values

- 0 – Do not log diagnostic statistics.
- 1 – Log diagnostic statistics.

### Example

```
<Var Trace="1"/>
```

### Trace File Syntax

When Trace is set to 1, Helix Proxy writes to its main installation directory a diagnostic log for each stream in each TFRC session. The diagnostic log is uniquely named using the following convention:

```
mdp_tfrc_session_ID_stream_ID.txt
```

Here, *session\_ID* is the RTSP session ID string, described in the section “SessionID” on page 302. The value *stream\_ID* is the unique, numeric stream identifier, with 0 representing video and 1 denoting audio.

Each diagnostic log file contains a header that indicates the current TFRC configuration parameters, as shown in the following example:

RTT IIR Filter: Not Used  
SlowStart Scalar: 2.0000  
SlowStart Rate: 16000  
Timeout Scalar: 0.7500  
Timeout Transmit: 0  
Rate Increase Limit Scalar: 1.2500

Following the header, the report records a new entry each time a receiver report arrives:

```
timestamp value media_rate  
timestamp value tfrc_rate  
timestamp value recvd_rate  
timestamp value state  
timestamp value num_lost  
timestamp value num_recv  
timestamp value loss  
timestamp value rtt  
timestamp value raw  
timestamp value sq_mean
```

Each line in a block of statistics begins with the same timestamp, which records the number of milliseconds between the start of the streaming session and the generation of the receiver report. Each line within a statistics block records a different value:

<code>media_rate</code>	Stream's encoded rate. Media rate values within a report will differ if streaming upshifting or downshifting occurred.
<code>tfrc_rate</code>	Tate that Helix Proxy transmitted the data.
<code>recvd_rate</code>	Rate at which the client received data according to Helix Proxy modeling.
<code>state</code>	Congestion control state: 0 is slow start, 1 is steady state.
<code>num_lost</code>	Number of packets lost in the last reporting interval.
<code>num_recv</code>	Number of packets received in the last reporting interval.
<code>loss</code>	TFRC loss event ratio.
<code>rtt</code>	Statistically smoothed RTT.
<code>raw</code>	RTT-derived sample from the current RTCP receiver report.
<code>sq_mean</code>	Square root of the smoothed RTT.

## List: BCC

Variables in the BCC list affect congestion control when Binomial Congestion Control (BCC) is used. The following example illustrates the variables used with BCC:

```
<List Name="BCC">
  <Var RTTUseIIR="0"/>
  <Var IncreaseCoefficient="512000"/>
  <Var DecreaseCoefficient="8"/>
  <Var IncreaseExponent="0.5"/>
  <Var DecreaseExponent="0.5"/>
  <Var IncreaseThreshold="0"/>
  <Var DecreaseThreshold="5"/>
  <Var LowerLimit="2048"/>
  <Var TargetRatio="4"/>
  <Var TimeoutTransmission="0"/>
  <Var Trace="1"/>
</List>
```

**Tip:** In general, you should not need to modify the settings for the BCC coefficients, exponents, or RTT filter. These parameters should be changed only by trained operators and RealNetworks staff to allow diagnostic field tests.

**Note:** The Type variable determines if the TFRC or BCC congestion control mechanism is used. See “Var: Type” on page 79.

**For More Information:** You can learn more about BCC at <http://nms.lcs.mit.edu/papers/binomial-infocom01.html>.

### Registry Value

```
config.MediaDelivery.UserAgentProfiles.MediaPlayerName.CongestionControl.BCC
```

## Var: RTTUseIIR

Determines whether to use the Infinite Impulse Response (IIR) filter for round-trip time (RTT) calculations. If the media player uses the b=RR method in SDP, RTTUseIIR should be set to 0. Setting a value of 1 trades smoothness for responsiveness.

### Tag Syntax

```
<Var RTTUseIIR="Boolean"/>
```

### Possible Values

- 0 – Do not use the IIR filter in RTT calculations.
- 1 – Use the IIR filter.

### Example

```
<Var RTTUseIIR="0"/>
```

## Var: IncreaseCoefficient

This variable sets an integer value that is used as a numerator in the rate increase function of the BCC algorithm.

### Tag Syntax

```
<Var IncreaseCoefficient="Integer"/>
```

### Possible Values

An integer value from 64000 to 512000. The default is 512000.

### Example

```
<Var IncreaseCoefficient="512000"/>
```

## Var: DecreaseCoefficient

This variable defines an integer value that is used as a scalar in the rate decrease function of the BCC algorithm.

### Tag Syntax

```
<Var DecreaseCoefficient="Integer"/>
```

### Possible Values

An integer from 2 to 16. The default is 8.

### Example

```
<Var DecreaseCoefficient="8"/>
```

## Var: IncreaseExponent

This variable sets a floating point value that is used as an exponent in the denominator of the rate increase function in the BCC algorithm.

### Tag Syntax

```
<Var IncreaseExponent="FloatingPoint"/>
```

### Possible Values

A decimal value between 0 and 1. The default is 0.5.

**Note:** The IncreaseExponent value plus the DecreaseExponent value must not be greater than 1.

### Example

```
<Var IncreaseExponent="0.5"/>
```

## Var: DecreaseExponent

This variable defines a floating point value that is used as an exponent in the rate decrease function of the BCC algorithm.

### Tag Syntax

```
<Var DecreaseExponent="FloatingPoint"/>
```

### Possible Values

A decimal value between 0 and 1. The default is 0.5.

### Example

```
<Var DecreaseExponent="0.5"/>
```

**Note:** The IncreaseExponent value plus the DecreaseExponent value must not be greater than 1.

## Var: IncreaseThreshold

This variable sets an integer value that represents a time in milliseconds of change in RTT beyond which the increase function will be applied to the rate.

### Tag Syntax

```
<Var IncreaseThreshold="Integer"/>
```

### Possible Values

Any positive integer starting with 0. The default is 0.

### Example

```
<Var IncreaseThreshold="0"/>
```

## Var: DecreaseThreshold

This variable sets an integer value that represents a time in milliseconds of change in RTT beyond which the decrease function will be applied to the rate.

### Tag Syntax

```
<Var DecreaseThreshold="Integer"/>
```

### Possible Values

Any positive integer starting with 1. The default is 5.

### Example

```
<Var DecreaseThreshold="5"/>
```

## Var: LowerLimit

The LowerLimit variable defines an integer value that limits the lowest bit rate at which BCC will ever transmit.

### Tag Syntax

```
<Var LowerLimit="Integer"/>
```

### Possible Values

A integer value of 128 or greater that expresses the lowest possible bit rate in bits per second. The default is 2048.

### Example

```
<Var LowerLimit="2048"/>
```

## Var: TargetRatio

This variable defines an integer value that represents the desired ratio of the decrease rate to the increase rate.

### Tag Syntax

```
<Var TargetRatio="Integer"/>
```

### Possible Values

Any positive integer starting at 1. The default value is 4.

**Warning!** Values less than 2 can result in unstable or unpredictable behavior of the BCC mechanism.

### Example

```
<Var TargetRatio="4"/>
```

## Var: TimeoutTransmission

The variables for List: Timeout (see page 83) determine when Helix Proxy declares that a media player has timed out. When BCC is used, the TimeoutTransmission variable determines the action that Helix Proxy takes when a timeout event occurs.

### Tag Syntax

```
<Var TimeoutTransmission="Boolean"/>
```

### Possible Values

- 0 – When a timeout occurs, apply the BCC decrease function. This is the default.
- 1 – Stop streaming on a timeout. If feedback reports resume, reinstate the BCC mechanism and begin streaming again at the media rate.

### Example

```
<Var TimeoutTransmission="0"/>
```

## Var: Trace

When set to 1, this variable causes Helix Proxy to record diagnostic statistics about each media player session that uses BCC.

### Tag Syntax

```
<Var Trace="Boolean"/>
```

### Possible Values

- 0 – Do not log diagnostic statistics.
- 1 – Log diagnostic statistics.

### Example

```
<Var Trace="1"/>
```

### Trace File Syntax

When Trace is set to 1, Helix Proxy writes to its main installation directory a diagnostic log for each stream in each BCC session. The diagnostic log is uniquely named using the following convention:

```
mdp_bcc_session_ID_stream_ID.txt
```

Here, *session\_ID* is the RTSP session ID string, described in the section “SessionID” on page 302. The value *stream\_ID* is the unique, numeric stream identifier, with 0 representing video and 1 denoting audio.

The diagnostic log contains a header that indicates the current BCC configuration parameters, as shown in the following example:

```
RTT Filter Algorithm: 0  
Stop Sending on Timeout: 0  
Increase Coefficient: 512000  
Decrease Coefficient : 8  
Increase Exponent: 0.5000  
Decrease Exponent: 0.5000  
Increase Threshold: 0  
Decrease Threshold: 5  
Lower Limit: 2048  
Target Ratio: 4
```

Following the header, the report records a new entry each time a receiver report arrives:

```
timestamp value send_rate  
timestamp value media_rate  
timestamp value rtt_filter  
timestamp value rtt_raw  
timestamp value num_recvd  
timestamp value num_lost  
timestamp value inc_coef
```

Each line in a block of statistics begins with the same timestamp, which records the number of milliseconds that elapsed from the start of the

streaming session to the generation of the receiver report. Each field within a statistics block records a different value:

send_rate	The transmission rate in bits per second that the Helix Proxy congestion control mechanism suggests.
media_rate	The rate of the media being transmitted in the present stream in bits per second.
rtt_filter	The smoothed RTT in milliseconds.
rtt_raw	The unfiltered RTT sample from the current client report in milliseconds.
num_rcvd	The number of packets received as reported in the current client report. This value is not cumulative for the duration of the session.
num_lost	The number of packets lost as reported in the current client report. This value is not cumulative for the duration of the session.
inc_coef	The dynamically adjusted increase coefficient.

## List: TCP

The variables in the TCP list affect only media players receiving data over TCP.

```
<List Name="TCP">
  <Var LimitRate="1"/>
</List>
```

### Registry Value

```
config.MediaDelivery.UserAgentProfiles.MediaPlayerName.CongestionControl.TCP
```

## Var: LimitRate

This variable determines whether Helix Proxy should limit the rate of media players receiving data over TCP to the channel rate, which is described in “Var: ChannelRate” on page 81.

### Tag Syntax

```
<Var LimitRate="Boolean" />
```

### Possible Values

- 0 – Do not limit rate to the channel rate when using TCP.
- 1 – Limit rate to the channel rate when using TCP. This is the default.

### Example

```
<Var LimitRate="1"/>
```

### List: Session

The <Session> list encompasses the rate manager properties.

```
<List Name="Session">  
  ...all rate manager properties...  
</List>
```

### Registry Value

```
config.MediaDelivery.UserAgentProfiles.MediaPlayerName.Session
```

### List: RateManager

The Helix Proxy rate manager models the media player buffer characteristics, providing information that allows the congestion control mechanism to vary the streaming rate.

```
<List Name="RateManager">  
  <Var BufferModel="ANNEXG"/>  
  <Var Type="ANNEXG"/>  
  <Var MultiStreamVerifier="AGGREGATE"/>  
  <Var ClientBufferLowWatermark="20"/>  
  <Var ClientBufferHighWatermark="120"/>  
  <Var TargetTimeLowWatermark="125"/>  
  <Var TargetTimeHighWatermark="250"/>  
  <Var ResendTimeLimit="1000"/>  
  <Var ClientBufferPeriod="7000"/>  
  <Var Trace="1"/>  
</List>
```

### Registry Value

```
config.MediaDelivery.UserAgentProfiles.MediaPlayerName.Session.RateManager
```

### Var: BufferModel

Defines the buffer modeling to use. Do not change this variable.

### Tag Syntax

```
<Var BufferModel="String"/>
```

**Possible Values**

Only ANNEXG is currently supported.

**Example**

```
<Var BufferModel="ANNEXG"/>
```

**Var: Type**

Defines the buffer type. Do not change this variable.

**Tag Syntax**

```
<Var Type="String"/>
```

**Possible Values**

Only ANNEXG is currently supported.

**Example**

```
<Var Type="ANNEXG"/>
```

**Var: MultiStreamVerifier**

Determines which stream the rate manager uses as the buffer model timeline.

**Tag Syntax**

```
<Var MultiStreamVerifier="String"/>
```

**Possible Values**

Only AGGREGATE is currently supported. This value assumes that the media player buffer is common to both audio and video.

**Example**

```
<Var MultiStreamVerifier="AGGREGATE"/>
```

**Var: ClientBufferLowWatermark**

Used by earlier, Helix-based media players, the ClientBufferLowWatermark variable sets the point at which Helix Proxy downshifts to a slower media rate. Reaching the low watermark indicates that the media player is consuming packets from its buffer faster than the network can deliver them.

Downshifting therefore causes the player to consume packets at a slower pace more appropriate for the network delivery speed.

#### Tag Syntax

```
<Var ClientBufferLowWatermark="Integer"/>
```

#### Possible Values

An integer that represents the percentage of the clip preroll to use as the low watermark. The value is typically less than 100. If a clip's preroll is four seconds, for example, and `ClientBufferLowWatermark` has a value of 50, Helix Proxy may downshift its streaming rate when its rate manager determines that the media player has two seconds of data in its buffer.

#### Example

```
<Var ClientBufferLowWatermark="50"/>
```

### Var: `ClientBufferHighWatermark`

Used by earlier, Helix-based media players, the `ClientBufferHighWatermark` variable sets the point at which Helix Proxy upshifts to a higher media rate. Reaching the high watermark indicates that the media player is consuming packets from its buffer slower than they are being streamed and delivered. Upshifting therefore causes the player to consume packets at a faster pace more appropriate for the packet delivery speed.

#### Tag Syntax

```
<Var ClientBufferHighWatermark="Integer"/>
```

#### Possible Values

An integer that represents the percentage of the clip preroll to use as the high watermark. The value is typically greater than 100. If a clip's preroll is five seconds, for example, and `ClientBufferHighWatermark` has a value of 120, Helix Proxy may upshift its streaming rate when its rate manager determines that the media player has five seconds of data in its buffer.

#### Example

```
<Var ClientBufferHighWatermark="120"/>
```

## Var: TargetTimeLowWatermark

Applicable to newer, Helix-based media players, the `TargetTimeLowWatermark` variable sets the point at which Helix Proxy may downshift to a slower media rate. Reaching the low watermark indicates that the media player is consuming packets from its buffer faster than the network can deliver them. Downshifting therefore causes the player to consume packets at a slower pace more appropriate for the network delivery speed.

**Tip:** You can set both the `TargetTimeLowWatermark` and `ClientBufferLowWatermark` variables for the same client. Helix Proxy uses the variable appropriate to the client and the rate control mechanism in use.

### Tag Syntax

```
<Var TargetTimeLowWatermark="Integer"/>
```

### Possible Values

An integer that represents the percentage of the media player's target time value to use as the low watermark. The client communicates the target time to Helix Proxy during the session startup. The time target value is the number of milliseconds of content that the client must have buffered to avoid entering a rebuffering state.

By default, the value for `TargetTimeLowWatermark` is 100, which equals the specified target time. Generally, the value should be 100 or higher. Suppose that the client's target time is 2000 milliseconds and `TargetTimeLowWatermark` has a value of 125. In this case, Helix Proxy may downshift its streaming rate when it determines that the client has just 2500 milliseconds of media buffered.

### Example

```
<Var TargetTimeLowWatermark="125"/>
```

## Var: TargetTimeHighWatermark

Used with newer, Helix-based media players, the `TargetTimeHighWatermark` variable sets the point at which Helix Proxy may upshift to a higher media rate. Reaching the high watermark indicates that the media player is consuming packets from its buffer slower than they are being streamed and

delivered. Upshifting therefore causes the player to consume packets at a faster pace more appropriate for the packet delivery speed.

**Tip:** You can set both the `TargetTimeHighWatermark` and `ClientBufferHighWatermark` variables for the same client. Helix Proxy uses the variable appropriate to the client and the rate control mechanism in use.

### Tag Syntax

```
<Var TargetTimeHighWatermark="Integer"/>
```

### Possible Values

An integer that represents the percentage of the media player's target time value to use as the high watermark. The value is typically greater than 100. The default is 200, which means that Helix Proxy may upshift its streaming rate when it determines that the client has buffered twice the amount of data than is necessary to avoid rebuffering. The value must be greater than that used for `TargetTimeLowWatermark`.

### Example

```
<Var TargetTimeHighWatermark="250"/>
```

## Var: ResendTimeLimit

This variable sets a limit to the packet resend requests that Helix Proxy honors. It prevents packet resends if Helix Proxy determines the requested data will arrive at the media player buffer too late to be of use.

### Tag Syntax

```
<Var ResendTimeLimit="Integer"/>
```

### Possible Values

- -1 – No time limit placed on packet resend requests. This is the default.
- An integer representing the number of milliseconds that Helix Proxy considers the data to be of use. For example, a value of 2000 means that Helix Proxy does not honor the resend request if it determines that the data will take more than two seconds to reach the media player after the resend was requested.

**Example**

```
<Var ResendTimeLimit="1000"/>
```

**Var: ClientBufferPeriod**

The ClientBufferPeriod variable sets the clip buffering time in milliseconds only if the media player does not obey the preroll time specified by the clip header, SDP, or target time.

**Tag Syntax**

```
<Var ClientBufferPeriod="Integer"/>
```

**Possible Values**

Positive integer indicating buffering time in milliseconds.

**Example**

```
<Var ClientBufferPeriod="5000"/>
```

**Var: Trace**

When set to 1, this variable causes Helix Proxy to record diagnostic statistics about the rate manager actions for each media player.

**Tag Syntax**

```
<Var Trace="Boolean"/>
```

**Possible Values**

- 0 – Do not log diagnostic statistics.
- 1 – Log diagnostic statistics.

**Example**

```
<Var Trace="1"/>
```

**Trace File Syntax**

When Trace is set to 1, Helix Proxy writes to its main installation directory a diagnostic log for each rate manager session. The logs allow you to evaluate the Helix Proxy buffer modeling for each media player. Each diagnostic log is uniquely named using the following naming convention:

ratemgr\_session\_ID.txt

Here, *session\_ID* is the RTSP session ID string, described in the section “SessionID” on page 302.

The diagnostic log contains a header that indicates the current media player buffer information, as shown in the following example:

```
PreDecodeBufferSize: 30000
PreDecodeBufferPeriod: 7000
BufferLowWaterMark: 3500
BufferHiWaterMark: 10500
```

The following table explains the buffer values.

