## CONTENTS

### INTRODUCTION

- Audience for this Guide ................................................................. 1
- How this Guide Is Organized .......................................................... 1
- Conventions Used in this Manual .................................................... 2
  - Terminology .................................................................................. 2
  - Typographical Conventions ......................................................... 2

### CHAPTER 1: FIREWALLS

- Media Server Placement ................................................................. 3
- Host Addresses for NAT Firewalls .................................................... 3
- Virtual IP Addresses ........................................................................ 4
  - How Virtual IP Addressing Works ................................................ 4
  - The Problem with Virtual Addressing .......................................... 4
  - Resolving the Problem .................................................................. 4
- RTSP Communication ...................................................................... 5
  - RTSP Control Channel ................................................................. 5
  - UDP Data Channel ........................................................................ 5
- HTTP Cloaking for RTSP and RTMP ................................................ 6
  - GET and POST Methods ............................................................ 6
  - Port 80 For HTTP Traffic ............................................................ 6
  - Port Hinting on Helix Server ......................................................... 6
- Encoders, Receivers, and Proxies ..................................................... 7
  - Communicating With Live Media Encoders ................................... 7
  - HTTP Broadcasts from Windows Media Encoder .......................... 7
  - Communicating With Receivers ................................................... 7
  - Communicating With Helix Proxies .............................................. 8

### CHAPTER 2: SNMP

- Understanding SNMP ................................................................. 9
- SNMP Plug-in .................................................................................. 9
- Master Agent ................................................................................. 9
- SNMP Protocol ............................................................................... 9
  - Management System and Management Information Base (MIB) ......................................................... 10
- Configuring the SNMP Plug-In ....................................................... 11
- Configuring the Master Agent ....................................................... 11
  - Modifying the Master Agent Configuration File ......................................................... 11
  - Defining Master Agent Addresses and Ports ........................................... 12
  - Setting Up SNMP Security .................................................................. 12
  - Defining a View Access Control Model ........................................... 13
- Running the Master Agent on Windows ........................................ 15
- Starting the Master Agent on UNIX .............................................. 16
- Running a Management System .................................................. 16
  - Monitor Tree .................................................................................. 16
  - Configuration Tree on Helix Server .............................................. 17
  - Configuration Tree on Helix Proxy .............................................. 18
CHAPTER 4: PLAYLIST MANAGEMENT ........................................................................................................ 30
Understanding Playlist Management ........................................................................................................ 30
Types of Playlist Sessions ..................................................................................................................... 30
Web Portal Requirements ..................................................................................................................... 32
Stream Requirements for RTSP Players ............................................................................................... 32
Segment Handling with HLS Clients ..................................................................................................... 34
Compatibility with Other Features ....................................................................................................... 35
Internally Controlled and Noncontrolled Playlist Sessions ................................................................ 36
  Initial Playlist Selection .................................................................................................................. 36
  Playlist Seeking ........................................................................................................................... 37
Externally Controlled Playlist Sessions .............................................................................................. 38
  Initial Playlist Selection .................................................................................................................. 38
  Chapter Skipping ........................................................................................................................... 40
  Playlist Seeking ........................................................................................................................... 42
  Playlist Control Parameters ......................................................................................................... 42
Playlist Format ................................................................................................................................ 45
SMIL Timing Values ........................................................................................................................... 45
Setting Metadata Values ..................................................................................................................... 46
Adding Clips to the Playlist .................................................................................................................. 48
Setting Up Chapters ............................................................................................................................ 51
Defining a Chapter or Clip ID ............................................................................................................. 51
Streaming a Secondary Playlist .......................................................................................................... 51
Playlist Examples .............................................................................................................................. 52
Uploading Playlists ............................................................................................................................ 54
  Adding a Playlist .......................................................................................................................... 54
  Updating a Playlist ....................................................................................................................... 54
  Deleting a Playlist ......................................................................................................................... 55
Logging and Error Codes ................................................................................................................... 55
<table>
<thead>
<tr>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>Logging Styles and Variables .......................................................... 55</td>
</tr>
<tr>
<td>Error Conditions .................................................................................. 55</td>
</tr>
<tr>
<td>HTTP Status Codes ............................................................................. 56</td>
</tr>
<tr>
<td>Configuring Helix Server for Playlist Control ...................................... 57</td>
</tr>
<tr>
<td>Set Playlist HTTP Ports ...................................................................... 57</td>
</tr>
<tr>
<td>Define Playlist Mount Points ............................................................. 57</td>
</tr>
<tr>
<td>Secure the Playlist Management System ............................................. 58</td>
</tr>
<tr>
<td>Enable HLS Segmentation .................................................................... 59</td>
</tr>
<tr>
<td>Enable Playlist Control Requests ...................................................... 59</td>
</tr>
<tr>
<td>CHAPTER 5: CHANNEL SWITCHING ...................................................... 60</td>
</tr>
<tr>
<td>Understanding Channel Switching ....................................................... 60</td>
</tr>
<tr>
<td>Third-Party Components ..................................................................... 61</td>
</tr>
<tr>
<td>Codecs and File Formats .................................................................. 62</td>
</tr>
<tr>
<td>Protocols and Transports .................................................................. 63</td>
</tr>
<tr>
<td>User Authentication ........................................................................... 63</td>
</tr>
<tr>
<td>Error Conditions ................................................................................ 63</td>
</tr>
<tr>
<td>Compatibility with Other Features .................................................... 64</td>
</tr>
<tr>
<td>Channel Controller ........................................................................... 64</td>
</tr>
<tr>
<td>Channel Switching Control Port ......................................................... 64</td>
</tr>
<tr>
<td>HTTP Request Parameters .................................................................. 65</td>
</tr>
<tr>
<td>HTTP Status Codes ........................................................................... 67</td>
</tr>
<tr>
<td>Channel Switching Workflows ............................................................ 67</td>
</tr>
<tr>
<td>Initial Channel Playback .................................................................... 67</td>
</tr>
<tr>
<td>Switch to a New Channel .................................................................. 68</td>
</tr>
<tr>
<td>End Session ....................................................................................... 69</td>
</tr>
<tr>
<td>Configuring Channel Switching .......................................................... 70</td>
</tr>
</tbody>
</table>

INDEX 71
Welcome to Helix™ Universal Server Version 15.0. This guide explains how to integrate Helix Server with third-party tools to set up features such as server-side playlist management. It also contains information about firewalls and integration with SNMP monitoring systems.

For More Information: For information about installing Helix Server, refer to Helix Media Delivery Platform Quick Start Guide. That guide also explains how to use the Helix Administrator interface and access the online help, which provides setup information for many server features.

Audience for this Guide

This guide is intended for system administrators who will manage media stream but not necessarily create the content. Information services professionals, and Web server administrators may also find this book useful.

How this Guide Is Organized

This administration guide contains the following chapters and appendixes.

Chapter 1: Firewalls
If you’re streaming media to users on the Internet, you’ll need to know how Helix Server, Helix Proxy, and other RealNetworks products interact with firewalls.

Chapter 2: SNMP
This chapter explains how to configure the Simple Network Monitoring Protocol (SNMP) plug-in and master agent to monitor Helix Server activity using third-party monitoring software.

Chapter 3: Splitting
Splitting is a method of delivering a stream from a RealNetworks encoder to Helix Server or from one Helix Server to another.

Chapter 4: Playlist Management
The server-side playlist feature streams a sequence of clips or broadcasts in a single RTSP session.

Chapter 5: Channel Switching
This chapter covers fast channel switching, which allows a user to change streams within a single RTSP session.
Conventions Used in this Manual

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

Terminology

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

Typographical Conventions

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

<table>
<thead>
<tr>
<th>Convention</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>syntax</strong></td>
<td>This font is used for syntax of configuration files, URLs, or command-line instructions.</td>
</tr>
<tr>
<td><strong>variables</strong></td>
<td>Italic text represents variables. Substitute values appropriate for your system.</td>
</tr>
<tr>
<td><strong>emphasis</strong></td>
<td>Bold text is used for emphasis.</td>
</tr>
<tr>
<td>...</td>
<td>Ellipses indicate nonessential information omitted from examples.</td>
</tr>
<tr>
<td>[ ]</td>
<td>Square brackets indicate optional material. If you choose to use the material within the brackets, don’t type the brackets themselves. An exception to this is in the basic access log, where statistics generated by the <em>StatsMask</em> variable are enclosed in regular brackets.</td>
</tr>
</tbody>
</table>
Firewalls may present communications problems for Helix Server and Helix Proxy. This chapter provides background on firewalls and port configurations. It recommends ways to work with firewalls to give viewers the best possible streaming media experience.

**Tip:** After installing Helix Server or Helix Proxy, you can find additional information about ports by navigating to **Server Setup > Ports** or **Proxy Setup > Ports** in Helix Administrator and clicking the **Help** link.

### Media Server Placement

If you are streaming content only to media clients inside your organization, place media servers and proxies inside your firewall. This requires no special configuration other than to provide virtual private network (VPN) access to any clients outside the firewall.

To stream content to clients on the Internet, it’s better not to locate Helix Server behind a firewall. For optimal streaming, Helix Server needs to use streaming protocols and to process incoming and outgoing UDP connections on a variety of ports. Although you may be able to change your organization’s security policy to enable optimal communication, this may hamper the effectiveness of the firewall.

The best solution may be to place Helix Server or Helix Proxy in a perimeter network, sometimes known as a De-Militarized Zone (DMZ). In this scenario, you fortify the connection between main and perimeter networks but allow a less stringent security policy in the perimeter. This keeps the main network secure while maintaining optimal connections between the Internet and media servers.

### Host Addresses for NAT Firewalls

If Helix Server resides on a local network and its IP address is not directly accessible from the Internet, media links that include client launch utilities may fail. This occurs because Helix Server instructs media players to contact it on its IP address, which is accessible only from within the local network. This situation can arise for numerous reasons:

- Helix Server sits behind a Network Address Translation (NAT) firewall.
- The network address exposed to Internet clients is a virtual IP (VIP).
- Media requests are routed to different servers using a load-balancing mechanism.

In these cases, you can add variables to the Helix Server configuration file to provide media clients with the address of the firewall or VIP, which can be configured to route the request to the appropriate Helix Server.
For More Information: The Helix Administrator online help topic Protocols > NAT Firewalls and Virtual IP Addresses explains how to add HostName and RTMPtrIdent2 variables to the configuration file.

Virtual IP Addresses

If you need to place a cluster of Helix Servers behind a virtual IP address, the network topology may have unintended consequences for RTSP traffic behind a restrictive firewall.

How Virtual IP Addressing Works

A typical IP address resolves to a single server. A virtual IP address resolves to a cluster of servers, typically through a hardware switch. Consider the case of a cluster of servers behind a hardware switch, where all servers share the same content but are configured with private IP addresses. In this scenario, only the hardware switch is assigned a public IP address. The switch receives all public communication, passing each request to a host in the cluster.

The Problem with Virtual Addressing

A problem arises from the combination of public and private IP addresses. If a firewall blocks streaming media protocols, the client communicates through HTTP cloaking. In most cases, this effectively bypasses firewall security, which typically allows HTTP traffic to pass. For cloaking to work, the client must be able to make two HTTP connections to the same Helix Server.

When a client uses HTTP cloaking, Helix Server replies to the initial HTTP connection with its actual IP address, not the virtual IP of the cluster. This allows the client to circumvent the hardware switch and establish its second HTTP connection directly with the Helix Server handling the request. But if that server uses a private IP address, the client cannot make the second, necessary connection, and HTTP cloaking fails.


Resolving the Problem

There are two ways to resolve the virtual addressing problem:

- Configure firewalls to allow connections by streaming media protocols. This is the ideal solution because communication between the server and clients will be the most efficient.
- Use globally routable IP addresses on all hosts behind a virtual IP address. This way clients can make HTTP-cloaked requests behind firewalls that restrict streaming media protocols. This requires your organization to register for a larger number of public IP addresses, however.

Note: If neither of these solutions is possible, some RTSP clients residing outside of highly restrictive firewalls may not be able to access content.
RTSP Communication

Helix Server and Helix Proxy use two connections to communicate with RTSP-based media clients, a control channel and a data channel. For data transport, RTSP clients typically prefer UDP, which may be blocked by restrictive firewalls.

Communication with an RTSP Media Player

RTSP Control Channel

Using the control channel, the server can request and receive passwords. Media Players use the control channel to send instructions such as pause or stop. Most players can work around control channel failures that arise when a firewall blocks the preferred protocol. The player typically shifts to TCP, which is less likely to be blocked than UDP.

**Note:** When a firewall exists between a media client and the media server, the IP address that appears in the access log’s IP address field may be the firewall address rather than the true client address.

**For More Information:** The Helix Administrator online help topic Helix Administrator > Server Setup > Ports > Server Default Ports lists the default ports used by media players to contact the server and set up a control channel.

UDP Data Channel

Once the RTSP control connection is established, the media player negotiates the data channel. Optimally, the data channel will use the UDP transport. If this fails, the player uses the established control channel for data. By default, servers use the following ports for UDP data transfer:

<table>
<thead>
<tr>
<th>Port Range</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Helix Server:</td>
<td>ports in the range 6970 to 32000</td>
</tr>
<tr>
<td>Helix Proxy:</td>
<td>ports in the range 6970 to 65535</td>
</tr>
</tbody>
</table>

After you install a server, you can modify the UDP port range by adding MinUDPPort and MaxUDPPort port values to the configuration file. For example:

```xml
<Var MinUDPPort="7240"/>
<Var MaxUDPPort="24100"/>
```

**For More Information:** Refer to the Helix Administrator online help topic Configuration File > Basic Setup Configuration > Protocols, Ports, and Password Configuration.

**Tip:** Helix Server and Helix Proxy can use a limited range of UDP ports to receive media player requests for lost packet resends. Specify this range in the Helix Administrator Ports page (Server Setup > Ports or Proxy Setup > Ports).
HTTP Cloaking for RTSP and RTMP

Some firewalls restrict streaming media protocols like RTSP or RTMP, preventing a media player from establishing the control connection. In these cases, Helix Server and the player circumvent the problem by disguising streaming media traffic as HTTP, a solution known as HTTP cloaking.

**Note:** HTTP is not designed for streaming long media files. As a result, HTTP cloaking of RTSP or RTMP streams may lower playback quality.

**GET and POST Methods**

The HTTP cloaking method must work around limitations in the HTTP protocol. For example, media players use two HTTP streams to connect to Helix Server. Because the player initiates both streams, the client firewall typically allows these connections as outgoing HTTP traffic.

The first HTTP connection uses the **GET** method, the standard means for a browser to request a Web page. At the receiving end, Helix Server strips off the HTTP disguise, using the encapsulated RTSP or RTMP information to determine what information to send the player.

Helix Server must then wait for the second HTTP connection from the same player to proceed with streaming the media. This second connection uses the HTTP **POST** method, the standard means for a Web server to send data to a browser.

Once both of these player-initiated streams are established, the media player and Helix Server can pass packets in two directions through a firewall that blocks RTSP or RTMP but allows streaming data that is cloaked as HTTP.

**Port 80 For HTTP Traffic**

For HTTP cloaking to work, the media player must connect to the Helix Server HTTP port. RealNetworks recommends setting the HTTP port on Helix Server to the standard port 80 during installation to provide the widest support of all media players.

**Warning!** When you install Helix Server and a Web server on the same machine, you need to take precautions before assigning port 80 to Helix Server. For more information, see *Helix Media Delivery Platform Quick Start Guide*.

**Port Hinting on Helix Server**

Port hinting offers a solution for a Helix Server that cannot use default port values. It allows Helix Server to send the proper port numbers to certain types of media players when a client launch utility is used. This feature is enabled by default.

**For More Information:** See the Helix Administrator online help topic *Helix Administrator > Server Setup > Ports > Communication Through Nonstandard Ports*. 
Encoders, Receivers, and Proxies

Once you have placed Helix Server in relation to your firewall, you need to consider the placement of other encoders, servers, and proxies. Generally, the same rules and limitations discussed in the preceding sections apply to placing these components as well.

Communicating With Live Media Encoders

If possible, place live media encoders in the perimeter network along with Helix Server. This may not always be possible, however. For example, encoders recording a remote, live event may need to reside on the public side of the firewall.

UDP and TCP Streams

Flash Media encoders typically use TCP for data transport. Helix Producer typically uses UDP for data transport although TCP is available as an option for traversing restrictive firewalls or transmitting data on a lossy network.