<b>Rate Manager Log File Buffer Values</b>	
Buffer Value	Meaning
PreDecodeBufferSize	The size in bytes of the client buffer model as specified by the VideoPreDecoderBufferSize value in the media player’s RDF file. For more on RDF files, refer to “Capabilities Exchange” on page 67.
PreDecodeBufferPeriod	The number of milliseconds for preroll as encoded in the clip or set by Var: ClientBufferPeriod (see page 103). The rate manager will not evict packets from the buffer model until the buffer fills to this value.
BufferLowWaterMark	The low watermark of buffered data in milliseconds. For newer, Helix-based media players, this is the value of Var: TargetTimeLowWatermark (see page 101). Otherwise, it is calculated as the PreDecodeBufferPeriod value multiplied by the percentage specified by Var: ClientBufferLowWatermark (see page 99).
BufferHiWaterMark	The high watermark of buffered data in milliseconds. For newer, Helix-based media players, this is the value of Var: TargetTimeHighWatermark (see page 101). Otherwise, it is calculated as the PreDecodeBufferPeriod value multiplied by the percentage specified by Var: ClientBufferHighWatermark (see page 100).

#### Rate Manager Log File Fields

Following the configuration information, the log file records a line for each packet processed by the Helix Proxy rate manager. Each line includes the following fields:

```
time state playback_time streamID action [name=value... -> name=value...]
```

The following table explains what each field entry indicates.

<b>Rater Manager Log File Fields</b>	
Field	Information Logged
<i>time</i>	The time in milliseconds that has elapsed since streaming was initiated.
<i>state</i>	The current state of the media player, which may be predecode (packet queue mode), decode (playback mode), or init (buffering or rebuffering mode).
<i>playback_time</i>	The number of milliseconds into the stream timeline that the action occurred.
<i>streamID</i>	A numeric stream ID for the packet. RealMedia streams use a value of 0 to indicate a video stream, and a value of 1 to denote an audio stream.
<i>action</i>	The action performed with the packet, as described in “Rate Manager Actions” on page 105.
<i>name=value</i>	An optional set of name and value pairs. The values recorded can vary depending on the type of media player and the action.
->	A marker that indicates that the subsequent name and value pairs are state values for the buffer model that the rate manager has updated because of the logged action.

#### Rate Manager Actions

The following table describes each of the possible *action* entries.

<b>Rate Manager Log File Actions</b>	
Action	Meaning
DownShift	A downshift request was made. The log file does not indicate if the downshift actually occurred.
Evicted_packet	The packet has left the buffer to be decoded. The <i>buffsize</i> and <i>buffdepth</i> values decrease in size accordingly.
Expired_packet	The buffer model indicates the packet has left the buffer to be decoded. However, rate manager still maintains information for the packet in case a feedback report indicates the packet is still in the buffer. The <i>buffsize</i> and <i>buffdepth</i> values decrease in size accordingly.
Late_packet	The packet is late and is not included in the buffer model.

(Table Page 1 of 2)

**Rate Manager Log File Actions (continued)**

Action	Meaning
Queued_packet	A stream packet has entered the buffer model. The size value indicates the size of the packet. The buffsize and buffdepth values increase in size accordingly.
State: decode->init:	Rate manager has entered a rebuffering state.
State: init->predecode:	Rate manager entered predecode state, queueing packets but not evicting them.
State: predecode->decode:	Rate manager has entered decode state, evicting packets from the buffer.
Stream_blocked	Transmission is blocked because the buffer is full. Helix Proxy responds by limiting its delivery rate to the actual media rate.
UpShift	An upshift request was made. The log file does not indicate if the upshift actually occurred.

(Table Page 2 of 2)

**General Name and Value Pairs**

The following table indicates the name and value pairs that may appear in a rate manager report for general packet operations.

**Rate Manager Log File Name and Value Pairs for General Operations**

Name and Value	Meaning
buffdepth= <i>ms</i>	Buffer size in milliseconds of media playback.
buffsize= <i>bytes</i>	Buffer size in bytes.
retry= <i>time</i>	Time in milliseconds when rate manager will check its buffer model to determine if a packet can be resent. Included when the action is Stream_blocked.
seq= <i>integer</i>	Packet sequence number.
size= <i>bytes</i>	Packet size in bytes.
ts= <i>time</i>	Packet timestamp, which indicates the number of milliseconds into the timeline that the packet data should be played.

**Rate Manager Logging Examples**

The following example indicates that a packet has been added to the buffer model. The rate manager increases the buffer size and depth in accordance with the packet size:

```
5840 decode 3233 0 Queued_packet seq=1139 ts=15883 size=268 ->
buffsize=269241 buffdepth=12650
```

The next example indicates that rate manager has expired a packet that the client should have played. The rate manager decreases the buffer size by the size of the packet:

```
5873 decode 3271 0 Expired_packet seq=139 ts=3270 size=61 -> bufsize=271331
```

### List: InputSource

This list affects how Helix Proxy interacts with multi-rate content files.

```
<List Name="InputSource">
  <Var InitialMediaRate="32000"/>
  <Var MinPreroll="3000"/>
  <Var NetworkAffinity="5000"/>
</List>
```

#### Registry Value

```
config.MediaDelivery.UserAgentProfiles.MediaPlayerName.InputSource
```

### Var: InitialMediaRate

This variable selects the initial media rate for a session when streaming from a multi-rate clip. This value must be smaller than the ChannelRate value.

**For More Information:** See “Var: ChannelRate” on page 81.

#### Tag Syntax

```
<Var InitialMediaRate=Integer/>
```

#### Possible Values

Positive integer representing bits per second (bps).

#### Example

```
<Var InitialMediaRate="16000"/>
```

### Var: MinPreroll

This variable affects only certain media players. It overrides the preroll value for a clip.

#### Tag Syntax

```
<Var MinPreroll=Integer/>
```

### Possible Values

Positive integer representing milliseconds. The default is 1000.

### Example

```
<Var MinPreroll="3000"/>
```

## Var: NetworkAffinity

The NetworkAffinity variable affects only RealNetworks 3GPP clients that are not compliant with Annex G. It sets the top streaming bit rate for the network channel. It therefore functions like the ChannelRate value described in “Var: ChannelRate” on page 81. You can set the value of NetworkAffinity arbitrarily high, but Helix Proxy may periodically overload the channel rate causing packet loss.

### Tag Syntax

```
<Var NetworkAffinity="Integer"/>
```

### Possible Values

Positive integer indicating maximum channel throughput in bits per second.

### Example

```
<Var NetworkAffinity="5000"/>
```

## List: StaticPushSource

This list affects stream of multi-rate, on-demand clips.

```
<List Name="StaticPushSource">  
  <Var AllowInitialExternalSubscription="0"/>  
</List>
```

### Registry Value

```
config.MediaDelivery.UserAgentProfiles.MediaPlayerName.InputSource.  
StaticPushSource
```

## Var: AllowInitialExternalSubscription

This variable determines whether Helix Proxy honors the initial rate subscription request from the media player. It affects RealNetworks media players only.

### Tag Syntax

```
<Var AllowInitialExternalSubscription="Boolean"/>
```

### Possible Values

- 0 – Do not honor the initial rate subscription request.
- 1 – Honor the initial rate subscription request. This is the default.

### Example

```
<Var AllowInitialExternalSubscription="0"/>
```



## MULTICASTING

This chapter explains back-channel multicasting, a method of multicasting in which 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 also enables the Proxy Monitor to track how many players are viewing the multicast.

**For More Information:** Refer to the multicasting chapter in *Helix Proxy Administration Guide*.

### Multicasting Syntax

The following is a sample of the multicasting syntax. This markup appears within the FSMount list.

```
<List Name="Multicast">
  <Var Enabled="True"/>
  <Var AnnounceSAP="True"/>
  <Var RTSPPort="554"/>
  <Var TTL="16"/>
  <Var Resend="True"/>
  <Var AddressRange="address-address"/>
  <Var DeliveryOnly="False"/>
  <List Name="Controllist">
    <List Name="100">
      <Var Allow="172.16.2.24:255.0.0.0"/>
    </List>
    <List Name="200">
      <Var Allow="201.34.23.0:255.255.255.254"/>
    </List>
  </List>
</List>
```

## List: Multicast

This list controls the back-channel multicasting features.

```
<List Name="Multicast">  
  ...list of back-channel multicast variables...  
</List>
```

### Registry Value

config.Multicast

## Var: Enabled

Use this variable to turn multicasting on or off.

### Tag Syntax

```
<Var Enabled="Boolean"/>
```

### Possible Values

- 0|False – Back-channel multicast is not available.
- 1|True – Back-channel multicast is available.

### Example

```
<Var Enabled="True"/>
```

## Var: AnnounceSAP

Indicates whether to send SAP files.

### Tag Syntax

```
<Var AnnounceSAP="Boolean"/>
```

### Possible Values

- 0|False – Do not send SAP files.
- 1|True – Send SAP files.

### Example

```
<Var AnnounceSAP="True"/>
```

## Var: RTSPPort

This is the client port number to which Helix Proxy will direct its RTSP streams.

### Tag Syntax

```
<Var RTSPPort="integer"/>
```

### Possible Values

The client's port number. The default value is 554.

### Example

```
<Var RTSPPort="554"/>
```

## Var: TTL

This variable defines the time-to-live for multicast packets travelling over the network. Each time a multicast data packet passes through a multicast-enabled router, its time-to-live decreases by 1. When the value is decremented to 0, the router discards the data packet.

### Tag Syntax

```
<Var TTL="integer"/>
```

### Possible Values

- 0 to 225 – The default value is 16.

### Example

```
<Var TTL="16"/>
```

## Var: Resend

This variable allows or denies requests from clients for resends of missing UDP packets.

### Tag Syntax

```
<Var Resend="Boolean"/>
```

### Possible Values

- 0|False – Deny requests from clients for resends.
- 1|True – Allow requests from clients for resends.

### Example

```
<Var Resend="True"/>
```

## Var: AddressRange

This defines the range of addresses to which you want to send streams, in the form of address-address. Helix Proxy uses the first available address in this range. If you are using other types of multicast, be sure that the address ranges are different and do not overlap. If your multicast streams are referenced in SMIL files, you will need one address for each stream.

### Tag Syntax

```
<Var AddressRange="address-address"/>
```

### Possible Values

- ###.###.###.###-###.###.###.### – IPv4 address range, must be within the range of 224.0.1.0 to 239.255.255.255

### Example

```
<Var AddressRange="224.0.1.0-224.0.1.3"/>
```

## Var: DeliveryOnly

Requires clients listed in Controllist to receive only multicast transmissions from Helix Proxy. When DeliveryOnly is False, clients on Controllist can receive both multicasts and unicasts. Default value is False.

### Tag Syntax

```
<Var DeliveryOnly="Boolean"/>
```

### Possible Values

- 0|False – Receive both multicasts and unicasts.
- 1|True – Receive only multicast transmissions.

**Example**

```
<Var DeliveryOnly="False"/>
```

**List: Controllist**

The Controllist list gives the addresses of clients required to receive multicast transmissions.

```
<List Name="Controllist">
  ...required multicast receiver lists...
</List>
```

**Registry Value**

```
config.Multicast.Controllist
```

**List: Rule Number**

Access Control rule number for this list. For more information, refer to “Access Control” on page 131.

```
<List Name="Rule Number">
  ...access rule...
</List>
```

**Registry Value**

```
config.Multicast.Controllist.integer
```

**Example**

```
<List Name="100"> ... </List>
```

**Var: Allow**

Address and netmask, separated by a colon, of clients allowed to receive multicast transmissions. There must always be at least one entry in Controllist.

**Tag Syntax**

```
<Var Allow="string"/>
```

**Possible Values**

- Any
- Specific IP address

- Range of IP addresses

To specify a range of IP addresses, either place a colon after the IP address and give the full subnet mask, or place a slash mark after the IP address and give the number of bytes for the subnet mask. For example, the following are equivalent values. Both examples specify the range of addresses from 172.16.3.0 to 172.16.3.254:

- 172.16.3.0:255.255.255.0
- 172.16.3.0/24

**Example**

```
<Var Allow="172.16.3.0/24"/>
```

## PULL SPLITTING

The pull splitting feature of Helix Proxy conserves bandwidth for broadcasting live material. This chapter explains the pull splitting variables.

### Pull Splitting Operation

When pull splitting is enabled, live streams are transmitted by Helix Server to the Helix Proxy receiver only after the first client makes a request. Every successive client that requests a live stream receives the stream directly from Helix Proxy. Helix Proxy does not have to obtain another stream from the origin Helix Server for each client—the live request stream is “shared” among all clients who request it.

**For More Information:** Refer to the proxy setup chapter in *Helix Proxy Administration Guide*.

### Server Compatibility

Helix Proxy can receive broadcasts from Helix Server, as well as earlier versions of RealSystem Server. However, depending on the server version with which Helix Proxy is communicating, different variables must be defined in the proxy configuration file:

- When splitting from Helix Server 9 and later, you need to define the FSMount list variables, some Proxy list variables, and the BroadcastReceiver list variables. Refer to these sections:
  - “File System Setup” on page 118.
  - “Broadcast Receiver Configuration” on page 121.
  - “Proxy List Splitting Variables” on page 127.

- When splitting from RealSystem Server versions G2 through 8, you need to define the appropriate FSMount list variables, as well as add some variables to the Proxy list. Refer to these sections:
  - “File System Setup” on page 118.
  - “Proxy List Splitting Variables” on page 127.

## File System Setup

The following sections describe the list and variable tags within the FSMount list that must be defined to enable pull splitting between Helix Proxy and Helix Server (version 9 and later), or to support backward compatibility between Helix Proxy and RealSystem Server G2 through 8.

### FSMount Syntax

To enable pull splitting between Helix Proxy and Helix Server, a proxy receiver list must be specified within the FSMount list.

#### Pull Splitting with Helix Server Version 9 and Later

To enable pull splitting between Helix Proxy and Helix Server, versions 9 and later, a broadcast distribution list must be specified within the FSMount list. The ShortName and MountPoint variables must be defined as follows:

```
<List Name="FSMount">
  <List Name="Broadcast Distribution List">
    <Var ShortName="pn-broadcast-receiver-fs"/>
    <Var MountPoint="/broadcast/" />
  </List>
  ...
</List>
```

#### Pull Splitting with RealSystem Server Versions G2 Through 8

To support backward-compatible pull splitting between Helix Proxy and RealSystem Server versions G2 through 8, a Splitter\_DoubleURL list must be included within the FSMount list. The ShortName, MountPoint, and Port variables must be defined as follows:

```
<List Name="FSMount">
  <List Name="Splitter_DoubleURL">
    <Var ShortName="pn-splitter"/>
    <Var MountPoint="/split/" />
  </List>
</List>
```

```

    <Var Port="3030"/>
    <Var SplitterProtocol="udp"/>
  </List>
  ...
</List>

```

**Note:** The SplitterProtocol variable is optional. If any of the required variables are not included, or if any of their values are not set as indicated above, Helix Proxy will not be able to operate in pull splitting mode.

### List: Broadcast Distribution List Name

To enable pull splitting, this list must be defined to specify the file system configuration for the broadcast distribution list.

**Note:** This list tag is used only for pull splitting from Helix Server 9 and later. For pull splitting from RealSystem Server versions G2 through 8, see “List: Splitter\_DoubleURL” on page 119.

#### Registry Value

```
config.FSMount.string
```

### List: Splitter\_DoubleURL

To enable pull splitting for Helix Proxy and RealSystem Server versions G2 through 8, this LIST must be defined to specify the file system configuration.

**Warning!** This list tag can *only* be used for splitting from RealSystem Server versions G2 through 8. For splitting from Helix Server 9 and later, see “List: Broadcast Distribution List Name” on page 119.

#### Registry Value

```
config.FSMount.Splitter_DoubleURL
```

### Var: ShortName

The ShortName variable specifies the short name for the pull splitting plug-in and must be defined, regardless of the Helix Server version with which Helix

Proxy is communicating. However, the value of the variable does depend on the Helix Server version.

### Tag Syntax

```
<Var ShortName="string"/>
```

### Possible Values

- pn-broadcast-receiver-fs – Default value of the short name for the pull splitting plug-in to use when communicating with Helix Server 9 and later.
- pn-splitter – Default value of the short name for the pull splitting plug-in to use when communicating with RealSystem Server versions G2 through 8.

### Examples

```
<Var ShortName="pn-broadcast-receiver-fs"/>
```

```
<Var ShortName="pn-splitter"/>
```

## Var: MountPoint

The MountPoint variable specifies the file system path to include in URLs that reference pull splitting streams. The variable must be defined, regardless of the Helix Server version with which Helix Proxy is communicating. However, the value of the variable does depend on the Helix Server version.

### Tag Syntax

```
<Var MountPoint="string"/>
```

### Possible Values

- /broadcast/ – Default splitting mount point for Helix Server 9 and later.
- /split/ – Default splitting mount point for RealSystem Server versions G2 through 8.
- /string/ – Any valid file system path to use as the splitting mount point default value for any supported version of Helix Server.

### Example

```
<Var MountPoint="/broadcast"/>
```

```
<Var MountPoint="/split/" />
```

### Var: Port

Used only with RealSystem Server versions G2 through 8, this variable specifies the port number to which Helix Proxy will listen for pull splitting requests.

#### Tag Syntax

```
<Var Port="integer" />
```

#### Possible Values

- 3030 – Default data channel for pull splitting requests.

#### Example

```
<Var Port="3030" />
```

### Var: SplitterProtocol

Used only with RealSystem Server versions G2 through 8, this variable specifies the protocol to use in transmitting streams. It must be set to the same value on both Helix Server and Helix Proxy.

#### Tag Syntax

```
<Var SplitterProtocol="string" />
```

#### Possible Values

- udp/unicast – This is the default.
- tcp – If your network allows only TCP, or you are splitting through a firewall, use the tcp value.

#### Example

```
<Var SplitterProtocol="udp" />
```

## Broadcast Receiver Configuration

To enable splitting between Helix Proxy and Helix Server, versions 9 and later, include the BroadcastReceiver list in the proxy configuration file. This list is

not necessary when receiving a streaming from RealSystem Server versions G2 through 8.