Default Server Ports for Live Data

The following are the default Helix Server ports used to receive data from RealNetworks encoders:

<table>
<thead>
<tr>
<th>Configuration</th>
<th>Default Ports</th>
</tr>
</thead>
<tbody>
<tr>
<td>Helix Producer in Helix Push mode</td>
<td>50001 to 50050</td>
</tr>
<tr>
<td>Helix Producer in Advanced Push or Multicast mode</td>
<td>30001 to 30020</td>
</tr>
<tr>
<td>Flash Media encoders</td>
<td>1935</td>
</tr>
<tr>
<td>Windows Media encoders in push mode</td>
<td>7077</td>
</tr>
</tbody>
</table>

For More Information: The Helix Administrator online help explains the Helix broadcast methods. Third-party encoders may transmit on a different set of ports. Refer to the specific encoder documentation for details.

HTTP Broadcasts from Windows Media Encoder

When Windows Media Encoder delivers a live Windows Media stream, the only protocol option is HTTP over TCP. With pull broadcasting, Helix Server pulls the live feed from a specific encoder port and receives the stream on its default HTTP port.

Note: With push broadcasting, Windows Media Encoder delivers the HTTP stream to a predefined port on Helix Server. Refer to the Helix Administrator online help topic Helix Administrator > Broadcasting > Windows Media Encoding > Setting up a Windows Media Push Broadcast.

Communicating With Receivers

By default, Helix Servers functioning as transmitters and receivers communicate using UDP. An option is available for them to use TCP instead. The data ports used by and transmitters and receivers are fully configurable. The default range is 30001 to 30020.

For More Information: Chapter 3 explains splitting.
Communicating With Helix Proxies

Helix Proxies commonly work behind a firewall. In this respect, a proxy-to-server connection behaves like a client-to-server connection. Helix Proxy first tries to connect to Helix Server with RTSP using UDP for data transport. If the firewall prohibits UDP connections, Helix Proxy tries TCP.

**Note:** Helix Proxy has no option for HTTP delivery. If a firewall prohibits RTSP between the origin Helix Server and Helix Proxy, the proxy will not be able to cache or split streams for media clients.
Using Simple Network Monitoring Protocol (SNMP), you can monitor Helix Server from an SNMP management system. This allows you to change Helix Server configuration from a third-party tool, and send notice of important events to an external program. This chapter explains how to set up the SNMP monitoring plug-in and the Helix Server master agent.

**Note:** Helix Proxy also supports SNMP as described in this chapter.

### Understanding SNMP

The following sections describe the components of the Helix Server SNMP monitoring system. Before implementing SNMP on Helix Server, be sure that you understand the basics of SNMP monitoring and know how to operate your chosen SNMP management system.

#### SNMP Plug-in

Helix Server includes an SNMP plug-in that monitors its registry for configuration values and events. The plug-in communicates to the master agent using a proprietary protocol. It can send important information about Helix Server operation to the master agent, and update the Helix Server configuration as instructed by the management system. You must configure the plug-in before it can operate.

**For More Information:** The section “Configuring the SNMP Plug-In” on page 11 explains how to set up the plug-in.

#### Master Agent

The SNMP plug-in communicates with the master agent, an executable program included with Helix Server. The master agent then communicates with the management system using the SNMP protocol. The SNMP plug-in and the management system never communicate directly. The master agent can run as an independent application or a Windows service. Once configured, the master agent generally runs without the need for user intervention.

**For More Information:** See “Configuring the Master Agent” on page 11.

#### SNMP Protocol

The master agent uses the SNMP protocol to communicate with the management system. It supports SNMP version 1 (SNMPv1), version 2c (SNMPv2c), and version 3 (SNMPv3). Versions 1 and 2 of the SNMP protocol do not encrypt messages between the two components, and are therefore recommended only when both Helix Server and the management system reside behind a firewall on the same private network.
Note: Helix Server does not support SNMP over Internet Protocol version 6 (IPv6). Components must use IPv4 addresses.

SNMP Version 3 Protocols

SNMPv3 is suitable for communications over an unprotected network. The User-based Security Model (USM) for SNMPv3 defines two authentication protocols, both of which are supported for Helix Server SNMP:

- HMAC-MD5-96
  This protocol is based on MD5. Operations using MD5 occur faster than those using SHA.
- HMAC-SHA-96
  This protocol is based on SHA-1. SHA provides a stronger security mechanism than MD5.

SNMP Version 3 Security Levels

SNMPv3 defines three levels of security. The lowest level (noAuthNoPriv) does not provide authentication or privacy, and is comparable to SNMP version 1. The second level (AuthNoPriv) provides authentication but no privacy. The third level (AuthPriv) provides authentication and encryption for all messages.

Management System and Management Information Base (MIB)

You can use any third-party SNMP monitoring tool as your management system. The management information base (MIB) determines the Helix Server configuration variables that the management system monitors and controls. It also defines the event traps that the SNMP plug-in can report to the master agent. Helix Server ships with a MIB configuration file named helixserver.my, located in the main Helix Server installation directory.

For More Information: The section “Running a Management System” on page 16 explains the monitoring trees that appear in the management system.
Configuring the SNMP Plug-In

You configure the SNMP plug-in through Helix Administrator. The configuration connects the plug-in to the master agent, and defines which events to report to the management system. To configure the SNMP plug-in, navigate to **Logging & Monitoring > SNMP** in Helix Administrator. Here you enable SNMP, define the master agent address, and set trap values. For details, refer to the Helix Administrator online help topic **Helix Administrator > Logging and Monitoring > SNMP**.

Configuring the Master Agent

The master agent is the intermediary through which the SNMP plug-in and the management system communicate. It must always run on the Helix Server machine. The following sections explain how to modify the master agent configuration file to define your system addresses, users, and security model.

Modifying the Master Agent Configuration File

You use the master.cfg file installed in the Helix Server installation directory to configure the master agent. This allows the master agent to communicate with the SNMP plug-in and the management system. It also defines the security level for each person who uses the management system. Edit this XML-formatted text file using any text, HTML, or XML editor. The following example shows the default configuration file:

```xml
<?xml version="1.0" encoding="US-ASCII"?>
<preferences version="0.5">
    <config ManagerAddress="127.0.0.1" ManagerSNMPPort="162"
        LocalSNMPPort="161" AgentXProtocolPort="705" EngineID="XXX"/>
    <security CommunityString="public"/>
    <SecurityModel ModelType="USM">
        <users UserName="xxx">
            <Authentication Type="MD5" Password="yyy"/>
            <Privacy Type="DES" Password="zzz"/>
        </users>
        <users UserName="unsecureUser">
            <Authentication Type="NONE" Password=""/>
            <Privacy Type="NONE" Password=""/>
        </users>
    </SecurityModel>
    <SecurityToGroup SecurityModel="USM" User="unsecureUser" Group="v3Group"/>
    <SecurityToGroup SecurityModel="USM" User="test" Group="testGroup"/>
    <SecurityToGroup SecurityModel="v2" User="vishal" Group="v1v2group"/>
    <SecurityToGroup SecurityModel="v1" User="public" Group="v1v2group"/>
</preferences>
</preferences>
```

<SecurityModel ModelType="VACM">
    <groups Name="v3Group" SecurityModel="USM" SecurityLevel="1"
        Context="" Notify_View="testView" Read_View="testView"
        Write_View="testView"/>
    <groups Name="testGroup" SecurityModel="USM" SecurityLevel="1"
        Context="" Notify_View="testView" Read_View="testView"
        Write_View="testView"/>
</SecurityModel>
```
Defining Master Agent Addresses and Ports

The following lines in the master agent configuration define the basic communication between the master agent, the SNMP plug-in, and the management system:

```xml
<config ManagerAddress="127.0.0.1" ManagerSNMPPort="162"
        LocalSNMPPort="161" AgentXProtocolPort="705" EngineID="XXX"/>
```

The following table explains the values you should set for these attributes.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>ManagerAddress</td>
<td>The IPv4 address of the management system. The master agent uses this address to send traps to the management system. You must specify an IP address, not a DNS name. The master agent does not support IPv6 addresses.</td>
</tr>
<tr>
<td>ManagerSNMPPort</td>
<td>The port used by the management system to listen for communication from the master agent. The default is port 162.</td>
</tr>
<tr>
<td>LocalSNMPPort</td>
<td>The local port used by the master agent for SNMP communications. The default is port 161. If another application uses an SNMP process, port 161 may be in use. In this case, specify a free port. Do not use port 162, as this can cause a system slowdown.</td>
</tr>
<tr>
<td>AgentXProtocolPort</td>
<td>The master agent port for AgentX, which is the protocol used to communicate with the Helix Server SNMP plug-in. The default is 705.</td>
</tr>
</tbody>
</table>

Setting Up SNMP Security

The following lines set the parameters for SNMP security. The USM security model defines the access rights for each person running the management system. The configuration file predefines two users. The first user operates with no security, which is equivalent to using SNMP version 1. The second user defines authentication and privacy, the highest security under SNMPv3. You can modify or delete these predefined users, as well as create additional users by adding new `<users>...</users>` lists within the USM section:

```xml
<security CommunityString="public"/>
<SecurityModel ModelType="USM">
    <users UserName="xxx">
        <Authentication Type="MD5" Password="yyy"/>
    </users>
</SecurityModel>
```
The following table defines the master agent configuration attributes that define the SNMP security level and permissions.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>CommunityString</td>
<td>Password used with SNMP version 1 or 2. You can ignore the &lt;security/&gt; tag if you are using SNMP version 3. If you are using version 1 or 2, you can ignore the USM settings.</td>
</tr>
<tr>
<td>UserName</td>
<td>Name of the user as defined in the management system.</td>
</tr>
<tr>
<td>Authentication Type</td>
<td>Type of authentication used with SNMPv3. Valid values are MD5 for the HMAC-MD5 algorithm, or SHA for the HMAC-SHA algorithm. A value of NONE indicates an unsecured user.</td>
</tr>
<tr>
<td>Privacy Type</td>
<td>For privacy type, you can enter NONE for no privacy or DES for CBC-DES encryption.</td>
</tr>
<tr>
<td>Password</td>
<td>Password for authentication or privacy. SNMPv3 uses separate passwords for authentication and privacy. You do not need to define a certain password, however, if you used NONE as the authentication or privacy type.</td>
</tr>
</tbody>
</table>

### Defining a View Access Control Model

The view access control model (VACM) available through SNMPv3 allows you to define precisely which Helix Server SNMP objects each viewer can see and control. VACM is optional, and you should be familiar with how it works within your SNMP management system before you define view privileges through the master agent configuration file. The following sections provide an example of how to set up view access for a specific user.

#### Assigning a User to a Group

Each person who uses VACM must be defined in the <SecurityModel> list as an SNMPv3 user. The following example shows a user defined to use SNMPv3 with authentication but no privacy:

```xml
<SecurityModel ModelType="USM">
  <users UserName="Maria">
    <Authentication Type="MD5" Password="tl73jkl98"/>
    <Privacy Type="NONE" Password=""/>
  </users>
  ....other users defined here...
</SecurityModel>
```

Using a <SecurityToGroup/> tag, you assign each user to a group name that you create. In the following example, the user Maria is assigned to a group named v3Group:

```xml
<SecurityToGroup SecurityModel="USM" User="Maria" Group="v3Group"/>
```
The following table explains the `<SecurityToGroup/>` tag attributes.

### VACM `<SecurityToGroup/>` Tag Attributes

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>SecurityModel</td>
<td>Security model for this user. Choose one of the following:</td>
</tr>
<tr>
<td></td>
<td>– v1 for SNMP version 1</td>
</tr>
<tr>
<td></td>
<td>– v2c for SNMP version 2c</td>
</tr>
<tr>
<td></td>
<td>– USM for SNMP version 3</td>
</tr>
<tr>
<td>User</td>
<td>The user’s name.</td>
</tr>
<tr>
<td>Group</td>
<td>The group to which the user is assigned. Groups are defined with <code>&lt;groups/&gt;</code> tags.</td>
</tr>
</tbody>
</table>

### Creating Groups

Within the `<SecurityModel>` list, a `<groups/>` tag defines each group. A group has three views, indicating which parts of Helix Server the user can monitor and control. In the following example, v3Group is assigned the `fullView` view for receiving traps and reading Helix Server variables. It is part of the `noView` view for writing configuration changes to the Helix Server registry:

```xml
<SecurityModel ModelType="VACM">
  <groups Name="v3Group" SecurityModel="USM" SecurityLevel="1"
    Context="" Notify_View="fullView" Read_View="fullView"
    Write_View="noView"/>
  ...
  ...
</SecurityModel>
```

The following table explains the `<groups/>` tag attributes.

### VACM `<groups/>` Tag Attributes

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The group name. Users are assigned to this group by including the name in the <code>&lt;SecurityToGroup/&gt;</code> tag.</td>
</tr>
<tr>
<td>SecurityModel</td>
<td>Security model for this group. Choose one of the following:</td>
</tr>
<tr>
<td></td>
<td>– v1 for SNMP version 1</td>
</tr>
<tr>
<td></td>
<td>– v2c for SNMP version 2c</td>
</tr>
<tr>
<td></td>
<td>– USM for SNMP version 3</td>
</tr>
<tr>
<td>SecurityLevel</td>
<td>Security level for this group. Choose one of the following:</td>
</tr>
<tr>
<td></td>
<td>– 0 for noAuthNoPriv</td>
</tr>
<tr>
<td></td>
<td>– 1 for authNoPriv</td>
</tr>
<tr>
<td></td>
<td>– 2 for authPriv</td>
</tr>
<tr>
<td>Context</td>
<td>An optional, named subset of object instances in the management information base.</td>
</tr>
<tr>
<td>Notify_View</td>
<td>The name of the view assigned to the group for receiving traps.</td>
</tr>
<tr>
<td>Read_View</td>
<td>The name of the view assigned to the group for reading SNMP objects values corresponding to Helix Server registry values.</td>
</tr>
<tr>
<td>Write_View</td>
<td>The name of the view assigned to the group for writing changes to SNMP object values and thereby changing Helix Server configuration values.</td>
</tr>
</tbody>
</table>
Defining Views

Within the <SecurityModel> list, a <views/> tag defines each view. The view identifies a group of objects by an OID from the management information base (MIB). All objects that fall under that OID are included in the view. In the following example, the fullView view is included while the noView view is excluded, allowing no access:

```
<SecurityModel ModelType="VACM">
  ...groups defined here...
  <views Name="fullView" OID="1.3" Mask="" Included="1"/>
  <views Name="noView" OID="1.3" Mask="" Included="0"/>
  ...more views defined here...
</SecurityModel>
```

The following table explains the <views/> tag attributes.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The view name. Groups are assigned up to three views (Notify_View, Read_View, and Write_View) in each &lt;groups/&gt; tag.</td>
</tr>
<tr>
<td>OID</td>
<td>Object ID of a node. All objects that fall under that node in a tree are available in the view. The MIB file used by the SNMP management system lists the OIDs of all nodes.</td>
</tr>
<tr>
<td>Mask</td>
<td>Optional mask value that applies to the OID. You can use this mask to provide finer control over the objects available in the view.</td>
</tr>
<tr>
<td>Included</td>
<td>A true or false value that includes or excludes the view. Use 1 to make the view available, 0 to turn exclude the view from use.</td>
</tr>
</tbody>
</table>

Running the Master Agent on Windows

On Windows, you can run the master agent as a service or as an application. The following sections explain how to start the master agent in either mode.

Restarting the Master Agent Service

If you installed the master agent as a Windows Service, the agent starts up automatically. If you change the master agent configuration, restart the agent service by locating the master agent service with Settings>Control Panel>Administrative Tools>Services. In the Services dialog, right-click on SNMP Master Agent and choose Restart.

Tip: Right-click SNMP Master Agent and choose Properties to change the master agent operation. Using this dialog, for example, you can disable automatic start-up or restart the service automatically if it fails.

Starting the Master Agent as an Application

The following procedure explains how to start the master agent as a Windows application. Do this only if the master agent has not been installed as a service, or you have disabled the service through the Services dialog.

Starting the master agent as a Windows program:

1. Open a command prompt using Start>Program>Accessories>Command Prompt.
2. Navigate to the Helix Server installation directory. For example:
cd “C:\Program Files\Real\Helix Server”

3. The master agent executable, master.exe, resides in the Bin subdirectory. Start it by entering the following:
   Bin\master.exe master.cfg

4. Start Helix Server as described in Helix Media Delivery Platform Quick Start Guide.

Starting the Master Agent on UNIX

The following procedure explains how to start the master agent as a UNIX background process.

To start the master agent on UNIX:

1. If Helix Server is running, shut it down as described in Helix Media Delivery Platform Quick Start Guide.

2. Log in as root.

3. From the command line, navigate to the Helix Server installation directory. For example:
   # cd /usr/local/Real/HelixServer

4. The master agent executable, master, resides in the Bin subdirectory. Start it as a background process:
   # ./Bin/master master.cfg &

5. Start Helix Server as described in Helix Media Delivery Platform Quick Start Guide.

Running a Management System

Once you have the SNMP plug-in and master agent configured and running, you can use your management system to monitor and control Helix Server. From the management system, locate the MIB file, which is named helixserver.my and resides in the Helix Server installation directory. Compile the MIB file if necessary for your management system. You then connect the management system to the master agent using the Helix Server IP address and port defined in the master agent configuration file.

For More Information: For information about compiling the MIB file and connecting your management system to the master agent, refer to your SNMP manager documentation. The section “Defining Master Agent Addresses and Ports” on page 12 explains the master agent port usage.

Monitor Tree

The MIB file produces monitoring trees that allow you to monitor Helix Server operation and control the settings of certain variables. The hsMonitor tree contains objects related to Helix
Server monitoring. (On Helix Proxy, it is the `hpMonitor` tree.) These objects, described in the following table, cannot be changed by the management system.

### Monitor Tree Objects

<table>
<thead>
<tr>
<th>Object</th>
<th>Information</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>hsClients</code></td>
<td>Number of media players currently connected. Covers all supported players and communications protocols.</td>
</tr>
<tr>
<td><code>hsRTSPClients</code></td>
<td>Number of media players currently communicating through the RTSP control protocol.</td>
</tr>
<tr>
<td><code>hsRTMPClients</code></td>
<td>Number of media players currently communicating through the RTMP control protocol.</td>
</tr>
<tr>
<td><code>hsHTTPClients</code></td>
<td>Number of media players currently communicating through the HTTP control protocol.</td>
</tr>
<tr>
<td><code>hsSecureClients</code></td>
<td>Number of media players currently communicating through the HTTPS or RTMPS control protocol.</td>
</tr>
<tr>
<td><code>hsUDPTransports</code></td>
<td>Number of media players currently using User Datagram Protocol (UDP).</td>
</tr>
<tr>
<td><code>hsTCPTransports</code></td>
<td>Number of media players currently using Transmission Control Protocol (TCP).</td>
</tr>
<tr>
<td><code>hsMulticastTransports</code></td>
<td>Number of media players connected on multicast.</td>
</tr>
<tr>
<td><code>hsBandwidthUsage</code></td>
<td>Total outgoing bandwidth in Kilobits per second (Kbps) used by Helix Server.</td>
</tr>
<tr>
<td><code>hsPercentCPUUsage</code></td>
<td>Percentage of CPU used by Helix Server processes.</td>
</tr>
<tr>
<td><code>hsMemoryUsage</code></td>
<td>Amount of memory used by Helix Server in bytes.</td>
</tr>
<tr>
<td><code>hsPlatform</code></td>
<td>Helix Server operating system.</td>
</tr>
<tr>
<td><code>hsVersion</code></td>
<td>Helix Server software version.</td>
</tr>
<tr>
<td><code>hsAccumulatedBandwidth</code></td>
<td>Network bandwidth use in Kilobits per second</td>
</tr>
<tr>
<td><code>hsEncoderCount</code></td>
<td>Number of media encoders currently delivering live streams.</td>
</tr>
<tr>
<td><code>hsUptime</code></td>
<td>Time since last Helix Server restart in seconds.</td>
</tr>
<tr>
<td><code>hsMMSClientCount</code></td>
<td>Number of media players currently communicating through the MMS control protocol.</td>
</tr>
</tbody>
</table>

### Configuration Tree on Helix Server

The `hsConfig` tree contains objects that map to Helix Server configuration variables, such as Helix Server ports and the various traps that can be set. You can monitor these objects using any version of SNMP.

### Helix Server Configuration Tree Objects

<table>
<thead>
<tr>
<th>Object</th>
<th>Information</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>hsPorts</code></td>
<td>This subtree allows you to change the main Helix Server communications ports.</td>
</tr>
<tr>
<td><code>hsTrap</code></td>
<td>The objects in this subtree correspond to the event traps defined for the SNMP plug-in. For more information, refer to “Configuring the SNMP Plug-In” on page 11.</td>
</tr>
</tbody>
</table>
Tip: A configuration change occurs immediately without the need to restart Helix Server. However, to make the change permanent, you need to write the change to the configuration file using the Control Tree.

Configuration Tree on Helix Proxy

The `hpConfig` tree contains objects that map to Helix Proxy configuration variables. You can monitor these objects using any version of SNMP.

<table>
<thead>
<tr>
<th>Object</th>
<th>Information</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>hpMaxConnections</code></td>
<td>Maximum number of clients that Helix Proxy will accept.</td>
</tr>
<tr>
<td><code>hpMaxGatewayBandwidth</code></td>
<td>Maximum gateway bandwidth allowed by Helix Proxy.</td>
</tr>
<tr>
<td><code>hpMaxProxyBandwidth</code></td>
<td>Maximum Helix Proxy bandwidth allowed.</td>
</tr>
</tbody>
</table>

Control Tree

The `hsControl` tree contains objects that you can use to control certain Helix Server operations. By setting `hsRestartServer` to true, for example, you can restart Helix Server remotely.

<table>
<thead>
<tr>
<th>Object</th>
<th>Information</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>hsWriteConfigFile</code></td>
<td>Writes configuration changes made by the management system to the configuration file when set to true or 1. This ensures that the configuration changes remain in effect after a restart. For more on the configuration file, refer to the Helix Server online help.</td>
</tr>
<tr>
<td><code>hsStopServer</code></td>
<td>Stops Helix Server when set to true or 1. For information on shutdown options, refer to the Helix Server online help.</td>
</tr>
<tr>
<td><code>hsRestartServer</code></td>
<td>Restarts Helix Server when set to true or 1.</td>
</tr>
</tbody>
</table>
Splitting enables one component to deliver a live or simulated live broadcast stream to another component. This allows Helix Producer to transmit a stream to one or more Helix Servers, for example, and a Helix Server to distribute a stream to additional Helix Servers. As a result, splitting fosters higher-quality broadcasts by distributing broadcast streams closer to viewers.

**Tip:** To distribute on-demand clips from one Helix Server to another, use the content caching feature described in the Helix Administrator online help topic [*Helix Administrator > Content Management > Content Caching*].

**For More Information:** Before you configure a splitting arrangement, be sure that you understand the concepts and procedures of the basic broadcast methods described in [*Helix Media Delivery Platform Quick Start Guide*].

### Understanding Splitting

A splitting arrangement involves encoders, transmitters, and receivers that can be set up for push splitting or pull splitting.

#### Definitions

**Encoder**
An encoder is software that generates a live or simulated live stream. RealNetworks encoders include Helix Producer and SLTA. Additional, supported encoders include Windows Media Encoder, Sorenson Broadcaster, Flash encoders, and standards-compliant MPEG-4 or 3GPP encoders.

**Encoder-to-Server Splitting**
Splitting technology is used with SLTA or Helix Producer broadcasting in Helix Advanced Push, Helix Multicast Push, or Helix Pull mode. In these cases, the encoder is the transmitter and Helix Server is set up as a receiver.

**Receiver**
A receiver is a Helix Server that acquires a stream from a transmitter and broadcasts it to media players. The transmitter may be another Helix Server, Helix Producer, or SLTA.

**Relay**
A relay is a Helix Server that functions as both a receiver and a transmitter. It acquires a stream from a transmitter and retransmits that stream to another receiver. It may also broadcast the stream to media players.