**Tip:** For a multi-rate or SureStream broadcast, transport the minimal set of live packets needed to support the union of current client stream subscriptions using this list. Insure that server-to-proxy bandwidth never oversubscribes to stream rates that are greater than those needed to fulfill the current client connections at the Helix Proxy. Dynamically subscribe and/or unsubscribe to SureStream subscription rates as needed, based on client connections at the Helix Proxy.

## Broadcast Receiver Syntax

The following sample demonstrates the broadcast receiver syntax:

```
<List Name="BroadcastReceiver">
  <List Name="Receivers"/>
    <List Name="ProxyReceiver">
      <Var Protocol="auto"/>
      <Var SureStreamAware="True"/>
      <Var PullSplitEnabled="True"/>
      <Var UseTCPForPullBackchannel="0"/>
      <Var PathPrefix="*" />
      <Var ResendSupported="1"/>
      <Var FECLevel="0"/>
      <Var UseRTSPInitiate="True"/>
    </List>
  </List>
</List>
```

## List: BroadcastReceiver

For Helix Proxy and Helix Server, versions 9 and later, this is the primary list tag used to identify the presence of Helix Proxy global broadcast receivers in the configuration file:

```
<List Name="BroadcastReceiver">
  ...additional proxy broadcast receiver information...
</List>
```

### Registry Value

config.BroadcastReceiver

## List: Receivers

The Receivers list contains a list of proxy broadcast receivers, where each Helix Proxy receiver is itself a list.

```
<List Name="Receivers">
  ...proxy receiver lists...
</List>
```

### Registry Value

config.BroadcastReceiver.Receivers

## List: Proxy Receiver Name

This list defines a proxy global broadcast receiver. Specify any name for the list, as long as it is unique and no longer than 64 bytes.

```
<List Name="Proxy Receiver Name">
  ...additional information about this proxy broadcast receiver...
</List>
```

### Registry Value

config.BroadcastReceiver.Receivers.*string*

### Example

```
<List Name="Proxy Receiver"> ... </List>
```

## Var: Protocol

This variable specifies the protocol to use in transmitting streams. It must be set to the same value on both Helix Server and the Helix Proxy. When this variable is not defined in the configuration file, Helix Proxy uses the UDP/unicast protocol.

### Tag Syntax

```
<Var Protocol="string"/>
```

### Possible Values

- auto – Enables automatic transport negotiation between Helix Proxy and Helix Server. Helix Proxy initiates data packet transmission using the optimal transport available by the network segments between Helix Proxy and Helix Server. The order of priority is UDP/unicast, then TCP.

- udp/unicast
- tcp – If your network only allows TCP or you are splitting through a firewall, use a value of tcp.

#### Example

```
<Var Protocol="auto"/>
```

### Var: SureStreamAware

The SureStreamAware variable is set on both Helix Server and Helix Proxy receiver to enable SureStream-aware splitting. This variable should be defined for each distinct broadcast distribution. When this variable is not defined in the configuration file, the default value is true.

#### Tag Syntax

```
<Var SureStreamAware="Boolean"/>
```

#### Possible Values

- True – Helix Proxy uses SureStream-aware splitting.
- False – All SureStream data streams are sent to Helix Proxy.

#### Example

```
<Var SureStreamAware="True"/>
```

### Var: PullSplitEnabled

The PullSplitEnabled variable must be set to True to enable pull splitting between Helix Proxy and Helix Server, versions 9 and later. When pull splitting is enabled, Helix Server transmits streams to Helix Proxy only after the first client makes a request.

**Warning!** When this variable is set to False, or not defined in the configuration file, Helix Proxy cannot split live broadcasts from Helix Server and later.

**For More Information:** See “Var: UseRTSPInitiate” on page 127.

#### Tag Syntax

```
<Var PullSplitEnabled="Boolean"/>
```

**Possible Values**

- True – Enable pull splitting.
- False – Disable pull splitting between Helix Proxy Helix Server.

**Example**

```
<Var PullSplitEnabled="True"/>
```

**Var: UseTCPForPullBackchannel**

When pull splitting is in use and the Protocol variable is set to udp/unicast or udp/multicast, UseTCPForPullBackchannel can be used to create a back channel for initiation and resend requests. Helix Server then does not transmit any streams to the Helix Proxy until the first client makes a request. When this variable is not defined in the configuration file, it is created by the plug-in and set to 0.

**Tag Syntax**

```
<Var UseTCPForPullBackchannel="Boolean"/>
```

**Possible Values**

- 1 – Use TCP for the pull back channel.
- 0 – Do not use TCP for the pull back channel.

**Example**

```
<Var UseTCPForPullBackchannel="0"/>
```

**Var: PathPrefix**

The path prefix specified in this variable indicates that this path contains a pull splitting request. The path prefix does not need to be unique for each broadcast. When this variable is not defined in the configuration file, it is created by the plug-in and by default, set to '\*’.

**Tag Syntax**

```
<Var PathPrefix="string"/>
```

**Possible Values**

- */string/* – any valid file system path to use a prefix

### Example

```
<Var PathPrefix="/broadcasts/" />
```

## Var: ResendSupported

ResendSupported can be used to make resend requests to Helix Proxy. When this variable is not defined in the configuration file, it is created by the plug-in and set to 1.

**Note:** Making resend requests to the Helix Proxy produces higher quality, but requires more overhead.

### Tag Syntax

```
<Var ResendSupported=Boolean />
```

### Possible Values

- 1 – Make resend requests to the Helix Proxy.
- 0 – Do not make resend requests to the Helix Proxy.

### Example

```
<Var ResendSupported="1" />
```

## Var: FECLevel

The FECLevel or Forward Error Correction rate is the percent of corrective packets sent (0 – 100). A higher number results in better quality, but consumes more network bandwidth. The default value for use over the Internet is 20. When using splitting within a network, set the value to 0, since the need for error correction is unlikely.

### Tag Syntax

```
<Var FECLevel=integer />
```

### Possible Values

- *integer* – Percent of corrective packets sent (0 – 100).

### Example

```
<Var FECLevel="32" />
```

## Var: UseRTSPInitiate

The UseRTSPInitiate variable must be set to True to support pull splitting between Helix Proxy and Helix Server. When pull splitting is enabled, the UseRTSPInitiate variable activates a broadcast distribution initiation method that connects the Helix Server to the RTSP port (554).

**Warning!** When this variable is set to False, or not defined in the configuration file, Helix Proxy cannot split live broadcasts from Helix Server.

**For More Information:** Refer to “Var: PullSplitEnabled” on page 124.

### Tag Syntax

```
<Var UseRTSPInitiate="Boolean"/>
```

### Possible Values

- True – Enable RTSP port broadcast distribution initiation.
- False – Disable RTSP port broadcast distribution initiation, and therefore, disable pull splitting between Helix Proxy and Helix Server.

### Example

```
<Var UseRTSPInitiate="True"/>
```

## Proxy List Splitting Variables

To enable pull splitting between Helix Proxy and Helix Server, certain variables must be defined in the Proxy list. Depending on the version of Helix Server with which Helix Proxy is communicating, different variables are required:

- To enable pull splitting between Helix Proxy and Helix Server, versions 9 and later, the Proxy list must define the BroadcastDistributionMountPoint variable, with a value the same as the MountPoint specified in the FSMount list:

```
<List Name="Proxy">
  ...
  <Var BroadcastDistributionMountPoint="/broadcast/" />
</List>
```

- To support backward-compatible pull splitting between Helix Proxy and RealSystem Server G2 through 8, the BitsaveEnable and BitsaveMountPoint variables must be defined, with the BitsaveMountPoint value the same as the MountPoint value specified in the FSMount list:

```
<List Name="Proxy">
  ...
  <Var BitsaveEnable="1"/>
  <Var BitsaveMountPoint="/split"/>
</List>
```

**Note:** Prior to Helix Proxy 9, the RTSPPort variable in the Proxy list was used to define the RTSP listen port. In the latest version of Helix Proxy, the Port variable in the root FSMount list is now used to specify the RTSP listen port. For more information, refer to “Var: Port” on page 121.

## Var: BroadcastDistributionMountPoint

For Helix Proxy and Helix Server, versions 9 and later, the BroadcastDistributionMountPoint variable must be specified for pull splitting to operate. The file system path specified in the variable is automatically added to URLs when splitting is enabled.

### Tag Syntax

```
<Var BroadcastDistributionMountPoint="string"/>
```

### Possible Values

- /broadcast/ – Default splitting mount point to add to URLs when pull splitting from Helix Server 9 and later.
- /string/ – Any valid file system path to use as the splitting mount point. However, the value specified for this variable should be the same as that defined in Var: MountPoint in the FSMount list.

### Example

```
<Var BroadcastDistributionMountPoint="/broadcast"/>
```

## Var: BitsaveEnable

For splitting from RealSystem Server versions G2 through 8, the BitsaveEnable variable enables Helix Proxy to perform pull splitting with Helix Server. When

the value is set to 1, Helix Proxy streams all live requests, rather than opening separate data channels between the Helix Server and the client.

#### Tag Syntax

```
<Var BitsaveEnable="Boolean"/>
```

#### Possible Values

- 1 – Enable pull splitting.
- 0 – Disable pull splitting.

#### Example

```
<Var BitsaveEnable="1"/>
```

### Var: BitsaveMountPoint

For splitting from RealSystem Server versions G2 through 8, the BitsaveMountPoint variable must be specified when performing pull splitting. The file system path specified in the variable is automatically added to URLs when splitting is enabled.

**Warning!** This variable can *only* be used for pull splitting from RealSystem Server versions G2 through 8. For pull splitting from Helix Server 9 and later, see “Var: BroadcastDistributionMountPoint” on page 128.

#### Tag Syntax

```
<Var BitsaveMountPoint="string"/>
```

#### Possible Values

- /split/ – Default splitting mount point to add to URLs when pull splitting from RealSystem Server versions G2 through 8.
- /string/ – Any valid file system path to use as the splitting mount point. However, the value specified for this variable should be the same as that defined in Var: MountPoint in the FSMount list.

#### Example

```
<Var BitsaveMountPoint="/split"/>
```



## ACCESS CONTROL AND AUTHENTICATION

This chapter describes access control and authentication features, which restrict access to Helix Proxy and its content.

### Access Control

The access control feature allows you to admit or deny access to Helix Proxy based on the IP address of a user making a request. Access control rules are enforced before authentication of requested content, and provide the broadest means for allowing or denying access to content or to Helix Administrator pages.

**For More Information:** See the access control chapter of *Helix Proxy Administration Guide*.

### Access Control Syntax

The following sample contains three access rules. The first rule allows access to any port from localhost, giving you local machine access to Helix Administrator. The second rule denies access to port 7070, which is reserved for Helix Proxy internal use, from any computer. (You should not change this rule.) The third 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:

```
<List Name="AccessControl">
  <List Name="0">
    <Var Description="Allow localhost access (do not edit)"/>
    <Var Access="Allow"/>
    <Var From="localhost"/>
    <Var To="any"/>
    <List Name="Ports">
      <Var Port_1="any"/>
    </List>
  </List>
</List>
```

```
<List Name="1">
  <Var Description="Deny connections to port 7070 (do not edit)"/>
  <Var Access="Deny"/>
  <Var From="any"/>
  <Var To="any"/>
  <List Name="Ports">
    <Var Port_1="7070"/>
  </List>
</List>
<List Name="2">
  <Var Description="Allow all other connections"/>
  <Var Access="Allow"/>
  <Var From="any"/>
  <Var To="any"/>
  <List Name="Ports">
    <Var Port_1="any"/>
  </List>
</List>
</List>
```

## List: AccessControl

The main AccessControl list contains all of the access control information:

```
<List Name="AccessControl">
  ...all access control information...
</List>
```

### Registry Value

config.AccessControl

## List: RuleNumber

A *RuleNumber* creates a rule within the AccessControl list:

```
<List Name="AccessControl">
  <List Name="RuleNumber">
    ...variables that create the rule...
  </List>
</List>
```

Use a unique integer value for *RuleNumber*. Helix Proxy processes rules in numeric order, searching the rules to find the first rule that matches the requesting address. Because Helix Proxy searches the list of rules in numerical order, make your broadest categories first.

**Tip:** A rule integer can be any length, but RealNetworks recommends more than one digit in case you need to add more rules later. With multiple digits, such as 100, 200, 300, and so on, new lists can be inserted between existing lists.

### Registry Value

```
config.AccessControl.RuleNumber
```

### Example

```
<List Name="100"> ... </List>
```

## Var: Access

Use an Access variable to create a rule that allows or denies access.

### Tag Syntax

```
<Var Access="value"/>
```

### Possible Values

- allow
- deny

### Example

```
<Var Access="deny"/>
```

## Var: From

Use From to set the address or address range of the client computer(s) whose access you want to limit.

### Tag Syntax

```
<Var From="string"/>
```

### Possible Values

- Any
- Specific IPv4 address.
- Specific IPv6 address.

**Note:** Client connections using an IPv6 address mapped to an IPv4 address are checked against IPv4-based rules.

- Range of IP addresses.

To specify a range of IP addresses, either place a colon after the IP address and give the full subnet mask, or place a slash mark after the IP address and give the number of bytes for the subnet mask. For example, the following are equivalent IPv4 values to use in the From variable:

- 172.16.3.0:255.255.255.0
- 172.16.3.0/24

Both examples specify the range of IPv4 addresses from 172.16.3.0 to 172.16.3.254. This is shown as Client IP Address in Helix Administrator.

#### Example

```
<Var To="172.16.3.0/24"/>
```

### Var: To

Use To to set the address of the host Helix Proxy or network card of the hosting machine.

#### Tag Syntax

```
<Var To="string"/>
```

#### Possible Values

- Any
- Specific IPv4 address.
- Specific IPv6 address.

#### Example

```
<Var To="Any"/>
```

### List: Ports

Use a list named Ports to associate a list of ports with a rule:

```
<List Name="Ports">  
  ...variables that define a port...  
</List>
```

**Registry Value**

```
config.AccessControl.RuleNumber.ports
```

**Var: Port\_n**

Use a `Port_n` variable, where `n` is an integer, to define ports that you want to regulate. For example, if Helix Proxy is set up to use port 554 to listen for RTSP, you would use a value of 554 to add the RTSP Port to the list of ports regulated by the given rule.

**Tag Syntax**

```
<Var Port_n="string"/>
```

**Possible Values**

Use port numbers assigned to the feature or features for which you want to regulate access.

**Example**

```
<Var Port_1="8888"/>
```

## Authentication Databases

Databases store user names and passwords of authorized users. You can use the following default databases:

- `Admin_Basic` – Flat file database used to define access to Helix Administrator.
- `Connect_RN5` – Flat file database used to define user connection validation.

Helix Proxy installation also includes structured database templates in ODBC-compliant formats in its database subdirectory.

**For More Information:** See the authentication chapter of *Helix Proxy Administration Guide*.

**Database Syntax**

The following sample illustrates the configuration syntax that defines authentication database use:

```
<List Name="Databases">
  <List Name="Admin_Basic">
    <Var PluginID="rn-db-flatfile"/>
    <Var Path="C:\Program Files\Real\Helix Proxy\adm_b_db"/>
  </List>
  <List Name="Connect_RN5">
    <Var PluginID="rn-db-flatfile"/>
    <Var Path="C:\Program Files\Real\Helix Proxy\con_r_db"/>
  </List>
</List>
```

## List: Databases

This is the master list of databases available for use with Helix Proxy. You must define databases in this list before you either identify realms or choose the specific realm and database that provides authentication for users.

```
<List Name="Databases">
  ...authentication databases...
</List>
```

### Registry Values

config.Databases

## List: Database Name

This list provides a descriptive and unique name for each database.

```
<List Name="Database Name">
  ...database information...
</List>
```

### Registry Values

config.Databases.*string*

### Example

```
<List Name="Connect_RN5"> ... </List>
```

## Var: PluginID

This variable provides the name of plug-in that will interact with the database. Use with all database types.

### Tag Syntax

```
<Var PluginID="value"/>
```

### Possible Values

- rn-db-flatfile – Text file. Requires use of only the Path variable.
- rn-db-odbc—ODBC-compliant databases. Requires use of the Name variable. The Password, TableNamePrefix, and User variables are optional.

### Example

```
<Var PluginID=rn-db-odbc"/>
```

## Var: Path

Required for text file databases, this variable sets the location where database files are stored.

### Tag Syntax

```
<Var Path="string"/>
```

### Possible Values

Use the full directory path where the database files are located.

### Example

```
<Var Path="C:\Program Files\Real\Helix Proxy\con_r_db"/>
```

## Var: Name

This is the name required by ODBC-compliant databases.

### Tag Syntax

```
<Var Name="string"/>
```

### Possible Values

Use the actual database name, rather than a description. Consult your database application documentation for more information.

### Example

```
<Var Name="authdemo.db"/>
```

## Var: User

User name optionally used by ODBC-compliant databases.

### Tag Syntax

```
<Var User="string"/>
```

### Possible Values

Any user name associated with the database.

### Example

```
<Var User="m"/>
```

## Var: Password

Password optionally used by ODBC-compliant databases.

### Tag Syntax

```
<Var Password="password"/>
```

### Possible Values

The password associated with the user mentioned above.

### Example

```
<Var Password="letmein"/>
```

## Var: TableNamePrefix

Optional prefix used to make field names unique, when used with an existing, ODBC-compliant database.

### Tag Syntax

```
<Var User="prefix"/>
```

### Possible Values

Use a briefly descriptive and unique prefix.

### Example

```
<Var TableNamePrefix="MRKT"/>
```

## Authentication Realms

Authentication realms associate databases with a protocol to encrypt their username, passwords, and other information. When you configure a realm, you associate a database with this realm and Helix Proxy references this database to verify a user's credentials. You can use the following realms supplied during the installation process:

- SecureAdmin—Realm used to authenticate Helix Administrator users.
- ConnectRealm—Realm used to authenticate proxy users.

**Warning!** Do not remove the SecureAdmin realm, as this will disable access to Helix Administrator.

**For More Information:** See the authentication chapter of *Helix Proxy Administration Guide*.

## Realm Syntax

The following sample illustrates the syntax that defines authentication realms:

```
<!-- AUTHENTICATION -->
<List Name="AuthenticationRealms">
  <List Name="SecureAdmin">
    <Var Realm="example.com.AdminRealm"/>
    <List Name="BasicAuthenticator">
      <Var PluginID="rn-auth-basic"/>
      <Var DatabaseID="Admin_Basic"/>
    </List>
  </List>
  <List Name="ConnectRealm">
    <Var Realm="example.com.ConnectRealm"/>
    <List Name="BasicAuthenticator">
      <Var PluginID="rn-auth-basic"/>
      <Var DatabaseID="Connect_RN5"/>
    </List>
  </List>
</List>
```

## List: AuthenticationRealms

This list identifies all available realms. You must define realms within this list before you choose the specific realm and database that provides authentication for users.

```
<List Name="AuthenticationRealms">  
  ...realms associated with databases...  
</List>
```

### Registry Values

config.AuthenticationRealms

## List: Realm Name

This list indicates realms and associated databases available for use with Helix Proxy. For the realm name value, use a brief, descriptive name.

```
<List Name="Realm Name">  
  ...realms associated with databases...  
</List>
```

### Registry Values

config.AuthenticationRealms.*string*

### Example

```
<List Name="SecureAdmin"> ... </List>
```

## Var: Realm

This variable provides the unique realm ID.

### Tag Syntax

```
<Var Realm="string.string" />
```

### Possible Values

Use a value that uniquely identifies the realm. The default realms conform to the following format:

*proxyname.RealmId*

In this syntax, *proxyname* is the hostname of the machine on which Helix Proxy is located, and *RealmId* is the name of the realm. You do not have to use this

convention, but you must include a period (.) in the realm ID or the realm will not work properly.

#### Example

```
<Var Realm="CUV4X-2K.ConnectRealm"/>
```

### List: Protocol Name

Authentication protocols determine the password encryption method used by Helix Proxy.

```
<List Name="Protocol Name">
  ...components to be used with a realm and protocol...
</List>
```

#### Registry Value

```
config.AuthenticationRealms.Realm Name.string
```

#### Possible Values

The proxy supports three protocols for encrypting user passwords:

- Basic—Encodes the user's name and password with the Base64 algorithm. Use BasicAuthenticator as the list name.
- RealSystem 5.0—Encodes user names and passwords in a more secure manner than the Basic protocol. Compatible with RealNetworks media players version 5 and up. Use RN5Authenticator as the list name.
- Windows NT LAN Manager—Uses the existing Windows NT or Windows 2000 database of users and groups. It also allows access control of content through NTFS file permissions. Use Authenticator as the list name.

#### Examples

```
<List Name="Authenticator">
```

### Var: PluginID

Name of the plug-in that will interact with a realm defined in Var: Realm.

#### Tag Syntax

```
<Var PluginID="string"/>
```

### Possible Values

- rn-auth-basic – For use with the Basic protocol.
- rn-auth-rn5 – For use with the RealSystem 5.0 protocol.
- rn-auth-sspi – For use with the Windows LAN Manager protocol.

### Example

```
<Var PluginID="rn-auth-basic"/>
```

## Var: DatabaseID

Database associated with the realm. Required for Basic and RealSystem 5.0 encryption protocols.

### Tag Syntax

```
<Var DatabaseID="string"/>
```

### Possible Values

Use a database name assigned in the *Database Name* List.

### Example

```
<Var DatabaseID="Connect_RN5"/>
```

## Var: Provider

Required provider name used only with the Windows NT LAN Manager authentication protocol.

### Tag Syntax

```
<Var Provider="string"/>
```

### Possible Values

Use a provider name such as NTLM.

### Example

```
<Var Provider="NTLM"/>
```

## Var: Group

Optional group name used only with the Windows NT LAN Manager authentication protocol.

### Tag Syntax

```
<Var Group="string"/>
```

### Possible Values

Use the same name as the value of the Group variable in Windows NT LAN Manager.

### Example

```
<Var Group="proxyusers"/>
```

## Proxy Authentication

Helix Proxy allows administrators to authenticate each client access through the proxy. Once proxy authentication is enabled, the client will be prompted to provide a valid user name and password prior to gaining access to a particular URL. The ProxyAuthentication list provides the means to:

- Enable the authentication feature (Var: Enabled).
- Choose a specific database for authentication (Var: DatabaseID).
- Choose a specific realm for authentication (Var: Realm).
- Allow users to view the same content from more than one site (Var: AllowDuplicateIDs).
- Identify those sites for which Helix Proxy will not require authentication (List: RuleList).

**For More Information:** See the authentication chapter of *Helix Proxy Administration Guide*.

## Proxy Authentication Syntax

The following sample demonstrates the markup that defines proxy authentication operation:

```
<List Name="ProxyAuthentication">
  <Var Enabled="0"/>
  <List Name="Authority">
    <Var Realm="example.com.ConnectRealm"/>
    <Var DatabaseID="Connect_RN5"/>
    <Var AllowDuplicateIDs="0"/>
  </List>
  <List Name="RuleList">
    <List Name="Rule1">
      <Var NoAuthenticateHost="*.example.com"/>
    </List>
    <Var PluginID="rn-auth-basic"/>
  </List>
</List>
```

### List: ProxyAuthentication

Use this list to enable authentication, and to choose features to validate users.

```
<List Name="ProxyAuthentication">
  ...authentication features...
</List>
```

#### Registry Values

config.ProxyAuthentication

### Var: Enabled

Enables or disables authentication.

#### Tag Syntax

```
<Var Enabled="Boolean"/>
```

#### Possible Values

- 0 – Authentication disabled.
- 1 – Authentication enabled.

#### Example

```
<Var Enabled="1"/>
```

**List: Authority**

Use this list to designate an authentication database and realm to be used, and whether to allow users to log in from more than one location.

```
<List Name="Authority">
    ...database and realm used for authentication...
</List>
```

**Registry Values**

```
config.ProxyAuthentication.Authority
```

**Var: DatabaseID**

Name of the specific database to use with authentication.

**Tag Syntax**

```
<Var DatabaseID="string"/>
```

**Possible Values**

Use the name of an existing database defined with List: Database Name.

**Example**

```
<Var DatabaseID="Connect_RN5"/>
```

**Var: Realm**

Name of the specific realm to use with authentication.

**Tag Syntax**

```
<Var Realm="string.string"/>
```

**Possible Values**

Use the name of an existing realm defined under List: Realm Name.

**Example**

```
<Var Realm="CUV4X-2K.ConnectRealm"/>
```

## Var: AllowDuplicateIDs

Optionally, you can allow users to use more than one client and view content from more than one location. You can also use this variable as a method of limiting access to groups. For example, you could set the AllowDuplicateIDs variable to 1 and assign all marketing employees one user name and password, then the entire department could then use this account to view content.

### Tag Syntax

```
<Var AllowDuplicateIDs="Boolean"/>
```

### Possible Values

- 0 – Do not allow users to log in from multiple locations.
- 1 – Allow users to log in from multiple locations.

### Example

```
<Var AllowDuplicateIDs="1"/>
```

## List: RuleList

This list defines sites that all users are allowed to visit without having to supply a user name and password.

```
<List Name="RuleList">  
  ...rules for non-authenticated access...  
</List>
```

### Registry Values

```
config.ProxyAuthentication.RuleList
```

## List: Rule Name

Use each instance of this list to contain one rule defining a site or sites where you will not require authentication.

```
<List Name="Rule Name1">  
  ...first set of non-authenticated sites or hosts...  
</List>  
<List Name="Rule Name2">  
  ...second set of non-authenticated sites or hosts...  
</List>
```

## Registry Values

config.ProxyAuthentication.RuleList.*string*

## Example

```
<List Name="Rule1">
```

## Var: NoAuthenticateHost

When authentication is enabled, indicates sites that users can access without a username and password.

## Tag Syntax

```
<Var NoAuthenticateHost="string" />
```

## Possible Values

The following are examples of values you can use. Use an asterisk (\*) as a wildcard:

- \*.example.com
- /sports/
- example\_file.rm
- \*.example.com/sports/\*\_file.rm

## Example

```
<Var NoAuthenticateHost="*.example.com" />
```



## ACCESS AND ERROR LOGS

Helix Proxy can record information about client connections and other events in logs written to files, displayed on the screen, or sent to outputs such as TCP sockets. This chapter describes the variables you can configure to control the amount and type of statistics gathered in logs.

**For More Information:** See the basic and advanced logging chapters of *Helix Proxy Administration Guide*. For information about RSS statistics, refer to *Helix Server and Helix Proxy Troubleshooting Guide*.

### Basic Access Log

The basic access log records access attempts by media players, as well as changes to configuration made through Helix Administrator. Output is written to a text file only. Most information is defined within the main LoggingTemplates list.

### Basic Logging Variables

The format of the basic access log is defined through a logging template, as described in “Basic Access Log Syntax” on page 151. Two standalone variables, however, also affect how the basic access log operates:

```
<Var LoggingStyle="5"/>  
<Var StatsMask="3"/>
```

### Var: LoggingStyle

Determines the over information Helix Proxy logs on each access request. For a description of the possible logging styles, see the logging styles section in the basic logging chapter of *Helix Proxy Administration Guide*.

### Tag Syntax

```
<Var LoggingStyle="Integer" />
```

### Possible Values

- 0, 1, 2, 3, 4, 5, 6 – The default value is logging style 5.

### Example

```
<Var LoggingStyle="5" />
```

## Var: StatsMask

Defines what information about the media player Helix Proxy logs on each access request. For a description of the possible client statistics reports, see the client statistics section in the basic logging chapter of *Helix Proxy Administration Guide*.

### Tag Syntax

```
<Var StatsMask="Integer" />
```

### Possible Values

There are four client statistics sets, represented by the integers 1, 2, 4, and 8. You can record any combination of the statistics sets by adding the integer values together as needed:

- 1 – statistics 1
- 2 – statistics 2
- 3 – statistics 1 and 2
- 4 – statistics 3
- 5 – statistics 1 and 3
- 6 – statistics 2 and 4
- 7 – statistics 1, 2, and 3
- 8 – statistics 4 (default)
- 9 – statistics 1 and 4
- 10 – statistics 2 and 4
- 11 – statistics 1, 2, and 4

- 12 – statistics 3 and 4
- 13 – statistics 1, 3, and 4
- 14 – statistics 2, 3, and 4
- 15 – statistics 1, 2, 3, and 4

### Example

```
<Var StatsMask="7"/>
```

## Basic Access Log Syntax

```
<List Name="LoggingTemplates">
  <List Name="Basic Access Log">
    <Var Type="AccessLog"/>
    <Var Enabled="1"/>
    <!-- LoggingStyle is configurable only for the AccessLog template type -->
    <Var LoggingStyle="5"/>
    <List Name="Outputs">
      <List Name="AccessLog">
        <Var LogRollSize="1"/>
        <Var Filename="Logs/rmaccess.log"/>
        <Var LogRollFrequency="10H"/>
        <Var Type="File"/>
      </List>
    </List>
  </List>
  ...other logging templates...
</List>
```

## List: Basic Access Log

This list defines the logging template. The name appears in the logging page in Helix Administrator.

```
<List Name="Basic Access Log">
  ...access log information here...
</List>
```

### Registry Value

```
config.LoggingTemplates.Basic Access Log
```

## Var: Type

Indicates the type of logging template.

### Tag Syntax

```
<Var Type="AccessLog"/>
```

### Possible Values

The AccessLog value is required.

## Var: Enabled

Turns logging on or off.

### Tag Syntax

```
<Var Enabled="Boolean"/>
```

### Possible Values

- 0 – logging off
- 1 – logging on

### Example

```
<Var Enabled="1"/>
```

## Var: Logging Style

Sets the logging style, which determines what information is recorded.

### Tag Syntax

```
<Var LoggingStyle="Integer"/>
```

### Possible Values

- 0, 1, 2, 3, 4, 5, or 6

### Example

```
<Var LoggingStyle="5"/>
```

## List: Outputs

This list determines the types of outputs for the basic access log. Only a single, text file output is allowed:

```
<List Name="Outputs">
  ...basic access log output defined here...
</List>
```

### Registry Value

```
config.LoggingTemplates.Basic Access Log.Outputs
```

## List: Access Log

This list defines each output for the basic access log. Only a single, text-based log is allowed:

```
<List Name="AccessLog">
  ...basic access log output values defined here...
</List>
```

### Registry Value

```
config.LoggingTemplates.Basic Access Log.Outputs.AccessLog
```

### Example

```
<List Name="AccessLog">
  <Var LogRollSize="1"/>
  <Var Filename="Logs/rmaccess.log"/>
  <Var LogRollFrequency="10H"/>
  <Var Type="File"/>
</List>
```

## Var: LogRollSize

Sets the size in Megabytes that the log file can reach before a new file is created. If unused, the log rolling frequency is used.

**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.

### Tag Syntax

```
<Var LogRollSize="Integer" />
```

### Possible Values

- 1 to maximum file system file size

### Example

```
<Var LogRollSize="5" />
```

## Var: Filename

Sets the path and file name of the output log file.

### Tag Syntax

```
<Var Filename="path" />
```

### Possible Values

An absolute path, or a path relative to the Helix Proxy main directory.

### Example

```
<Var Filename="Logs/rmaccess.log" />
```

## Var: LogRollFrequency

Sets the time after which a new log file is created. If unused, the log rolling size is used.

### Tag Syntax

```
<Var LogRollFrequency="IntegerH|D|W|M" />
```

### Possible Values

An integer followed by a letter sets a value in hours, days, weeks, or months:

- 1-40
- H – hours
- D – days
- W – weeks
- M – months

**Example**

```
<Var LogRollFrequency="10D"/>
```

**Var: Type**

Sets the type of output. Only a text file is allowed.

**Tag Syntax**

```
<Var Type="File">
```

**Possible Values**

- File

**Basic Error Log**

The basic error log records errors encountered by Helix Proxy. Output is written to a text file only. All information is defined within the main LoggingTemplates list.

**Basic Error Log Syntax**

```
<List Name="LoggingTemplates">
  ...other logging templates...
  <List Name="Basic Error Log">
    <Var Type="ErrorLog"/>
    <Var Enabled="1"/>
    <List Name="Outputs">
      <List Name="ErrorLog">
        <Var LogRollSize="1"/>
        <Var Filename="Logs/rmerror.log"/>
        <Var LogRollFrequency="10H"/>
        <Var Type="File"/>
      </List>
    </List>
  </List>
  ...other logging templates...
</List>
```

**List: Basic Error Log**

This list defines the logging template. The name appears in the logging page in Helix Administrator.

```
<List Name="Basic Error Log">  
  ...error log information here...  
</List>
```

### Registry Value

config.LoggingTemplates.Basic Error Log

### Var: Type

Indicates the type of logging template.

#### Tag Syntax

```
<Var Type="ErrorLog" />
```

#### Possible Values

The ErrorLog value is required.

### Var: Enabled

Turns logging on or off.

#### Tag Syntax

```
<Var Enabled="Boolean" />
```

#### Possible Values

- 0 – logging off
- 1 – logging on

#### Example

```
<Var Enabled="1" />
```

### List: Outputs

This list determines the types of outputs for the basic error log. Only a single, text file output is allowed:

```
<List Name="Outputs">  
  ...basic error log output defined here...  
</List>
```

## Registry Value

config.LoggingTemplates.Basic Error Log.Outputs

## List: Error Log

This list defines each output for the basic error log. Only a single, text-based log is allowed:

```
<List Name="ErrorLog">
  ...basic error log output values defined here...
</List>
```

## Registry Value

config.LoggingTemplates.Basic Error Log.Outputs.ErrorLog

## Example

```
<List Name="ErrorLog">
  <Var LogRollSize="1"/>
  <Var Filename="Logs/rmerror.log"/>
  <Var LogRollFrequency="10H"/>
  <Var Type="File"/>
</List>
```

## Var: LogRollSize

Sets the size in Megabytes that the log file can reach before a new file is created. If unused, the log rolling frequency is used.

**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.

## Tag Syntax

```
<Var LogRollSize="Integer"/>
```

## Possible Values

- 1 to maximum file system file size

### Example

```
<Var LogRollSize="5"/>
```

## Var: Filename

Sets the path and file name of the output log file.

### Tag Syntax

```
<Var Filename="path"/>
```

### Possible Values

An absolute path, or a path relative to the Helix Proxy main directory.

### Example

```
<Var Filename="Logs/rmerror.log"/>
```

## Var: LogRollFrequency

Sets the time after which a new log file is created. If unused, the log rolling size is used.

### Tag Syntax

```
<Var LogRollFrequency="IntegerH|D|W|M"/>
```

### Possible Values

An integer followed by a letter sets a value in hours, days, weeks, or months:

- 1-40
- H – hours
- D – days
- W – weeks
- M – months

### Example

```
<Var LogRollFrequency="12H"/>
```

## Var: Type

Sets the type of output. Only a text file is allowed.

### Tag Syntax

```
<Var Type="File">
```

### Possible Values

- File

## Advanced Logging Templates

In addition to the predefined basic access and error logs, you can define any number of advanced templates that can send information to various outputs, such as text files, the screen console, or a TCP socket. There are four types of advanced logging templates:

- watch templates

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 templates

A client statistics template periodically reports statistics from all media player connections. This type of template can generate reports about the number of resent packets and the media players' average bit rates, for example. You can generate a report when each media player disconnects, or at periodic intervals, such as every minute.

- session templates

When a system component connects or disconnects, Helix Proxy 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.

- interval templates

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 Proxy health, for example.

**Tip:** The different types of templates have similar definitions, but may include unique variables. The following sections list the logging template types that use each list or variable.

**For More Information:** See the advanced logging chapter of *Helix Proxy Administration Guide*.

## Advanced Logging Syntax

The following is a sample of an advanced logging template. All information is defined within the main LoggingTemplates list.

```
<List Name="LoggingTemplates">
  ...other logging templates...
  <List Name="Extended Logging">
    <Var Type="Session"/>
    <Var Enabled="1"/>
    <Var Format="%time% %Client.*.Addr% %Client.*.Protocol%
      %Client.*.RequestMethod% %Client.*.Session.*.URL%
      %Client.*.User-Agent%"/>
    <Var Interval="00:01:00"/>
    <List Name="Outputs">
      <List Name="StdOut1">
        <Var Type="StdOut"/>
      </List>
    </List>
    <Var Description="This template reports player access information"/>
    <Var WatchRoot="BroadcastReceiver.Statistics.*"/>
    <Var DelFormat="%time%"/>
  </List>
  ...other logging templates...
</List>
```

### List: Name

This list defines the logging template. The user-defined name appears in the logging page in Helix Administrator.

```
<List Name="Name">
  ...customized logging information here...
</List>
```

### Registry Value

config.LoggingTemplates.*Name*

### Logging Template Types

- watch
- session
- interval
- client stats

#### Example

```
<List Name="Extended Logging">
  ...customized logging information here...
</List>
```

### Var: Type

Indicates the type of logging template.