Pull Splitting
In pull splitting, a receiver initiates the splitting session by requesting the stream from the transmitter. This typically occurs when the first media player requests the stream from the receiver.

Push Splitting
In push splitting, a transmitter initiates the splitting session by delivering the broadcast stream to one or more receivers. The session starts as soon as the encoder begins to generate stream packets.

Server-to-Server Splitting
In server-to-server splitting, a Helix Server transmitter unicasts or multicasts a live or simulated live stream to one or more Helix Server receivers. This allows you to distribute the same stream to multiple Helix Servers across an extended network.

Transmitter
A transmitter is a Helix Server that forwards a stream to a Helix Server receiver. Helix Producer and SLTA can also function as transmitters.

Push Splitting
The following illustration shows the push splitting. Here the encoder delivers the stream to the transmitter, which initiates a connection to the receiver. When a media player requests the broadcast, the receiver is ready to deliver the stream. In this form of splitting, Web page links typically point to the receiver. However, a transmitter can also replicate the broadcast stream to media players.

Note: A transmitter can pull the stream from a pull-enabled encoder but push it to receivers. This lets you start the encoder without making the broadcast available to media players. Use a private URL on the transmitter to pull the stream from the encoder when the broadcast should start. Once this happens, the transmitter automatically pushes the stream to receivers.

Pull Splitting
The following illustration shows pull splitting. Here the transmitter does not deliver the stream to the receiver until the first media player makes a request (step 1). There’s a slight delay as the receiver requests (step 2), receives (step 3), and delivers (step 4) the stream. After that, the stream is live on the receiver, and subsequent player requests do not involve the session setup delay of step 2.
Pull Splitting

Although pull splitting results in greater latency than push splitting on the first stream request by a media player, it can save on bandwidth because the stream is not transmitted to the receiver if no media player requests are active on the receiver. As well, if all media players disconnect from the receiver before the broadcast ends, the data stream between transmitter and receiver is dropped. Hence pull splitting does not consume bandwidth between transmitter and receiver if no one is viewing the broadcast on that receiver.

**Tip:** You can combine push and pull splitting for optimal results. Suppose that you are delivering a broadcast across many different time zones. You could push the stream to receivers that reside in daytime zones. Where it’s late at night and there are fewer potential viewers, you could have receivers pull the stream only on viewer request.

Push Encoding with Pull Splitting

If you set up pull splitting between transmitters and receivers, you can still use Helix Producer or SLTA to push the stream to the transmitter. This cues the broadcast so that the transmitter can respond faster to pull-splitting requests from receivers.

One-to-Many Splitting

A common splitting arrangement uses a single transmitter to broadcast to multiple receivers. If you do this through unicasting, each receiver gets a unique stream, so bandwidth consumption increases with each receiver. Multicasting uses less bandwidth, and is a better solution if all components are on a multicast-enabled network.

The following illustrations shows unicasting through push splitting, though you can also use pull splitting. Here, each receiver connects to the transmitter only when it needs the stream. Server-to-server multicasting is not available with pull splitting, however.

One-to-Many Splitting
Tip: Helix Producer and SLTA can function as transmitters that deliver streams to multiple Helix Server receivers. In this case, a Helix Server is not needed as a transmitter. To reduce outgoing bandwidth, use multicast delivery between the Helix Server receivers and the Helix Producer or SLTA transmitter.

One-to-One Chaining

Another option is to use one-to-one chaining, in which each receiver transmits to another receiver. Receivers in the middle of the chain thereby function as relays. This option is viable if a group of servers is spread across a wide area, and uses unicasting over the Internet to communicate. A transmitter in San Francisco might push a stream to a Tokyo receiver, which pushes it to a Sydney receiver, and so on.

Although you can use pull splitting with a relay chain, push splitting suits this setup better. With pull splitting, there may be a long latency period if the first broadcast request comes from far down the chain. In this case, the request has to make its way back the chain, causing each receiver in the chain to pull the stream from the preceding transmitter.

Tip: Pull-splitting links for a relay chain may become long and increase the likelihood of incorrect broadcast URLs. See the Helix Administrator online help topic Helix Administrator > Broadcast Distribution > Splitting Basics > Links to Split Content > URL Aliases for information about using URL aliases to shorten broadcast URLs.

Setting Up Split Broadcasts

You can split most types of broadcast streams originating from supported encoders. The broadcast mount points and set-up methods may vary with each type of broadcast method, however.

For More Information: The Helix Administrator online help section Helix Administrator > Broadcast Distribution > Splitting Basics > Links to Split Content explains URLs for split broadcast streams.

Helix Producer Broadcasts

You can split any stream originating from Helix Producer, including RealMedia and H.264/AAC in an MPEG-4 or 3GPP output. The broadcast mount point for these streams is /broadcast/.

For More Information: For details about how to set up a specific type of broadcast on Helix Producer, refer to its online help.
Helix Push Broadcasts

The Helix Push broadcast method is the most basic method for delivering a stream from Helix Producer:

- In the Broadcast Distribution > Transmitter page on that Helix Server, you define the server as a transmitter.
- The other Helix Servers that receive the stream are defined as receivers in their Broadcast Distribution > Receiver pages.

Helix Advanced Push and Helix Multicast Push

The Helix Advanced Push and Multicast Push broadcast methods are virtually identical except that they use the transport udp/unicast or udp/multicast throughout the splitting chain:

- The Helix Server that receives the stream from Helix Producer is defined as a receiver in its Broadcast Distribution > Receiver page. Helix Producer acts as the transmitter in this setup.
- In the Broadcast Distribution > Transmitter page on the Helix Server that gets the stream from the encoder, you define the server as a transmitter. The server that receives the stream from Helix Producer therefore functions as a relay.
- The other Helix Servers that receive the split stream are defined as receivers in their Broadcast Distribution > Receiver pages.

Helix Pull

In the Helix Pull broadcast method, Helix Producer is a pull-enabled transmitter that transmits packets only upon request from pull-enabled Helix Servers receivers:

- The Helix Server that pulls the stream from Helix Producer is defined as a pull-enabled receiver in its Broadcast Distribution > Receiver page.
- Additional Helix Servers can pull the stream from the initial Helix Server receiver by defining that receiver as a pull-source in their Broadcast Distribution > Receiver pages.

SLTA Broadcasts

The SLTA utility allows you to stream on-demand clips from a Helix Server machine as if they were live broadcasts:

- The broadcast mount point for these streams is /broadcast/.
- In its basic mode, SLTA functions like Helix Producer in a Helix Push broadcast.
- In its advanced mode, SLTA can function like Helix Producer in a Helix Advanced Push, Helix Multicast Push, or Helix Pull broadcast.

For More Information: See the SLTA section of the Helix Administrator online help.

RTP-based Broadcasts

You can split a stream from most third-party MPEG-4 or 3GPP encoders:
• The mount point for these broadcasts is /rtpencoder/.
• You define the encoding method in the Helix Administrator Broadcasting > QT & RTP Encoding page for the Helix Server that receives the stream from the encoder.
• In the Broadcast Distribution > Transmitter page on that Helix Server, you define the server as a transmitter.
• The other Helix Servers that should receive the stream are defined as receivers in their Broadcast Distribution > Receiver pages.

**Flash Broadcasts**

Flash broadcasts that encode H.264/AAC streams can be split to different receivers:
• The mount point for these broadcasts is /rtsplive/rtmplive/.
  • The /rtmplive/ mount point signifies a broadcast from a Flash encoder.
  • The /rtsplive/ mount point precedes /rtmplive/ to allow Helix Server to repackage the stream as needed for splitting.
• You define the Flash broadcasting method in the Helix Administrator Broadcasting > Flash Media Encoding page for the Helix Server that receives the stream from the Flash encoder.
  
  **Note:** The encoder setup page includes an option to allow repackaging to an RTSP-based format. This option must be set to Yes to enable splitting.
• In the Broadcast Distribution > Transmitter page on the Helix Server where the Flash encoder connects, define the server as a transmitter.
• The other Helix Servers that should receive the stream are defined as receivers in their Broadcast Distribution > Receiver pages.

**Windows Media Broadcasts**

Windows Media push or pull broadcasts can be split to Helix Server receivers:
• The broadcast mount point is /wmtencoder/.
• You set up the Windows Media encoder connection (push or pull) in the Helix Administrator Broadcasting > Windows Media Encoding page for the Helix Server that receives the stream from the encoder.
• For a push broadcast:
  • Define the Helix Server that receives the stream from the encoder as a transmitter in its Broadcast Distribution > Transmitter page.
  • The other Helix Servers that receive the stream are defined as receivers in their Broadcast Distribution > Receiver pages.
• For a pull broadcast:
  • The Helix Server that pulls the stream from the encoder is defined as a pull-enabled receiver in its Broadcast Distribution > Receiver page.
• Additional Helix Servers can pull the stream from the initial Helix Server receiver by defining that receiver as a pull-source in their **Broadcast Distribution > Receiver** pages.