#### Tag Syntax

```
<Var Type="Value"/>
```

#### Possible Values

- Watch
- Session
- Interval
- ClientStats

#### Example

```
<Var Type="Watch"/>
```

### Var: Enabled

Turns logging on or off.

#### Tag Syntax

```
<Var Enabled="Boolean"/>
```

#### Logging Template Types

- watch

- session
- interval
- client stats

#### Possible Values

- 0 – logging off
- 1 – logging on

#### Example

```
<Var Enabled="1"/>
```

### Var: Format

Various variables define the format of the advanced logging template. The format can include plain text and Helix Proxy registry values.

#### Tag Syntax

```
<Var Format="Text"/>
```

#### Logging Template Types

- watch – The variable name is Format.
- session – The variable name is AddFormat for a watch set when the session begins. It is DelFormat for a watch set when the session ends. You can use one or both variables.
- interval – The variable name is Format.
- client stats – The variable name is FormatOnTimer for client statistics that are periodically reported. It is FormatOnDisconnect for client statistics reported when the media player stops the presentation. You can use one or both variables.

**Tip:** To log the same report periodically, and when the client disconnects, you can use a single variable named Format.

#### Possible Values

Boilerplate text and Helix Proxy registry values surrounded by percentage signs (%).

**Example**

```
<Var Format="%time% %Client.*.Addr% %Client.*.Protocol%
%Client.*.RequestMethod% %Client.*.Session.*.URL% %Client.*.User-Agent%"/>
```

**Var: Interval**

Sets the frequency that the log report is generated.

**Tag Syntax**

```
<Var Interval="Time"/>
```

**Logging Template Types**

- watch
- interval
- client stats

**Possible Values**

- HH:MM:SS

**Example**

```
<Var Interval="00:01:00"/>
```

**Var: Description**

Sets a description for the template that appears in Helix Administrator.

**Tag Syntax**

```
<Var Description="Text"/>
```

**Logging Template Types**

- watch
- session
- interval
- client stats

**Possible Values**

Any text.

### Example

```
<Var Description="This template reports player access information"/>
```

## Var: WatchRoot

Determines the connection type to watch for a Session template.

### Tag Syntax

```
<Var WatchRoot="Text"/>
```

### Logging Template Types

- session

### Possible Values

- BroadcastReceiver.Statistics.\* – splitting receivers
- BroadcastDistribution.Statistics.\* – splitting transmitters
- LiveConnections.\* – live broadcast encoders
- LiveArchiving.Archiver.\* – live broadcast archiving
- Server.ConfigLog.\* – configuration file changes

### Example

```
<Var Description="Var WatchRoot="BroadcastDistribution.Statistics.*"/>
```

## List: Outputs

This list determines the types of outputs for the advanced logging template. Each report can have multiple outputs, such as writing log entries to a text file and the console.

```
<List Name="Outputs">  
  ...log outputs defined here...  
</List>
```

### Registry Value

```
config.LoggingTemplates.Name.Outputs
```

## List: OutputTypeN

This list sets each output for the custom logging template. The list name can be user-defined. If no user-defined name is provided, Helix Administrator creates a name that varies depending on the type of output chose. Each name is appended with a number to indicate

```
<List Name="OutputType">
  ...output values defined here...
</List>
```

### Registry Value

```
config.LoggingTemplates.Name.Outputs.OutputTypeN
```

### Possible List Names

If the list name is not user-defined, Helix Administrator creates a list with one of the following names:

- File*N* – standard output file
- HTTPPost*N* – HTTP POST operation
- NTEventLog*N* – Windows NT event logger (Windows Only)
- Pipe*N* – UNIX system pipe (UNIX only)
- StdErr*N* – standard error output, typically a console window
- StdOut*N* – standard output, typically a console window
- SysLog*N* – UNIX system log (UNIX only)
- TCP*N* – outbound TCP port
- TCPListen*N* – inbound TCP port
- UDP*N* – outbound UDP port
- UDPMCast*N* – outbound multicast UDP port

### Example

```
<List Name="Outputs">
  <List Name="StdOut1">
    ...output variables defined here...
  </List>
</List>
```

## List Output Variables

Each output list defines variables for the output type. For example, an output list that writes the log output to an outbound TCP port defines the outgoing IP address and the TCP listen port. The following sections explain the variables that are set depending on the output list type.

### File

The file output list defines the following variables:

- **Filename** – Name of the file in a path relative to the Helix Server installation directory.
- **LogRollFrequency** – Frequency to start a new log file. For details, see “Var: LogRollFrequency” on page 154.
- **LogRollSize** – File size in Megabytes at which to start a new log file. For details, see “Var: LogRollSize” on page 153.
- **Type** – Type of output. Value must be File.

### Example

```
<List Name="File1">  
  <Var Filename="output.txt"/>  
  <Var LogRollFrequency="10H"/>  
  <Var LogRollSize="1"/>  
  <Var Type="File"/>  
</List>
```

### HTTPPost

An HTTP POST output defines the following variables:

- **Dest** – DNS name, IPv4 address, or IPv6 address of the destination server, along with the path to the CGI program that accepts the POST information.
- **Port** – HTTP port of the destination server.
- **Type** – Type of output. Value must be HTTPPost.

**Example**

```
<List Name="HTTPPost1">
  <Var Dest="logger.example.com/cgi-bin/report.py"/>
  <Var Port="8080"/>
  <Var Type="HTTPPost"/>
</List>
```

**NTEventLog**

The Windows NT event logger output defines the following variables:

- Priority – NT event priority. Can be one of the following:
  - LOG\_ERR
  - LOG\_INFO
  - LOG\_WARNING
- Type – Type of output. Value must be NTEventLog.

**Example**

```
<List Name="NTEventLog1">
  <Var Priority="LOG_INFO"/>
  <Var Type="NTEventLog"/>
</List>
```

**Pipe**

The UNIX pipe output defines the following variables:

- Command – UNIX command to pipe to.
- Type – Type of output. Value must be Pipe.

**Example**

```
<List Name="Pipe1">
  <Var Type="Pipe"/>
  <Var Command="/usr/bin/postprocessor.pl"/>
</List>
```

**StdErr**

The standard error output defines the following variable:

- Type – Type of output. Value must be StdErr.

**Example**

```
<List Name="StdErr1">  
  <Var Type="StdErr"/>  
</List>
```

**StdOut**

The standard output defines the following variable:

- Type – Type of output. Value must be StdOut.

**Example**

```
<List Name="StdOut1">  
  <Var Type="StdOut"/>  
</List>
```

**SysLog**

The UNIX system log output defines the following variables:

- Ident – Any user-defined identifier.
- Type – Type of output. Value must be SysLog.
- Priority – One of the following system priorities:
  - LOG\_ALERT
  - LOG\_CRIT
  - LOG\_EMERG
  - LOG\_ERR
  - LOG\_DEBUG
  - LOG\_INFO
  - LOG\_NOTICE
  - LOG\_WARNING

**Example**

```
<List Name = "SysLog1">  
  <Var Type = "Syslog"/>  
  <Var Ident = "RealServerOutput"/>  
  <Var Priority = "LOG_INFO"/>  
</List>
```

## TCP

A TCP socket output defines the following variables:

- Dest – IP address or DNS name of the destination socket.
- Port – TCP port of the destination server.
- Type – Type of output. Value must be TCP.

### Example

```
<List Name="TCP1">
  <Var Dest="111.112.34.107"/>
  <Var Port="4567"/>
  <Var Type="TCP"/>
</List>
```

## TCPListen

A TCP listen output defines the following variables:

- Port – TCP listen port of the destination.
- Type – Type of output. Value must be TCPListen.

### Example

```
<List Name="TCPListen1">
  <Var Port="9876"/>
  <Var Type="TCPListen"/>
</List>
```

## UDP

A UDP output defines the following variables:

- Dest – IP address or DNS name of the destination.
- Port – UDP port of the destination.
- Type – Type of output. Value must be UDP.

### Example

```
<List Name="UDP1">
  <Var Dest="111.112.34.107"/>
  <Var Port="18765"/>
  <Var Type="UDP"/>
</List>
```

## UDPMCast

A UDP multicast output defines the following variables:

- Dest – Multicast IP address of the destination.
- Port – UDP port of the destination.
- Type – Type of output. Value must be UDPMCast.

### Example

```
<List Name="UDPMCast1">  
  <Var Dest="239.255.255.255"/>  
  <Var Port="9098"/>  
  <Var Type="UDPMCast"/>  
</List>
```

## Simple Network Monitoring Protocol (SNMP) Configuration

Using the Simple Network Monitoring Protocol (SNMP), you can monitor Helix Proxy from an SNMP management system. This allows you to change Helix Proxy configuration from a third-party tool, and send notice of important events to an external program.

**For More Information:** See the SNMP chapter of *Helix Proxy Administration Guide* for background information about SNMP, as well as instructions for configuring the SNMP master agent.

## SNMP Syntax

The following sample illustrates SNMP configuration:

```
<List Name="SNMP">  
  <Var Activate="0"/>  
  <Var MasterAddress="127.0.0.1/705"/>  
  <Var SendTrap="1"/>  
  <Var SNMPTrapInterval="20"/>  
  <List Name="TrapThresholds">  
    <Var GlobalCPUUtilization="70"/>  
    <Var ServerStart="1"/>  
    <Var MemoryUsageWaterMark_1="128000"/>  
    <Var MemoryUsageWaterMark_2="180000"/>  
    <Var MemoryUsageWaterMark_3="210000"/>  
    <Var OutBandwidthWaterMark_1="4000"/>  
    <Var OutBandwidthWaterMark_2="7000"/>
```

```

    <Var OutBandwidthWaterMark_3="9000"/>
    <Var MaxConnections="500"/>
  </List>
</List>

```

## List: SNMP

This list enables the SNMP plug-in. It should not be contained within any other lists:

```

<List Name="SNMP">
  ...SNMP plug-in operation defined here...
</List>

```

### Registry Value

config.SNMP

## Var: Activate

Enables or disables SNMP on Helix Proxy.

### Tag Syntax

```
<Var Activate="Boolean"/>
```

### Possible Values

- 0|False - Disable the SNMP plug-in.
- 1|True - Enable the SNMP plug-in.

**Note:** The SNMP feature must also be licensed to operate on Helix Proxy.

### Example

```
<Var Activate="1"/>
```

## Var: Master Address

DNS or IPv4 address and port of the master agent.

### Tag Syntax

```
<Var MasterAddress="Address/port"/>
```

### Possible Values

IP address and port. The values must match the values of the `ManagerAddress` and `AgentXProtocolPort` variables in the master agent configuration file (`master.cfg`). The default port is 705.

**Tip:** The agent typically resides on the Helix Proxy machine in the Helix Proxy Bin directory, so you can generally use the localhost address of 127.0.0.1.

**Note:** IPv6 addressing is not supported.

### Example

```
<Var MasterAddress="127.0.0.1/705"/>
```

## Var: SendTrap

Enables the SNMP plug-in to trap events and send notification of these events to the SNMP management system by way of the master agent.

### Tag Syntax

```
<Var SendTrap="Boolean"/>
```

### Possible Values

- 0|False – Do not trap Helix Proxy events.
- 1|True – Trap Helix Proxy events and send to the management system.

### Example

```
<Var SendTrap="1"/>
```

## Var: SNMPTrapInterval

Determines how frequently a trap is triggered while the trap condition remains active.

### Tag Syntax

```
<Var SNMPTrapInterval="Integer"/>
```

**Possible Values**

Integer that indicates a frequency in seconds. For example, if the interval is 30 and you trap CPU utilization, the trap triggers every 30 seconds as long as the CPU stays above the specified level of use. The default is 20.

**Example**

```
<Var SNMPTrapInterval="30"/>
```

**List: TrapThresholds**

This list defines the traps triggered by the SNMP plug-in:

```
<List Name="TrapThresholds">
  ...SNMP traps defined here...
</List>
```

**Registry Value**

```
config.SNMP.TrapThresholds
```

**Var: GlobalCPUUtilization**

Sets a trap when total machine CPU utilization reaches a specified percentage.

**Tag Syntax**

```
<Var GlobalCPUUtilization="Integer"/>
```

**Possible Values**

Integer from 0 to 100 that indicates the percentage of CPU utilization that triggers the trap. The default is 70.

**Tip:** CPU utilization is reported by the operating system. On multiprocessor machines, this figure represents the aggregate load for all processors.

**Example**

```
<Var GlobalCPUUtilization="75"/>
```

**Var: ServerStart**

Sets a trap when Helix Proxy restarts.

### Tag Syntax

```
<Var ServerStart="Boolean"/>
```

### Possible Values

- 0|False – Do not trap Helix Proxy restarts.
- 1|True – Trap Helix Proxy restarts.

### Example

```
<Var ServerStart="1"/>
```

## Var: MemoryUsageWaterMark\_n

Three variables allow you to set up to three traps reported when Helix Proxy memory usage rises above a specified amount.

### Tag Syntax

```
<Var MemoryUsageWaterMark_n="Kilobytes"/>
```

### Possible Values

Integer value for the number of Kilobytes of memory use that triggers the trap. If you set 128000, 180000, and 230000, for example, the SNMP plug-in reports when memory usage exceeds approximately 128, 180, and 230 Megabytes.

**Tip:** Set the traps at 50, 70, and 90 percent, respectively, of memory allotted to Helix Proxy. The values shown above represent these percentages on a machine in which Helix Proxy has a dedicated allotment of 256 Megabytes.

### Example

```
<Var MemoryUsageWaterMark_1="128000"/>  
<Var MemoryUsageWaterMark_2="180000"/>  
<Var MemoryUsageWaterMark_3="230000"/>
```

## Var: OutBandwidthWaterMark\_n

Defines up to three traps that trigger when the Helix Proxy outgoing bandwidth use rises above the amount specified.

### Tag Syntax

```
<Var OutBandwidthWaterMark_n="Kilobits per second"/>
```

### Possible Values

Integer value for bandwidth in Kilobits per second (Kbps) that triggers the trap. If you set 2000, 4000, and 8000, for example, the SNMP plug-in reports when memory usage exceeds approximately 2, 4, and 8 Megabits per second.

### Example

```
<Var OutBandwidthWaterMark_1="2000"/>
<Var OutBandwidthWaterMark_2="4000"/>
<Var OutBandwidthWaterMark_3="8000"/>
```

## Var: MaxConnections

Sets a trap when a certain number of streaming connections has been established.

### Tag Syntax

```
<Var MaxConnections="Integer"/>
```

### Possible Values

Integer that indicates the number of simultaneous connections that triggers the trap. The default is 500.

### Example

```
<Var MaxConnections="1000"/>
```





## REGISTRY PROPERTIES

The following chapters describe the Helix Proxy registry properties that you can add to advanced logging reports.



## REGISTRY PROPERTIES

At startup, Helix Proxy reads the configuration file values and loads them into its registry, which is distinct from the main registry on Windows operating systems. Registry values record the Helix Proxy configuration, and record system information that you can extract through advanced logging templates.

**For More Information:** For instructions about customizing report templates, refer to “Advanced Logging Templates” on page 159, as well as the advanced logging chapter of *Helix Proxy Administration Guide*.

### Viewing the Registry

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 then entering your user name and password for Helix Administrator. Typically, you can use the local host address (127.0.0.1), along with the Helix Administrator port chosen during installation of Helix Proxy:

```
http://address:AdminPort/admin/regview.html
```

The menu on the left provides access to the registry values. The config category lists the configuration values. For example, the path for the FSMount list is the following:

```
config.FSMount
```

To view the FSMount information, expand the config list by clicking its plus sign (+) in the registry viewer. You can then expand the FSMount information by clicking its icon, and so on. When you click a list entry, the variables and values defined for that list appear in the frame on the right. Additional categories record other registry values not related to configuration, such as information about each media player currently receiving a stream.

**Tip:** The Helix Proxy registry viewer can be a useful diagnostic tool to determine if Helix Proxy has loaded into memory the values that you have changed in the configuration file.

## Date and Time Values

Helix Proxy includes several global properties that you can add to advanced reports to include basic information, such as the time of day. These properties do not correspond to actual registry values, however.

### Date (MM/DD/YY)

Indicates the current date in the format MM/DD/YY.

#### Report Format

%Date%

#### Example

For example, the date, December 17, 2005, is written as follows:

12/17/05

### Time of day by local time zone (HH:MM:SS)

Provides the current time of day in the local time zone in the format HH:MM:SS.

#### Report Format

%Time%

#### Example

The local time 1:36.52 P.M. is represented as follows:

13:36:52

### Greenwich Mean Time (HH:MM:SS)

Displays the current Greenwich Mean Time (GMT) in the format HH:MM:SS.

#### Report Format

%GMTIME%

**Example**

For example, the GMT time written as:

17:43:33

is the same time as 9:43.33 AM Pacific Standard Time.

**Local Time Zone Difference**

Indicates the difference between local time and Greenwich Mean Time.

**Report Format**

%TZDiff%

**Example**

Pacific Standard Time, which is eight hours earlier than GMT, is represented as the following:

-0800

**Hour of the day by local time zone (HH)**

Displays the current hour by local time zone in the format HH.

**Report Format**

%Hour%

**Example**

The hour for the local time 1:36.52 P.M. is represented as follows:

13

**Minute of the day (MM)**

Indicates the current minute in the format MM.

**Report Format**

%Min%

**Example**

The minute for the local time 1:36.52 P.M. is represented as follows:

36

## Second of the day (SS)

Adds the current second in the format SS.

### Report Format

`%Sec%`

### Example

The second for the local time 1:36.52 P.M. is represented as follows:

52

## Report Formatting Values

The following properties help you to format reports.

### Double Quote

The double quote global value inserts double quotation marks into an advanced logging template.

### Report Format

`%Quot%`

### Example

The following entry in an advanced logging template:

`%Quot%This sentence is enclosed in quotation marks.%Quot%`

appears as the following in the report output:

“This sentence is enclosed in quotation marks.”

### New Line

The new line indicator moves the following report output to a new line. Carriage returns you enter in a report that you are creating through Helix Administrator are also recognized as new lines.

### Report Format

`\n`

**Example**

`\n` This is a new line.

**Tab**

The tab indicator inserts a tab in the report output.

**Report Format**

`\t`

**Example**

The following text is separated by a tab:`\t` The tabbed text follows.

## Obsolete Registry Properties

The following registry properties are obsolete and no longer used in Helix Proxy:

- `Server.PNAclientCount`
- `Server.Errors.LastEmerg`
- `Server.Errors.LastAlert`
- `Server.Errors.LastCrit`
- `Server.Errors.LastErr`
- `Server.Errors.LastWarning`
- `Server.Errors.LastNotice`
- `Server.Errors.LastInfo`
- `Server.Errors.LastDebug`



## CLIENT PROPERTIES

This chapter explains the client and client session properties. The client properties record information about media players. The client session properties record properties for each requested media stream.

**Note:** To prevent adverse effects on Helix Proxy performance, the client properties are no longer stored in the registry and are not accessible through the registry viewer. You can view client properties through customized reports, however, using the report formats described in this chapter.

### Client Properties

The client properties record data about every media player requesting a stream from Helix Proxy. These properties concern player identification and network settings. When a client connects, Helix Proxy assigns the player a ConnId (see page 188). Until the user shuts down the media player, all client properties are stored under that connection ID. Each stream is then recorded as a separate session with the properties explained in the section “Client Session Properties” on page 196.

#### Media Player Information Properties

The following table summarizes the client properties that supply information about media players.

**Media Player Identification Properties**

Property	Summary
CBID	Optional cookie-based ID for RealNetworks media players.
ClientID	Platform information for RealNetworks media players.
ConnId	Sequential ID assigned to each media player as it connects.
GUID	Globally unique ID that some media players may report.

(Table Page 1 of 2)

**Media Player Identification Properties (continued)**

Property	Summary
Language	Media player's preferred language.
User-Agent	Player and platform information for all media players.

(Table Page 2 of 2)

**Media Player Networking Properties**

The following table summarizes the client properties that provide network data about media players.

**Client Networking Properties**

Property	Summary
ControlBytesSent	Number of bytes sent over the control channel.
IPAddress	Player's IP address.
IsCloaked	Whether the streaming protocol is cloaked as HTTP.
IsRDT	Whether RealNetworks' proprietary data packet format is used.
PlayerStartTime	Connection time according to RealNetworks players.
Port	Port the media players uses to receive data.
Protocol	Application-level protocol such as RTSP used for the stream.
RequestMethod	HTTP-style request method used.
StartTime	Connection time according to the server.
Version	Version of the application-level protocol used.

**CBID**

The CBID value is a randomly generated, unique, cookie-based ID that Helix Proxy attempts to set for RealNetworks media players on the following conditions:

- The server has `config.EnableCookieBasedIDs` enabled.
- The server does not have `config.DisableClientGuid` enabled.
- The player does not send a GUID (see page 188) or sends an all-zero GUID.

The CBID property records a number in the form `nnnnnnnn-nnnn-nnnn-nnnn-nnnnnnnnnn`. Although Helix Proxy attempts to set this for the player, it does not record if the player accepted the cookie. If the player did accept the cookie, it will report the cookie ID on the next access. Otherwise, Helix Proxy again attempts to set another cookie-based ID when the player next connects.

**Tip:** In the standard access log, the [client\_ID] field records the CBID value if it exists. Otherwise, it logs the GUID value or UNKNOWN if neither value is returned.

### Report Format

```
%Client.*.CBID%
```

## ClientID

The ClientID property provides information about RealNetworks clients, such as version and operating system. This property does not record any information for other types of clients.

**For More Information:** See User-Agent on page 194 for information about client information reported by other media players.

**Tip:** In the standard access log, the [client\_info] field records the ClientID value for RealNetworks media players and the User-Agent value for other media players.

### Report Format

```
%Client.*.ClientID%
```

### Possible Values

For RealNetworks clients, ClientID uses the following format:

```
[platform_version_client_type_distribution_language_CPU]
```

The following information is recorded:

<i>platform</i>	Operating system client software runs on, such as WinNT, Mac, and so on.
<i>version</i>	Operating system version number.
<i>client</i>	Version number of the client software.
<i>type</i>	Type of client software.
<i>distribution</i>	Distribution code of the client software.
<i>language</i>	Language setting in client software.
<i>CPU</i>	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]
```

## ConnId

Helix Proxy assigns a unique integer to each client connection, starting with 1 and incrementing by 1 for each new connection. If a user closes the media player, reopens it, and requests a subsequent clip, Helix Proxy assigns the player a new ID. The same media player therefore may have multiple ConnID values, but only one connection will be active at a time. After a Helix Proxy restart, the ConnID values restart at 1.

**Tip:** In the standard access log, the [presentation\_id] field records the connection ID.

### Report Format

```
%Client.*.ConnId%
```

## ControlBytesSent

ControlBytesSent is an integer that indicates the number of bytes sent over the media player's control channel. This data includes statistics reports sent to Helix Proxy, as well as requests for client information made by Helix Proxy.

**Tip:** In the standard access log, the number of control bytes sent is recorded as part of the bytes\_sent field.

**For More Information:** The amount of data bytes sent is recorded by the session property BytesSent (see page 198).

### Report Format

```
%Client.*.ControlBytesSent%
```

## GUID

The GUID property records a globally unique ID reported by the media player. The value is a string in the form nnnnnnnn-nnnn-nnnn-nnnn-nnnnnnnnnnnn, such as the following:

```
3ab16eb2-9f30-4c1f-acb6-25dfba5ba0da
```

If the media player sends an all-zero GUID because GUID reporting is disabled by the user, or if Helix Proxy has `config.DisableClientGuids` set to 1, the GUID consists of zeros:

```
00000000-0000-0000-0000-000000000000
```

If the media player does not generate a GUID at all, a value is not recorded.

**For More Information:** RealNetworks media players may also have a cookie-based ID recorded by the `CBID` property (see page 186).

**Tip:** In the standard access log, the `[client_ID]` field records the `CBID` value if it exists. Otherwise, it logs the GUID value or `UNKNOWN` if neither value is returned.

#### Report Format

```
%Client.*.GUID%
```

### IPAddress

`IPAddress` records the dotted IPv4 or IPv6 address of the client computer.

**Tip:** In the standard access log, `client_address` field records this value.

**For More Information:** The Helix Proxy IP address is recorded under the client session properties as `InterfaceAddress` (see page 201).

#### Report Format

```
%Client.*.IPAddress%
```

### IsCloaked

`IsCloaked` indicates whether the media player is receiving media directly over a streaming protocol such as RTSP or MMS, or whether the streaming protocol is cloaked as HTTP to avoid firewall restrictions.

#### Report Format

```
%Client.*.IsCloaked%
```

### Possible Values

- 0 – Stream protocol is not cloaked.
- 1 – Stream protocol is cloaked as HTTP.

## IsRDT

IsRDT indicates if the RealNetworks proprietary packet format, RDT, is used. If false, another packet format is used, such as the the standards-based RTP packet format.

### Report Format

`%Client.*.IsRDT%`

### Possible Values

- 0 – RTP or another packet format used.
- 1 – RealNetworks RDT packet format used.

## Language

The language property reports a language code that indicates the client's preferred language. All RealNetworks media players report one of the values listed in the following table. Other media players may report the same set of codes, may not report a language, or may report a language using a different code.

### Report Format

`%Client.*.Language%`

### Possible Values

The following table lists the language codes and the associated languages that RealNetworks media players report.

**Language Codes**

Code	Language	Code	Language
af	Afrikaans	es-ni	Spanish (Nicaragua)
ar-ae	Arabic (U.A.E.)	es-pa	Spanish (Panama)
ar-bh	Arabic (Bahrain)	es-pe	Spanish (Peru)

(Table Page 1 of 3)

**Language Codes (continued)**

Code	Language	Code	Language
ar-dz	Arabic (Algeria)	es-pr	Spanish (Puerto Rico)
ar-eg	Arabic (Egypt)	es-py	Spanish (Paraguay)
ar-iq	Arabic (Iraq)	es-sv	Spanish (El Salvador)
ar-jo	Arabic (Jordan)	es-uy	Spanish (Uruguay)
ar-kw	Arabic (Kuwait)	es-ve	Spanish (Venezuela)
ar-lb	Arabic (Lebanon)	et	Estonian
ar-ly	Arabic (Libya)	eu	Basque
ar-ma	Arabic (Morocco)	fi	Finnish
ar-om	Arabic (Oman)	fo	Faeroese
ar-qa	Arabic (Qatar)	fr	French (Standard)
ar-sa	Arabic (Saudi Arabia)	fr-be	French (Belgian)
ar-sy	Arabic (Syria)	fr-ca	French (Canadian)
ar-tn	Arabic (Tunisia)	fr-ch	French (Swiss)
ar-ye	Arabic (Yemen)	fr-lu	French (Luxembourg)
bg	Bulgarian	he	Hebrew
ca	Catalan	hr	Croatian
cs	Czech	hu	Hungarian
da	Danish	in	Indonesian
de	German (Standard)	is	Icelandic
de-at	German (Austrian)	it	Italian (Standard)
de-ch	German (Swiss)	it-ch	Italian (Swiss)
de-li	German (Liechtenstein)	ja	Japanese
de-lu	German (Luxembourg)	ko	Korean
el	Greek	ko	Korean (Johab)
en	English	lt	Lithuanian
en	English (Caribbean)	lv	Latvian
en-au	English (Australian)	nl	Dutch (Standard)
en-bz	English (Belize)	nl-be	Dutch (Belgian)
en-ca	English (Canadian)	no	Norwegian
en-gb	English (British)	pl	Polish
en-ie	English (Ireland)	pt	Portuguese (Standard)

(Table Page 2 of 3)

**Language Codes (continued)**

Code	Language	Code	Language
en-jm	English (Jamaica)	pt-br	Portuguese (Brazilian)
en-nz	English (New Zealand)	ro	Romanian
en-tt	English (Trinidad)	sk	Slovak
en-us	English (United States)	sl	Slovenian
en-za	English (South Africa)	sq	Albanian
es	Spanish (Spain)	sr	Serbian
es-ar	Spanish (Argentina)	sv	Swedish
es-bo	Spanish (Bolivia)	sv-fi	Swedish (Finland)
es-cl	Spanish (Chile)	th	Thai
es-co	Spanish (Colombia)	tr	Turkish
es-cr	Spanish (Costa Rica)	uk	Ukrainian
es-do	Spanish (Dominican Republic)	vi	Vietnamese
es-ec	Spanish (Ecuador)	zh-cn	Chinese (People's Republic)
es-gt	Spanish (Guatemala)	zh-hk	Chinese (Hong Kong)
es-hn	Spanish (Honduras)	zh-sg	Chinese (Singapore)
es-mx	Spanish (Mexican)	zh-tw	Chinese (Taiwan)

(Table Page 3 of 3)

**PlayerStartTime**

PlayerStartTime gives the time that a RealNetworks media player made its initial streaming media request. Other media players do not report this information. The value, which is set according to the media player clock, is reported in the RTSP request headers. It gives the date and time, as well as the offset from Coordinated Universal Time.

**For More Information:** The StartTime property (see page 194) provides the start time as recorded by the Helix Proxy clock for all media players.

**Report Format**

```
%Client.*.PlayerStartTime%
```

**Possible Values**

Any date and time in the following format:

[DD/MM/YYYY:HH:MM:SS -UT]

The following example date and time is for March 28, 2005 at 2:51:11 P.M. The Helix Proxy is eight hours behind Coordinated Universal Time:

[28/03/2005:14:51:11 -08:00]

## Port

The Port value indicates the UDP or TCP port that the media player uses to receive data.

### Report Format

%Client.\*.Port%

## Protocol

Protocol indicates the application-level protocol used to deliver the stream. Cloaked streams are listed under the primary protocol. For example, an RTSP stream cloaked as HTTP is listed as using the RTSP protocol.

**For More Information:** The Version property (see page 195) provides the protocol version. The IsCloaked property (see page 189) indicates that the stream is cloaked.

**Tip:** In the standard access log, the protocol is reported as part of the "GET filename protocol/version" field.

### Report Format

%Client.\*.Protocol%

### Possible Values

- RTSP – RTSP-based communication between Helix Proxy, RealPlayer, and any other RTSP-capable player.
- HTTP – HTTP-based communication between Helix Proxy and media players or Web browsers.
- MMS – MMS-based communication to Windows Media Player.

## RequestMethod

RequestMethod provides the HTTP-style request method used by the media player.

### Report Format

`%Client.*.RequestMethod%`

### Possible Values

- Get – HTTP Get method, or any protocol method other than Post. This value is typically reported.
- Post – HTTP Post method.

## StartTime

The StartTime property records the time when the media player contacted Helix Proxy. The timestamp is set according to the Helix Proxy clock in the following format:

`DD/MMM/YYYY:HH:MM:SS`

For example:

`17/Dec/2003:08:24:29`

**For More Information:** PlayerStartTime (see page 192) gives the client-derived start time for RealNetworks media players. SessionStartTime (see page 207) records the starting times of individual streams requested by the media player.

**Tip:** In the standard access log, the [start\_time] field records the start time.

### Report Format

`%Client.*.StartTime%`

## User-Agent

The User-Agent property reports the user identification sent to Helix Proxy in the headers of the client's RTSP, HTTP, or MMS request.

**Tip:** In the standard access log, the [client\_info] field records the ClientID value for RealNetworks media players and the User-Agent value for all other media players.

### Report Format

```
%Client.*.User-Agent%
```

### Possible Values

The User-Agent value varies for each media player. The following sections provide examples for commonly used players.

#### RealNetworks Players

RealNetworks media players report their version number and operating system, as in the following example:

```
RealMedia Player (HelixDNAClient)/10.0.0.0 (win32)
```

#### Windows Media Player

For Windows Media Player, the User-Agent value records the player version like this:

```
NSPlayer/7.1.0.3055
```

#### QuickTime Player

For QuickTime Player, User-Agent records the player version and the operating system. For example:

```
QTS (qtver=5.0.2;os=Windows NT 5.0)
```

#### Unknown Clients

If Helix Proxy cannot gather the client information from the request headers, UNKNOWN appears as the User-Agent value.

## Version

The version of the application-level protocol (such as RTSP or HTTP) used to deliver the clip. The Protocol property (see page 193) supplies the protocol type.

**Tip:** In the standard access log, the protocol version is reported as part of the "GET filename protocol/version" field.

## Report Format

%Client.\*.Version%

## Client Session Properties

The client session properties record data about the streams that a media player requests. Each stream is indexed under a `SessionID` (see page 207) that falls under the general `ConnId` (see page 188). The streams may be played in sequence, or in parallel through a SMIL presentation. A video stream that includes both audio and video stream data is recorded as a single session.

### Session Information Properties

The following table summarizes the session properties that supply basic information about the media stream.

Media Session Information Properties	
Property	Summary
<code>AvgBitrate</code>	Average bit rate for the session stream.
<code>BytesSent</code>	Number of bytes streamed to the player.
<code>Duration</code>	Length of an on-demand clip in milliseconds.
<code>DurationSeconds</code>	Length of an on-demand clip rounded to seconds.
<code>FileSize</code>	Clip size in bytes.
<code>InterfaceAddress</code>	Helix Proxy IP address used to deliver the stream.
<code>IsMulticastUsed</code>	Whether the stream is multicast.
<code>IsRAStreamFound</code>	Whether the stream is encoded as RealAudio.
<code>IsREStreamFound</code>	Whether the stream includes encoded event URLs.
<code>IsRIStreamFound</code>	Whether the stream includes a clickable image map.
<code>IsRVStreamFound</code>	Whether the stream is encoded as RealVideo.
<code>IsUDP</code>	Whether the stream uses UDP.
<code>PlayerRequestedURL</code>	Full URL requested by the media player.
<code>PlayTime</code>	Time in milliseconds spent in streaming.
<code>ProxyConnectionType</code>	Type of proxied connection used.
<code>SendingTime</code>	Time in seconds from start to end of the session.
<code>SessionID</code>	ID assigned to this stream.
<code>SessionStartTime</code>	Starting timestamp for the stream.
<code>URL</code>	Base URL for the stream.

### Session Quality of Service Properties

The following table summarizes the session properties that provide information about the quality of service for the stream delivery.

**Media Session Quality of Service Properties**

Property	Summary
ASMSubscribes	Number of streaming rulebook subscriptions for RealNetworks players.
ASMUnsubscribes	Number of streaming rulebook unsubscriptions for RealNetworks players.
EstimatedPlayerBufferOverruns	Estimated number of times the player's buffer overfilled.
EstimatedPlayerBufferUnderruns	Estimated number of times the player's buffer ran too low.
FailedResends	Number of packet resends that failed.
LogStats	Player's logging statistics.
PacketsLost	Number of packets that were lost.
PacketsSent	Number of packets sent.
RRFrequency	Average frequency of client receiver reports.
RTT	Total round-trip time for the link.
Status	RTSP status code.
SuccessfulResends	Number of packets successfully resent.
TotalBitrateAdaptations	Number of streaming speed changes.
TotalDownshifts	Number of shifts to a lower streaming speed.
TotalUpshifts	Number of shifts to a higher streaming speed.

### ASMSubscribes

ASMSubscribes indicates the number of times that RealPlayer subscribes to a new Adaptive Stream Management (ASM) rule. These rules allow RealPlayer to modify the stream bandwidth or other stream criteria in response to changing network conditions. (A change to ASMSubscribes therefore does not necessarily indicate a change in the streaming bandwidth.) An ASMSubscribes value is recorded for Helix Proxy rate control, as well as client-side SureStream rate adaptation. For all clients other than RealNetworks media players, the value is always 0.

**For More Information:** See also TotalBitrateAdaptations on page 209. For information about rate control, refer to “Rate Control” on page 72.

### Report Format

```
%Client.*.Session.*.ASMSubscribes%
```

## ASMUnsubscribes

ASMUnsubscribes indicates the number of times that RealPlayer unsubscribes from an ASM rule. For all clients other than RealNetworks media players, the value is always 0. An ASMUnsubscribes value is recorded for Helix Proxy rate control, as well as client-side SureStream rate adaptation.

### Report Format

```
%Client.*.Session.*.ASMUnsubscribes%
```

## AvgBitrate

The AvgBitrate property records the stream’s average bit rate in Kilobytes per second (KB/s).

**Tip:** In the standard access log, the average bit rate is reported in the `average_bit_rate` field.

### Report Format