### Splitting Different Streams to Different Receivers

You may set up your transmitters and receivers to split every broadcast the same way. In this case, you always use a certain Helix Server as your primary transmitter. The other Helix Servers always function as receivers or relays. This is not necessary, however, and a single network of Helix Servers can support numerous splitting arrangements, enabling you to transmit from any Helix Server. As well, you can split broadcasts from a single transmitter in many different ways.

When you set up splitting, you can create multiple transmitter and receiver definitions on each Helix Server. When you set up a push transmitter, for example, you define how it connects to each receiver. For a single Helix Server receiver on one physical machine, you can create multiple receiver definitions. Flash broadcasts can use one definition, for example, while MPEG-4 broadcasts use another. This lets you multicast one format, for instance, while unicasting another.

### Virtual Paths for Stream Direction

You can use multiple receiver definitions to split broadcasts in different ways according to the stream source names that appear in broadcast URLs. Each source name has three parts:

1. **mount point**—Every broadcast uses a mount point. Broadcasts pushed by Helix Producer use the `/broadcast/` mount point, for example, while broadcasts from a standards-based MPEG-4 encoder use `/rtpencoder/`.

2. **virtual path**—An optional path inserted between the mount point and the stream name allows you to selectively split broadcast streams on the same mount point. How you define the path differs for each media format:
   - With RealNetworks encoders, you define the path name when setting up the broadcast method.
   - For Windows Media pull broadcasts, you define the path when configuring the encoder pull session on Helix Server. If using push encoding, you define the path through Windows Media Encoder.
   - For broadcasts from RTP-based encoders, the path name reflects the subdirectory where the encoder placed the SDP file.
   - Flash encoders precede the stream name with an application name that functions as the virtual path.

3. **stream name**—In all broadcasts, the stream name appears last in the URL and looks like an on-demand clip name, often ending with the media format’s standard file extension.

### Stream Routing Example

The following illustration shows how you can use mount points, paths, and stream names within receiver definitions to split different broadcasts in different ways. Three encoders connect to the same transmitter and deliver three separate streams. The first stream, `live.mp4`,...
uses no path. The second and third streams, `news/breaking.mp4` and `news/hourly.mp4`, use the same path name but different stream names.

**Stream Direction through Mount Points, Paths, and File Names**

Each receiver in the preceding illustration uses a different broadcast source path. These path definitions are created on the transmitter, which directs each stream to its receiver:

- The first receiver accepts all broadcasts that use the broadcast mount point, `/broadcast/`. In this example, it receives all three broadcast streams.
- The second receiver gets all broadcasts that use the `/broadcast/news/` mount point and path. It therefore receives streams 2 and 3 but not stream 1, which does not use the `news/` path.
- The third receiver gets only the broadcast stream that uses the `/broadcast/news/hourly.mp4` mount point, path, and stream name.

**Multicasting Split Streams**

Helix Producer, SLTA, and Helix Server support multicast delivery if all hardware components are on a multicast-enabled network. The following sections provide examples of how to use multicasting to create efficient splitting arrangements.

**Unicast Delivery, Multicast Distribution**

To deliver a split stream, you can combine unicasting and multicasting. For example, you can unicast streams to receivers across the Internet, an intranet, a wide area network, or a local area network. Then, within an intranet, you can multicast the stream from the receivers to media players. The following illustration shows this type of delivery.
Unicast Delivery, Multicast Distribution

Dual Unicast and Multicast Transport Methods

If your server network is multicast-enabled, you can simultaneously unicast and multicast a broadcast stream to receivers. Duplicate packets arriving by different transport methods increase the network overhead but do not cause problems for receivers. When a receiver reassembles the broadcast stream, it uses the packets that arrive first, regardless of their transport methods. If a unicast packet is late or missing, for example, the receiver may get the right packet through multicast. The following figure illustrates this dual delivery.

Broadcasting by Multiple Methods

Tip: Server-to-server multicasting requires a multicast-enabled network just like server-to-player multicasting. To configure server-to-server multicasts, you simply select multicast as your transport method when configuring transmitters and receivers.

Creating Redundant Streams

Helix Server supports encoder redundancy for all media formats. This redundancy carries over automatically with splitting, requiring no special configuration for transmitters and receivers. As shown in the following illustration, separate encoders deliver the same stream (delimited with an integer, as in live.mp4.1 and live.mp4.2) to the transmitter. If the primary stream fails,
the transmitter uses the backup stream. It sends the receiver one stream, which may come from either the primary or the backup encoder.

**Simple Redundant Source for Splitting**

For More Information: Refer to the Helix Administrator online help topic *Helix Administrator > Broadcasting > Broadcast Redundancy* for details about broadcast redundancy.

**Transmitter Redundancy**

The setup in the preceding illustration provides a single level of encoder redundancy. You can increase redundancy within a splitting arrangement by adding transmitter redundancy, as shown in the following illustration. Here, both the primary and backup encoders send streams to two Helix Servers that each split the stream to three receivers. (Helix Producer and SLTA can push the same stream to multiple Helix Servers.) Under normal conditions, each receiver gets a version of *live.mp4* from each transmitter.

**Redundant Sources and Redundant Transmitters**

If the primary encoder fails, both transmitters switch to the stream from the backup encoder. Again, the receivers get two streams, one from each transmitter. Note that in this configuration, each receiver still receives a stream even if one encoder *and* one transmitter fail.
This provides both encoder and transmitter redundancy. Three out of the four encoder and transmitter components would have to fail for the entire broadcast to fail.

**Transport Redundancy**

The following illustration shows multiple encoders and delivery methods used to provide encoder and transport redundancy. The primary and backup encoders both unicast a separate stream to each receiver, as well as multicast the broadcast stream to all receivers. Each receiver gets four streams. It uses the primary live.mp4.1 stream as long as those packets arrive by either unicast or multicast. If the primary encoder fails, or its transport methods are blocked, each receiver switches to the live.mp4.2 backup sent over unicast or multicast.

Redundant Encoders Using Unicasting and Multicasting

Note that the receivers in the preceding illustration could also function as transmitters that split streams to other receivers, as described in “Transmitter Redundancy” on page 28. This would provide three layers of redundancy at the encoder, transmitter, and transport levels. Although such a complex arrangement and high degree of redundancy is generally not necessary, RealNetworks components provide support for all of these layers, which you can put together as needed.
CHAPTER 4

PLAYLIST MANAGEMENT

This chapter explains server-side playlist management, which streams a sequence of clips or broadcasts to RTSP-based or HLS media players. For certain types of playlists sent to RTSP media players, the viewer can instruct Helix Server to skip to different playlist entries.

Note: DASH and Flash clients are not supported for server-side playlist management.

Understanding Playlist Management

To create a playlist, a viewer interacts with a third-party Web portal to define a sequence of on-demand clips, live broadcasts, or simulated-live broadcasts. The portal creates a text file (extension .hpl) that uses the XML-based SMIL markup to define the playlist features. It then uploads the playlist to Helix Server. When the viewer requests the playlist, the media selections play sequentially.

Playlist of Four Clips Streaming Sequentially

The preceding figure illustrates a set of four on-demand clips lasting three minutes apiece. The entire presentation appears to the media player as a single, 12-minute session. RTSP media players receive the content in a single RTSP session while HLS clients maintain a persistent HTTP connection. As the playlist streams, transitions between the clips occur seamlessly, without the need for rebuffering.

Types of Playlist Sessions

Helix Server supports the following types of playlist management sessions:

- externally controlled (RTSP media players only)
  In this type of session, the viewer can skip to various parts of the playlist by sending HTTP directives to Helix Server outside of the RTSP connection.
- internally controlled (RTSP media players and HLS clients)
An internally controlled playlist appears to the viewer to be a single on-demand clip. The viewer can seek through the playlist using the media player’s seek commands.

- noncontrolled (RTSP media players and HLS clients)

A noncontrolled playlist session appears to be a single live stream. The viewer cannot skip or seek through the contents.

Externally Controlled Session

Available with RTSP media players, an externally controlled session appears to be a single live stream even if the playlist contains no live broadcasts. The media player’s timeline controls don’t function, and RTSP seeking is disabled. However, the viewer can issue seek and skip commands outside of the RTSP channel by clicking HTTP links that are proxied to Helix Server by a third-party Web server. Playlist control is therefore external to the RTSP channel.

To support an externally controlled session, a playlist must include certain features such as chapter markers that designate the allowed skip points. The RTSP URL used to request the playlist must also contain specific query string parameters, such as an ID value that allows Helix Server to identify the RTSP session referred to in HTTP-based skip directives.

For More Information: See “Externally Controlled Playlist Sessions” on page 38.

Internally Controlled Session

In an internally controlled playlist session, a viewer using an RTSP media player or an HLS client can seek through the playlist by issuing seek commands directly to Helix Server. Hence, playlist control is internal to the media stream. External, HTTP-based directives sent by a third-party Web server are not supported. For a playlist to be internally controllable, it must consist entirely of on-demand media streams that have defined durations. This gives the playlist an overall timeline and allows it to function like a single clip.