```
%Client.*.Session.*AvgBitrate%
```

## BytesSent

This property records the number of bytes streamed to the client. This includes media data as well as transport packet overhead. It does not include messages sent to the media player on a control channel. The value is recorded for on-demand clips and live broadcasts.

**Tip:** In the standard access log, the number of bytes sent, as well as control bytes sent (see ControlBytesSent on page 188), is recorded as part of the `bytes_sent` field.

**Note:** The FileSize value (see page 201) reports the actual size of the file on disk. Because FileSize includes file header

information whereas BytesSent includes packet overhead, the two values typically differ by a small degree. If the user stopped an on-demand presentation before its normal completion, the BytesSent value may be significantly smaller than FileSize.

#### Report Format

```
%Client.*.Session.*.BytesSent%
```

### Duration

This property indicates the total length of an on-demand clip in milliseconds. Because Helix Proxy extracts this value from the file at the beginning of the streaming session, the value does not indicate how long the streaming presentation actually lasted. Live broadcasts always have a value of 0.

#### Report Format

```
%Client.*.Session.*.Duration%
```

### DurationSeconds

This property records the same stream duration as the Duration property, rounding it to the nearest second.

**Tip:** In the standard access log, the file\_time field gives the file duration in seconds.

#### Report Format

```
%Client.*.Session.*.DurationSeconds%
```

### EstimatedPlayerBufferOverruns

The EstimatedPlayerBufferOverruns property is used only with Helix Proxy rate control. It indicates the number of times the media player's buffer filled beyond capacity. This condition typically indicates that the network is delivering data faster than the client uses it. This condition generally precedes an upshift in the media rate.

**Tip:** This number of overruns is an estimate only, based on the Helix Proxy buffer model for the player. A value is not recorded if the clip offers a single rate or the session uses the SureStream client-side rate adaptation mechanism.

**For More Information:** For more on server-side rate control, refer to “Rate Control” on page 72.

#### Report Format

```
%Client.*.Session.*.EstimatedPlayerBufferOVERRUNS%
```

### EstimatedPlayerBufferUnderruns

The `EstimatedPlayerBufferUnderruns` property indicates the number of times the media player’s buffer drained. This condition, which results in the player pausing the presentation to rebuffer the content, typically indicates that the network is delivering data slower than the client uses it. This condition generally precedes a downshift in the media rate.

**Tip:** This number of underruns is an estimate only, based on the Helix Proxy buffer model for the player. A value is not recorded if the clip offers a single rate or the session uses the `SureStream` client-side rate adaptation mechanism.

**For More Information:** For more on server-side rate control, refer to “Rate Control” on page 72.

#### Report Format

```
%Client.*.Session.*.EstimatedPlayerBufferUNDERRUNS%
```

### FailedResends

This property records the number of packet resends that failed because the resent packets did not arrive, or the packets arrived too late to be of use. If Helix Proxy does not honor resend requests from clients, this value is always 0.

**Tip:** In the standard access log, the `failed_resends` field reports the number of failed resends.

**For More Information:** `SuccessfulResends` (see page 209) reports the successful resend attempts.

#### Report Format

```
%Client.*.Session.*.FailedResends%
```

## FileSize

This property records the size in bytes of the requested clip on the Helix Proxy file system. For live broadcasts, the value is always 0.

**Note:** The BytesSent (see page 198) value reports the amount of data streamed to the media player. Because FileSize includes file header information whereas BytesSent includes packet overhead, the two values typically differ by a small degree.

**Tip:** In the standard access log, the file\_size field gives the file size in bytes.

### Report Format

```
%Client.*.Session.*.FileSize%
```

## InterfaceAddress

This property stores the IPv4 or IPv6 address that Helix Proxy uses to deliver the session data. Domain names are not recorded.

**Tip:** The media player's IP address is recorded under the general client properties as IPAddress (see page 189).

### Report Format

```
%Client.*.Session.*.InterfaceAddress%
```

## IsMulticastUsed

This property indicates if the session is part of a multicast.

### Report Format

```
%Client.*.Session.*.IsMulticastUsed%
```

### Possible Values

- 0 – Multicast not used. Session was a unicast.
- 1 – Back-channel or scalable multicast used for the streaming session.

## IsRAStreamFound

Indicates the presence of a RealAudio stream in the session. Only RealNetworks media players report this value. For all other media players, the value 0 is recorded.

**Tip:** In the standard access log, this is the first value of the [stream\_components] field.

### Report Format

```
%Client.*.Session.*.IsRAStreamFound%
```

### Possible Values

- 0 – Session contains no RealAudio stream. It may contain an audio stream in a different format, however.
- 1 – Session contains at least one RealAudio stream.

## IsREStreamFound

This indicates the presence of an events stream in a RealMedia session. The events stream causes the clip to open URLs in the viewer's browser automatically during the presentation. Only RealNetworks media players report this value. For all other media players, the value 0 is recorded.

**Tip:** In the standard access log, this is the third value of the [stream\_components] field.

### Report Format

```
%Client.*.Session.*.IsREStreamFound%
```

### Possible Values

- 0 – Session contains no events stream.
- 1 – Session contains an events stream.

## IsRIStreamFound

This property indicates the presence of an image map in the session. An image map contains clickable hot spots that can open links in the viewer's browser. Only RealNetworks media players report this value. For all other media players, the value 0 is recorded.

**Tip:** In the standard access log, this is the fourth value of the [stream\_components] field.

#### Report Format

```
%Client.*.Session.*.IsRISStreamFound%
```

#### Possible Values

- 0 – Session contains no image map stream.
- 1 – Session contains an image map stream.

### IsRVStreamFound

This property indicates the presence of a RealVideo stream in the session. Only RealNetworks media players report this value. For all other media players, the value 0 is recorded.

**Tip:** In the standard access log, this is the second value of the [stream\_components] field.

#### Report Format

```
%Client.*.Session.*.IsRVStreamFound%
```

#### Possible Values

- 0 – Session contains no RealVideo stream. It may contain a video stream in a different format, however.
- 1 – Session contains at least one RealVideo stream.

### IsUDP

Indicates if the media packets are transmitted using User Datagram Protocol (UDP) rather than TCP.

#### Report Format

```
%Client.*.Session%.*.IsUDP
```

#### Possible Values

- 0 – TCP is used.
- 1 – UDP unicast or multicast is used.

## LogStats

The LogStats property records the client statistics generated by the media player. RealPlayer can generate various statistics based on Helix Proxy statistics settings. Windows Media Player can generate a limited set of statistics. If the media player does not generate statistics, UNKNOWN is recorded. Statistics are reported in the same format as they appear in the standard access log.

**For More Information:** For a description of the possible client statistics reports, see the client statistics section in the basic logging chapter of *Helix Proxy Administration Guide*.

### Report Format

```
%Client.*.Session.*.LogStats%
```

## PacketsLost

This property indicates the number of data packets lost, either because the packets did not arrive, or they arrived too late to be of use.

### Report Format

```
%Client.*.Session.*.PacketsLost%
```

## PacketsSent

This property indicates the number of data packets sent by Helix Proxy. It does not include packets sent on a control channel. The total includes packets resent because of client request.

**Tip:** The number of packets received is equal to the number of packets sent minus the number of packets lost.

### Report Format

```
%Client.*.Session.*.PacketsSent%
```

## PlayerRequestedURL

This property supplies the full URL to the clip, broadcast, or HTML page as requested by the client, as in `rtsp://helixserver.example.com/media.rm`. Helix Proxy gathers the value from the request header, such as RTSP DESCRIBE or

HTTP GET/POST. The value may include query parameters, mount points (such as /secure/ for authenticated media access), or Helix Server aliases.

**Tip:** To report on the URL without extra parameters and with aliases resolved, use the URL property (see page 210).

#### Report Format

```
%Client.*.Session.*.PlayerRequestedURL%
```

### PlayTime

Only RealPlayer 11 and later report a value for the PlayTime property. This value represents the cumulative time in milliseconds that the session played as recorded by the client. The time value excludes time spent for initial buffering, for rebuffering, or user-initiated pausing.

**For More Information:** See also SendingTime on page 206.

**Note:** In the standard access log, statistics type 4 reports the PlayTime value.

#### Report Format

```
%Client.*.Session.*.PlayTime%
```

### ProxyConnectionType

This property provides information about the type of proxy stream used (live or on-demand) and how the proxy delivered the media stream.

#### Report Format

```
%Client.*.Session.*.ProxyConnectionType%
```

#### Possible Values

- Live Pass-Through – Live broadcast in which Helix Proxy passed the Helix Server stream directly to the client without splitting it.
- Live Split – Live broadcast in which Helix Proxy received a single stream from Helix Server, then split the stream to the client.
- Demand Pass-Through – On-demand clip served directly from Helix Server without being cached by Helix Proxy.

- Demand Cache Hit – On-demand clip served from the Helix Proxy cache.
- Unknown – Unknown proxy action. Typically means that the client request was made directly to Helix Server rather than through Helix Proxy.
- Accounting Only – Accounting channel request made by Helix Proxy to Helix Server.

## RRFrequency

Used with rate control, this property indicates the average frequency in milliseconds that Helix Proxy has received receiver reports (RTCP RR) from the media client. It is reported only for streams using RTP. For RealPlayer streams using the RealNetworks proprietary RDT packet format, the value is always 0.

**For More Information:** The `RtcpRRratio` configuration variable reserves bandwidth for receiver reports. See “Var: `RtcpRRratio`” on page 78.

### Report Format

```
%Client.*.Session.*.RRFrequency%
```

## RTT

Used with rate control, this property indicates the average round-trip time for data to travel from Helix Proxy to the media client, then back to Helix Proxy. The value is in milliseconds.

### Report Format

```
%Client.*.Session.*.RTT%
```

## SendingTime

This property indicates the time in seconds between initialization of the session and the end of playback. The value includes initial buffering time, any rebuffering time, and any time that the viewer paused the presentation.

Because RealNetworks media players close the connection when the clip reaches the end of its timeline or the viewer stops the clip, the `SendingTime` value expresses the total duration of the 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 session stopped but the player remained idle.

**For More Information:** See also `PlayTime` on page 205.

**Note:** In the standard access log, the `[connected_time]` field records this value.

### Report Format

```
%Client.*.Session.*.SendingTime%
```

## SessionID

Integer representing the ID for this session. The value starts at 1 and increments by 1 for each new session requested by the media player.

### Report Format

```
%Client.*.Session.*.SessionID%
```

## SessionStartTime

`SessionStartTime` provides the time that Helix Proxy began to stream the session, according to the Helix Proxy clock. It uses the following format:

```
[DD/MM/YY:HH:MM:SS -UT]
```

The following example date and time is for March 28, 2005 at 2:51:11 P.M. The Helix Proxy is eight hours behind Coordinated Universal Time:

```
[28/03/2005:14:51:11 -08:00]
```

**For More Information:** `StartTime` (see page 194) provides the time when the media player initially contacted Helix Proxy.

### Report Format

```
%Client.*.Session.*.SessionStartTime%
```

## Status

The `Status` property records the HTTP or RTSP status code.

**For More Information:** The RTSP RFC, available at <http://www.ietf.org/rfc/rfc2326.txt>, defines the RTSP status codes.

## Report Format

%Client.\*.Session.\*.Status%

## Possible Values

RTSP status codes are similar to HTTP 1.1 status codes. Codes in the 200's reflect success. Those in the 300's indicate redirection. The 400-level codes denote client errors, while 500-level codes indicate server errors. The following table summarizes the RTSP codes.

RTSP Status Codes			
Code	Meaning	Code	Meaning
100	Continue	413	Request Entity Too Large
200	OK	414	Request-URI Too Large
201	Created	415	Unsupported Media Type
250	Low on Storage Space	451	Parameter Not Understood
300	Multiple Choices	452	Conference Not Found
301	Moved Permanently	453	Not Enough Bandwidth
302	Moved Temporarily	454	Session Not Found
303	See Other	455	Method Not Valid in This State
304	Not Modified	456	Header Field Not Valid for Resource
305	Use Proxy	457	Invalid Range
400	Bad Request	458	Parameter Is Read-Only
401	Unauthorized	459	Aggregate Operation Not Allowed
402	Payment Required	460	Only Aggregate Operation Allowed
403	Forbidden	461	Unsupported Transport
404	Not Found	462	Destination Unreachable
405	Method Not Allowed	500	Internal Server Error
406	Not Acceptable	501	Not Implemented
407	Proxy Authentication Required	502	Bad Gateway
408	Request Timeout	503	Service Unavailable
410	Gone	504	Gateway Timeout

(Table Page 1 of 2)

**RTSP Status Codes (continued)**

Code	Meaning	Code	Meaning
411	Length Required	505	RTSP Version Not Supported
412	Precondition Failed	551	Option Not Supported

(Table Page 2 of 2)

**SuccessfulResends**

This property records the number of packet resends that succeeded. If Helix Proxy does not honor resend requests from clients, this value is always 0.

**Tip:** In the standard access log, the resends field reports the number of successful resends.

**For More Information:** FailedResends (see page 200) reports the unsuccessful resend attempts.

**Report Format**

```
%Client.*.Session.*.SuccessfulResends%
```

**TotalBitrateAdaptations**

This property value is used with Helix Proxy rate control and media formats that support multiple streaming speeds or multiple codecs. The value indicates the number of times that the media player shifted between the different codecs or bit rates. A value of 0 means that no media format shifting occurred, that the media clip was encoded at a single bit rate, or that client-side stream adaptation (SureStream) was used.

**Note:** For RealNetworks media players, this property also records ASM subscribes and unsubscribes that may not involve changes to the streaming bit rate. See ASMSubscribes on page 197 and ASMUnsubscribes on page 198.

**For More Information:** For information about rate control, refer to “Rate Control” on page 72.

**Report Format**

```
%Client.*.Session.*.TotalBitrateAdaptations%
```

## TotalDownshifts

This property indicates the number of times that Helix Proxy lowered the media rate for the multi-rate clip or broadcast.

**Tip:** For many media players, this value added to TotalUpshifts equals the value for TotalBitrateAdaptations. This may not be the case for RealNetworks media players, however, because the TotalBitrateAdaptations value also reflects ASM rule changes.

### Report Format

```
%Client.*.Session.*.TotalDownshifts%
```

## TotalUpshifts

This property indicates the number of times that Helix Proxy increased the media rate for the multi-rate clip or broadcast.

**Tip:** For many media players, this value added to TotalDownshifts equals the value for TotalBitrateAdaptations. This may not be the case for RealNetworks media players, however, because the TotalBitrateAdaptations value also reflects ASM rule changes.

### Report Format

```
%Client.*.Session.*.TotalUpshifts%
```

## URL

The URL property provides the base URL indicating the path to the clip or broadcast. It does not include any URL query parameters or Helix Server mount points, such as the /secure/ mount point for authenticated media access. For example, if the requested clip resides in the server's Content directory, the URL value is just the clip name, as in video.rm.

**Tip:** For the full URL requested by the media player, use PlayerRequestedURL (see page 204) instead.

### Report Format

```
%Client.*.Session.*.URL%
```

## PROXY REGISTRY PROPERTIES

The proxy registry properties record Helix Proxy statistics about bandwidth, memory use, current media player count, and so on.

### Proxy Registry Values

Using the basic proxy properties, you can create advanced logging reports that provide information about proxy load and performance, as well as the number of media players requesting streams. The property values are reset on a Helix Proxy restart.

#### BandwidthOutput

BandwidthOutput lists the bits per second, averaged over the last ten seconds, being delivered to the network for media players and Helix Administrator.

##### Registry Path

Server.BandwidthOutput

##### Report Format

%Server.BandwidthOutput%

#### BandwidthOutputPerPlayer

The BandwidthOutputPerPlayer property provides the average bandwidth in bits per second consumed by all media players currently receiving on-demand, live, or simulated live streams. The number is averaged across all media player sessions for the last ten seconds.

##### Registry Path

Server.BandwidthOutputPerPlayer

### Report Format

`%Server.BandwidthOutputPerPlayer%`

## BandwidthUsage

BandwidthUsage reports the total number of bytes delivered to the network since the last Helix Proxy restart.

### Registry Path

`Server.BandwidthUsage`

### Report Format

`%Server.BandwidthUsage%`

## BytesInUse

BytesInUse reports the number of bytes of memory currently used by Helix Proxy.

### Registry Path

`Server.BytesInUse`

### Report Format

`%Server.BytesInUse%`

## BytesMemoryUsage

BytesMemoryUsage indicates the total amount of memory in bytes allocated to Helix Proxy. Total memory allocation is set at Helix Proxy start-up.

**For More Information:** For information about changing the memory allocation, refer to the installation chapter of *Helix Proxy Administration Guide*.

### Registry Path

`Server.BytesMemoryUsage`

### Report Format

`%Server.BytesMemoryUsage%`

## ClientCount

ClientCount reports the number of currently active media players.

**Tip:** The properties HTTPClientCount, MMSClientCount, RTSPClientCount, and CloakedClientCount report media players connected through a specific streaming protocol.

### Registry Path

Server.ClientCount

### Report Format

%Server.ClientCount%

## CloakedClientCount

CloakedClientCount reports the number of active media players currently receiving media streams cloaked as HTTP. This may include RealPlayers and RTP-based players receiving cloaked RTSP streams, as well as Windows Media Players receiving cloaked MMS streams.

**Tip:** To view the total number of media players, use ClientCount (see page 213). See also the properties HTTPClientCount (see page 214), MMSClientCount (see page 215), and RTSPClientCount (see page 217).

### Registry Path

Server.CloakedClientCount

### Report Format

%Server.CloakedClientCount%

## FileObjectCount

The FileObjectCount property reports the number of media file objects currently open by Helix Proxy.

### Registry Path

Server.FileObjectCount

### Report Format

`%Server.FileObjectCount%`

## HTTPClientCount

HTTPClientCount lists the number of media players and Web browsers currently receiving data over HTTP. This does not include media players receiving data over RTSP or MMS protocol cloaked as HTTP.

**Tip:** To view the total number of media players, use ClientCount. Media players receiving media over RTSP or MMS cloaked as HTTP are recorded by the CloakedClientCount property. See also the properties MMSClientCount (see page 215) and RTSPClientCount (see page 217).

### Registry Path

`Server.HTTPClientCount`

### Report Format

`%Server.HTTPClientCount%`

## Last10Seconds

Several properties of the form `Last10SecondsProperty` provide a snapshot of the current proxy health. Each property reports an integer or an average value indicating Helix Proxy operations for the last ten seconds.

### Registry Path

`Server.Last10SecondsProperty`

### Report Format

`%Server.Last10SecondsProperty%`

### Properties

- `Last10SecondsBytesServedPerPlayer` – Average number of bytes delivered to each media player.
- `Last10SecondsENoBufs` – Number of network UDP errors.
- `Last10SecondsMainLoopIters` – Number of times the main Helix Proxy loop has run. High numbers indicate heavier loads.

- Last10SecondsMemoryOps – Number of memory allocations and deallocations.
- Last10SecondsMutexCollisions – Number of times one proxy thread was forced to wait for another thread to finish.
- Last10SecondsNewClients – Number of clients that have connected in the last ten seconds.
- Last10SecondsOtherUDPErrors – Number of miscellaneous UDP errors.
- Last10SecondsOverloads – Number of times the proxy has fallen behind in delivering packets to the network.
- Last10SecondsPackets – Number of packets written to the network in the last ten seconds.
- Last10SecondsSchedulerItems – Number of times a process has requested a service from the proxy scheduler.

## LoadState

The LoadState property lists one of four values that provides a general guide to the current Helix Proxy load.

### Registry Path

Server.LoadState

### Report Format

%Server.LoadState%

### Possible Values

- Normal
- High Load
- Extreme Load
- Unknown

## MMSClientCount

The MMSClientCount property indicates the number of media players currently receiving MMS streams.

**Tip:** To view the total number of media players, use `ClientCount` (see page 213). Media players receiving media cloaked as HTTP are recorded by the `CloakedClientCount` property (see page 213). See also the properties `HTTPClientCount` (see page 214) and `RTSPClientCount` (see page 217).

**Registry Path**

`Server.MMSClientCount`

**Report Format**

`%Server.MMSClientCount%`

## **MulticastTransportCount**

The `MulticastTransportCount` property indicates the number of media players currently connected in a back-channel or Windows Media multicast. This property does not provide information about scalable multicasts because client counts are not known during the multicast.

**Registry Path**

`Server.MulticastTransportCount`

**Report Format**

`%Server.MulticastTransportCount%`

## **PercentAggCPUUsage**

`PercentAggCPUUsage` records the average amount of CPU used by Helix Proxy since the last restart. CPU utilization is reported by the operating system. On multiprocessor machines, this figure represents the aggregate load for all processors.

**Registry Path**

`Server.PercentAggCPUUsage`

**Report Format**

`%Server.PercentAggCPUUsage%`

## PercentCPUUsage

PercentCPUUsage records the current amount of CPU used by Helix Proxy. CPU utilization is reported by the operating system. On multiprocessor machines, this figure represents the aggregate load for all processors.

### Registry Path

Server.PercentCPUUsage

### Report Format

%Server.PercentCPUUsage%

## RTSPAttemptedStreamed

RTSPAttemptedStreamed indicates the total number of media players that have attempted to connect to Helix Proxy using the RTSP protocol since the last Helix Proxy restart.

### Registry Path

Server.RTSPAttemptedStreamed

### Report Format

%Server.RTSPAttemptedStreamed%

## RTSPClientCount

RTSPClientCount indicates the number of media players currently communicating with Helix Proxy using RTSP.

**Tip:** To view the total number of media players, use ClientCount (see page 213). Media players receiving media cloaked as HTTP are recorded by the CloakedClientCount (see page 213) property. See also the properties HTTPClientCount (see page 214) and MMSPClientCount (see page 215).

### Registry Path

Server.RTSPClientCount

### Report Format

%Server.RTSPClientCount%

## TotalCAs

The TotalCAs property indicates the number of crash avoidance states that Helix Proxy has entered since startup.

### Registry Path

Server.TotalCAs

### Report Format

%Server.TotalCAs%

## TCPTransportCount

TCPTransportCount lists the total number of clients currently receiving data over TCP, regardless of the streaming protocol (RTSP, HTTP, or MMS).

### Registry Path

Server.TCPTransportCount

### Report Format

%Server.TCPTransportCount%

## UDPTransportCount

UDPTransportCount lists the total number of clients currently receiving data over UDP, regardless of the streaming protocol (RTSP, HTTP, or MMS).

### Registry Path

Server.UDPTransportCount

### Report Format

%Server.UDPTransportCount%

## Uptime

The Uptime property gives the amount of time in seconds since the last Helix Proxy restart.

### Registry Path

Server.Uptime

**Report Format**

`%Server.Uptime%`

**platform**

The platform property provides a text string that identifies the operating system, such as WinNT.

**Registry Path**

`Server.platform`

**Report Format**

`%Server.platform%`

**version**

The version property gives the Helix Proxy version number, such as 11.0.0.420.

**Registry Path**

`Server.version`

**Report Format**

`%Server.version%`



# INDEX

- A**
  - access control
    - Access variable, 133
    - AccessControl list, 132
    - Port\_# variable, 135
    - Ports variable, 134
    - Rule Number list, 132
    - To variable, 134
    - vFrom variable, 133
  - access log
    - Access Log list, 153
    - Basic Access Log list, 151
    - Enabled variable, 152
    - Filename variable, 154
    - LoggingStyle variable, 149, 152
    - LogRollFrequency variable, 154
    - LogRollSize variable, 153
    - Outputs list, 153
    - StatsMask variable, 150
    - Type variable, 152, 155
    - upgrade issues, 11
  - advanced logging
    - Description variable, 163
    - Enabled variable, 161
    - File list, 166
    - Format variable, 162
    - HTTPPost list, 166
    - Interval variable, 163
    - NTEventLog list, 167
    - Outputs list, 164
    - Pipe list, 167
    - StdErr list, 167
    - StdOut list, 168
    - SysLog list, 168
    - TCP list, 169
    - TCPListen list, 169
    - Type variable, 161
    - UDP list, 169
    - UDPMCast list, 170
    - upgrade issues, 11
    - WatchRoot variable, 164
  - authentication
    - client access
      - AllowDuplicateIDs variable, 146
      - Authority list, 145
      - DatabaseID variable, 145
      - Enabled variable, 144
      - NoAuthenticateHost variable, 147
      - ProxyAuthentication list, 144
      - Realm variable, 145
      - Rule Name list, 146
      - RuleList list, 146
    - realms
      - Authentication Realms list, 140
      - DatabaseID variable, 142
      - Group variable, 143
      - PluginID variable, 141
      - Protocol Name list, 141
      - Provider variable, 142
      - Realm Name list, 140
      - Realm variable, 140
  - see also* databases
- B**
  - bandwidth
    - automatic detection
      - AutoBWDetection list, 36
      - SendBWpackets variable, 37
    - legacy negotiation, 10
    - MaxGatewayBandwidth variable, 31
    - MaxProxyBandwidth variable, 31
- C**
  - caching
    - BasePath variable, 49
    - CacheMountPoint variable, 51
    - Enabled variable, 50

- MaxCacheSizeMB variable, 50
- MountPoint variable, 49
- RNCache list, 50
- RNCache Local File System list, 49
- ShortName variable, 49
- capabilities exchange
  - BasePath variable, 69
  - CapabilityExchange list, 69
  - CapExProfiles list, 68
  - DefaultProfiles list, 70
  - IgnoreCapExHeaders variable, 72
  - MaxProfileSize variable, 70
  - MountPoint variable, 68
  - ProfileCacheSize variable, 69
  - ShortName variable, 68
  - URI variable, 71
  - UserAgent variable, 71
- client properties
  - CBID, 186
  - ClientID, 187
  - ConnID, 188
  - ControlBytesSent, 188
  - GUID, 188
  - IPAddress, 189
  - IsCloaked, 189
  - IsRDT, 190
  - Language, 190
  - PlayerStartTime, 192
  - Port, 193
  - Protocol, 193
  - RequestMethod, 194
  - StartTime, 194
  - User-Agent, 194
  - Version, 195
- client session properties
  - ASMSubscribes, 197
  - ASMUnsubscribes, 198
  - AvgBitrate, 198
  - BytesSent, 198
  - Duration, 199
  - DurationSeconds, 199
  - EstimatedPlayerBufferOVERRUNS, 199
  - EstimatedPlayerBufferUnderrun, 200
  - FailedResends, 200
  - FileSize, 201
  - InterfaceAddress, 201

- IsMulticastUsed, 201
- IsRAStreamFound, 202
- IsREStreamFound, 202
- IsRIStreamFound, 202
- IsRVStreamFound, 203
- IsUDP, 203
- LogStats, 204
- PacketsLost, 204
- PacketsSent, 204
- PlayerRequestedURL, 204
- PlayTime, 205
- ProxyConnectionType, 205
- RRFrequency, 206
- RTT, 206
- SendingTime, 206
- SessionID, 207
- SessionStartTime, 207
- Status, 207
- SuccessfulResends, 209
- TotalBitrateAdaptations, 209
- TotalDownshifts, 210
- TotalUpshifts, 210
- URL, 210
- configuration file
  - backup, 14
  - case-sensitivity, 15
  - comment tag, 15
  - editing, 13
  - list tag, 15
  - multiple files, 14
  - security, 13
  - syntax, 13
  - variable tag, 16

**D** databases

- Name variable, 137
- Password variable, 138
- Path variable, 137
- TableNamePrefix variable, 138
- User variable, 138

datatype packetization

- Datatypes list, 60
- DefaultMaxPacketSize variable, 61
- ForcePacketization variable, 62
- MIME types list, 61

- delayed shutdown
    - ClientDisconnectInterval variable, 34
    - LogPlayerTermination variable, 35
    - NewClientConnectionsAllowed variable, 35
    - ServerShutdown list, 34
    - ShutdownProceedTime variable, 34
  - differentiated services
    - Control variable, 64
    - DiffServ list, 64
    - Media variable, 64
- E** error log
  - Basic Error Log list, 155
  - Enabled variable, 156
  - Error Log list, 157
  - Filename variable, 158
  - LogRollFrequency variable, 158
  - LogRollSize variable, 157
  - Outputs list, 156
  - Type variable, 156, 158
- Extensible Markup Language (XML) *see* XML
- F** FSMount list, 17
- G** global properties
  - date, 180
  - double quote, 182
  - Greenwich MeanTime, 180
  - hour, 181
  - local time difference, 181
  - minute, 181
  - new line, 182
  - second, 182
  - tab, 183
  - time of day, 180
- H** Helix Administrator
  - BaseMountPoint variable
    - for files, 39
  - BasePath variable
    - for docs, 39
    - for HTML, 39
    - for images, 40
    - for javamonitor, 40
  - manual changes on UNIX, 17
  - MonitorPassword variable, 25
  - MountPoint variable
    - for docs, 39
    - for files, 39
    - for HTML, 39
    - for images, 39
    - for javamonitor, 40
  - Realm variable, 39
  - RealSystem Administrator DOCS list, 39
  - RealSystem Administrator Files list, 38
  - RealSystem Administrator HTML list, 39
  - RealSystem Administrator IMAGES list, 39
  - RealSystem Administrator JAVAMONITOR list, 40
  - ShortName variable
    - for docs, 39
    - for files, 38
    - for HTML, 39
    - for images, 39
    - for javamonitor, 40
- HTTP
  - HTTP Deliverable list, 43
  - Path\_*n* list, 44
- I** IP addresses
  - Address\_XX variable, 27
  - IPBindings list, 27
  - UseIPv4Only variable, 29
- L** logging
  - see* access log
  - see* advanced logging
  - see* error log
- M** MMS port, 24
  - multicasting
    - AddressRange variable, 114
    - Allow variable, 115
    - AnnounceSAP variable, 112
    - ControllList list, 115
    - DeliveryOnly variable, 114
    - Enabled variable, 112
    - Multicast list, 112

- Resend variable, 113
- RTSPPort variable, 113
- Rule Number list, 115
- TTL variable, 113

- O** on-demand streaming
  - BasePath variable, 43
  - FSMount list, 41
  - MountPoint variable, 42
  - Plugin Directory variable, 41
  - ShortName variable, 42

- P** paths
  - ErrorLogPath variable, 23
  - LicenseDirectory variable, 22
  - LogPath variable, 23
  - PidPath variable, 22
  - PluginDirectory variable, 21
  - SupportPluginDirectory variable, 22

PNA, 10

ports

- AdminPort variable, 25
- HTTPPort variable, 24
- MMSPort variable, 24
- MonitorPort variable, 25
- RTSPPort variable, 23

protocol options

- PreferClientTCP variable, 47
- Protocols list, 46
- RTSP list, 47

proxy

- KeepAliveInterval variable, 31
- MaxProxyConnections variable, 30

pull splitting

- BitsaveEnable variable, 128
- BitsaveMountPoint variable, 129
- BroadcastDistributionMountPoint variable, 128
- BroadcastReceiver list, 122
- FECLevel variable, 126
- file system setup, 118
- PathPrefix variable, 125
- Protocol variable, 123
- Proxy Receiver Name list, 123
- PullSplitEnabled variable, 124

- Receivers list, 123
- ResendSupported variable, 126
- Splitter\_DoubleURL list, 119
- SplitterProtocol variable, 121
- SureStreamAware variable, 124
- UseRTSPInitiate variable, 127
- UseTCPForPullBackchannel variable, 125

**R** rate control

BCC

- BCC list, 91
- DecreaseCoefficient variable, 92
- DecreaseExponent variable, 93
- DecreaseThreshold variable, 94
- IncreaseCoefficient variable, 92
- IncreaseExponent variable, 93
- IncreaseThreshold variable, 93
- LowerLimit variable, 94
- RTTUseIIR, 91
- specification, 91
- TargetRatio variable, 94
- TimeoutTransmission variable, 95
- trace log format, 96
- Trace variable, 95

ConfigHelixRAEnabled variable, 75

congestion control

- ChannelRate variable, 81
- CongestionControl list, 79
- InitialRate variable, 81
- MaxBurst variable, 80
- MaxRateScalar variable, 82
- PassThrough variable, 80
- Type variable, 79
- UseClientRateReq variable, 82

InactivityTimeout variable, 77

input source

- AllowInitialExternalSubscription variable, 108
- InitialMediaRate variable, 107
- InputSource list, 107
- MinPreroll variable, 107
- NetworkAffinity variable, 108
- StaticPushSource list, 108

Media Player Name list, 75

MediaDelivery list, 74

rate manager

- BufferModel variable, 98
  - ClientBufferHighWatermark variable, 100
  - ClientBufferLowWatermark variable, 99
  - ClientBufferPeriod variable, 103
  - MultiStreamVerifier variable, 99
  - RateManager list, 98
  - ResendTimeLimit variable, 102
  - Session list, 98
  - TargetTimeHighWatermark variable, 101
  - TargetTimeLowWatermark variable, 101
  - trace log format, 103
  - Trace variable, 103
  - Type variable, 99
  - tcp
    - LimitRate variable, 97
    - TCP list, 97
  - TFRC
    - RTTUseIIR variable, 86
    - SlowStartScalar variable, 88
    - specification, 85
    - StartRate variable, 88
    - TFRC list, 85
    - TimeoutScalar variable, 87
    - TimeoutTransmission variable, 86
    - trace log format, 89
    - Trace variable, 89
    - UseSlowStart variable, 87
  - timeout
    - Disable variable, 84
    - InitialTimeou variable, 83
    - Interval variable, 84
    - MaxTimeouts variable, 84
    - Timeout list, 83
  - transport
    - RtcpRRratio variable, 78
    - RtcpRSratio variable, 78
    - Transport list, 77
  - UseMediaDeliveryPipeline variable, 76
  - UserAgent variable, 76
  - UserAgentProfiles list, 75
  - RealPlayer
    - see also* clients
  - redirection of clients
    - Port variable, 52
    - RedirectToAddress variable, 52
    - RedirectToPort variable, 53
    - RTSPRedirection list, 52
  - redundant proxies
    - Alternate Name list, 58
    - Alternates list, 58
    - Host variable, 59
    - Port variable, 59
    - ProxyAlternates list, 58
  - registry properties, 179
    - URL for viewing, 179
  - routing
    - Description variable, 55
    - ParentMEIPort variable, 56
    - ParentName variable, 57
    - ParentRTSPPort variable, 56
    - ProxyRoutingTable list, 54
    - Rule Number list, 54
    - Rule variable, 55
    - UseParentProxy variable, 55
- S**
- server properties
    - BandwidthOutput, 211
    - BandwidthOutputPerPlayer, 211
    - BandwidthUsage, 212
    - BytesInUse, 212
    - BytesMemoryUsage, 212
    - ClientCount, 213
    - CloakedClientCount, 213
    - FileObjectCount, 213
    - HTTPClientCount, 214
    - Last10SecondsBytesServedPerPlayer, 214
    - Last10SecondsENoBufs, 214
    - Last10SecondsMainLoopIters, 214
    - Last10SecondsMemoryOps, 215
    - Last10SecondsMutexCollisions, 215
    - Last10SecondsNewClients, 215
    - Last10SecondsOverloads, 215
    - Last10SecondsPackets, 215
    - Last10SecondsSchedulerItems, 215
    - Last10SecondsUDPErrors, 215
    - LoadState, 215

- MMSClientCount, 215
- MulticastTransportCount, 216
- PercentAggCPUUsage, 216
- PercentCPUUsage, 217
- platform, 219
- RTSPAttemptedStreamed, 217
- RTSPClientCount, 217
- TCPTransportCount, 218
- TotalCAs, 218
- UDPTransportCount, 218
- Uptime, 218
- version, 219
- shutdown, *see* delayed shutdown
- SIGHUP command, 17
- SNMP
  - Activate variable, 171
  - GlobalCPUUtilization variable, 173
  - Master Address variable, 171
  - MaxConnections variable, 175
  - MemoryUsageWaterMark variables, 174
  - OutBandwidthWaterMark variables, 174
  - SendTrap variable, 172
  - ServerStart variable, 173
  - SNMP list, 171
  - SNMPTrapInterval variable, 172
  - TrapThresholds list, 173
- StreamerCount variable, 44
  
- T** TCP
  - buffer size, 45
  - preference for media players, 47
  - transport options
    - TCPSendBufferSize variable, 45
    - UDPSendBufferSize variable, 46
  - troubleshooting guide, 4
  
- U** UDP buffer size, 46
- UNIX
  - Group variable, 32
  - SIGHUP, 17
  - User variable, 33
  
- X** XML
  - comment tag, 15
  - configuration file, 14
  - declaration tag, 15