Tip: Playlists for internally controlled sessions do not require features such as chapter markers. However, a playlist designed for externally controlled sessions will work for internally controlled or noncontrolled sessions. Helix Server sets the session type based on the playlist contents and the elements in the request URL. It ignores any playlist components not supported by the session type.

For More Information: See “Internally Controlled and Noncontrolled Playlist Sessions” on page 36.

Noncontrolled Session

In a noncontrolled playlist session, an RTSP media player or HLS client treats the entire session as if it were a single, live stream. No seek commands are allowed. The viewer does not have control over the timeline other than to start and stop the playlist session.

The playlist components determine if the session is internally controlled or noncontrolled. A noncontrolled session is used whenever Helix Server cannot calculate the overall playlist timeline, which is necessary for it to manage seeking through a playlist. Including any of the following in the playlist results in a noncontrolled session:

- live or simulated-live broadcast
A playlist that includes at least one live or simulated-live broadcast functions as a noncontrolled session because the indeterminate timeline of a broadcast prevents Helix Server from calculating an overall playlist timeline.

- undefined clip duration
  If one or more on-demand clips do not have durations defined in the playlist, Helix Server cannot calculate seek times across clips. It therefore streams the playlist contents in a noncontrolled session.

- non-skippable clip
  If at least one clip in the playlist has been declared non-skippable (see page 50), Helix Server streams the playlist contents in a noncontrolled session. This default action can be modified, however.

**Web Portal Requirements**

The Web portal is a third-party application that gives users access to playlists. It may provide standard playlists for all users or allow each user to create a customized playlist. The portal submits playlists to Helix Server according to the workflow described in the section “Initial Playlist Selection” on page 38.

*Note:* The user builds the playlist and issues skip directives by interacting with the Web portal. The playlist management feature does not set any specific browser requirements.

**Basic Portal Requirements**

Minimally, the Web portal must be able to do the following:

- Process any necessary registration and billing information required from the viewer.
- Configure playlists in the XML-based SMIL format described as the section “Playlist Format” on page 45.
- Upload playlists to the required Helix Server mount point as described in the section “Uploading Playlists” on page 54.

**Requirements for Managing Externally Controlled Playlists**

To implement externally controlled playlist sessions, the portal must also do the following:

- Issue HTTP or HTTPS directives to the Helix Server controller port as described in the section “Externally Controlled Playlist Sessions” on page 38.
- Add required parameters to an SDP file returned by Helix Server as explained in the section “Initial Playlist Selection” on page 38.

**Stream Requirements for RTSP Players**

Any media player that is compliant with the RTSP standard and that supports the chosen streaming format should be able to render a playlist session. Because an RTSP player cannot switch its decoding method within an RTSP session, all streams within a single playlist must be encoded using the same streaming rates and the same codec (same profile and level).
Tip: Because stream quality can differ across encoders, creating content using the same hardware or software encoder helps to prevent streaming errors.

Supported Codecs

Playlist management for RTSP players works with the following codecs.

**Video Codecs**
- H.264 (recommended)
- H.263
- MPEG-4
- RealVideo

**Audio Codecs**
- AAC or AAC+ (recommended)
- Enhanced AAC+
- AMR-NB or AMR-WB
- RealAudio

Supported File Formats

Supported file formats for RTSP Players are the following:
- MPEG-4 (.mp4 and variants)
- 3GPP (.3gp)
- RealMedia (.ra, .rv, .rm)
- QuickTime (.mov)
- F4V (.f4v)

Tip: Playlist management does not function with Flash clients, which use the RTMP protocol. However, you can use the F4V format when streaming to RTSP clients that support the H.264 and AAC codecs used to compress the media.

Unsupported Formats

The following formats are **not** supported:
- Windows Media and the MMS streaming protocol
- Flash FLV clips (.flv)
- *any* format that use digital rights management (DRM) protection

Protocols and Transports

Playlist management for RTSP players works with RTSP and RTSP streams cloaked as HTTP. The network transport can be either UDP or TCP. Playlist sessions support both standards-based RTP and the RealNetworks RDT data packet format, which is used with RealPlayer and Helix-based media players.
Segment Handling with HLS Clients

Playlist sessions for HLS clients can use the same .hpl playlist used with RTSP media players as long as the content is codec-compatible across all devices. As noted in the following sections, though, HLS clients handle live broadcasts and rate changes between clips differently than RTSP players. This can affect how playlists are structured.

**Note:** Playlist management supports iPhone, iPod Touch, and iPad devices running iOS 3.0 or later. Other devices that support HTTP live streaming in accordance with Apple specifications may also be able to use playlists.

**For More Information:** See the Helix Administrator online help for information about how Helix Server generates transport segments and supports HLS clients. Media segmentation must be configured before you can deliver playlist sessions to HLS clients.

/m3ugen/ Mount Point

An HLS client must request the .hpl playlist using the /m3ugen/ mount point, which causes Helix Server to segment the resource and return an .m3u8 playlist.

**Note:** Only the request URL for the .hpl playlist uses the /m3ugen/ mount point. Within the .hpl playlist, media resources are designated by relative URLs that do not include the /m3ugen/ mount point.

**For More Information:** For .hpl playlist examples, see “Specifying the Media Source” on page 48. The Helix Administrator online help explains how to use /m3ugen/ with HLS client requests.

Codecs and Rate Handling

Content for HLS clients must be encoded as H.264 video and AAC audio. Unlike RTSP players, HLS clients can switch streaming rates within a playlist. For example, an HLS client can play one stream encoded at 100 Kbps and then play another stream at 200 Kbps. In an .m3u8 playlist, Helix Server indicates where bandwidth options change, allowing the HLS client to choose streaming options appropriately.

**Tip:** To make a single playlist accessible to both RTSP media players and HLS clients, use H.264/AAC encoding at the same bandwidth (or set of bandwidths), profile, and level for all media sources.

**For More Information:** The Helix Administrator online help provides more information about the encoding requirements for HLS clients.

Broadcasts

The .hpl playlist for an HLS client may contain one (and only one) link to a live or simulated live broadcast. Because the cessation of a broadcast stream closes the device’s persistent HTTP connection, the broadcast URL must be the last media resource in the playlist.

**Note:** When a live broadcast is used, the request URL must include a GUID that enables Helix Server to keep track of the session. See “Session ID” on page 43 for details about GUID values.
Segment Creation

Requesting a clip or live broadcast through an .hpl playlist does not alter the location or naming of segment files. If segments for song100.mp4 already exist on Helix Server, for example, those segments are used if song100.mp4 is requested as part of an .hpl playlist. If Helix Server segments song100.mp4 because of an .hpl request, those segments are used if an HLS client later requests song100.mp4 outside of a playlist management session.

For More Information: The section Helix Administrator online help explains the default location and naming conventions for segments and playlists.

Session Playlists

When an HLS client requests an .hpl file, Helix Server returns to the device a playlist or set of playlists in the .m3u8 format:

- An internally controlled session has an overall length calculated by Helix Server. The .m3u8 playlist returned to the HLS client lists all segments for all media sources, noting where encoding rates change.

- A noncontrolled session does not have a known, overall length. Helix Server treats it as a live broadcast and sends the HLS client a series of .m3u8 playlists. Each playlist provides the URLs for three segments and indicates where encoding rates change.

The .m3u8 playlist or playlists generated from the .hpl playlist use the location and naming conventions described in Helix Administrator online help section Media Types > HLS Clients > HLS File Names. As noted in the section “Segment Creation” on page 35, the media segments are not stored along with the master session playlist.

Compatibility with Other Features

The following table summarizes the compatibility of playlist management with other Helix Server features.

<table>
<thead>
<tr>
<th>Other Feature</th>
<th>Compatibility Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>bookmarking</td>
<td>Bookmarking works with internally controlled sessions that appear to be a single on-demand clip. For more on bookmarking, refer to the Helix Administrator online help.</td>
</tr>
<tr>
<td>reduced start-up delay</td>
<td>Reduced start-up delay is compatible with playlist management. It applies to RTSP clients only.</td>
</tr>
<tr>
<td>rate adaptation</td>
<td>Rate adaptation for on-demand clips and live broadcasts is compatible with playlists streamed to RTSP media clients. See also “Rate Control Flag” on page 45.</td>
</tr>
<tr>
<td>content caching</td>
<td>You can use the content caching mechanism to propagate source clips and .hpl playlists between multiple Helix Servers on a network.</td>
</tr>
<tr>
<td>multicasting</td>
<td>Playlists are not compatible with multicasts.</td>
</tr>
<tr>
<td>splitting</td>
<td>Playlist sessions can include split, live streams, which Chapter 3 describes. However, there will be a delay if a request is for a pull-split stream that has not yet been pulled on the receiver that manages the session user’s session.</td>
</tr>
<tr>
<td>basic logging</td>
<td>To record playlist management events in the access log file, use basic logging style 8.</td>
</tr>
</tbody>
</table>

(Table Page 1 of 2)
Internally Controlled and Noncontrolled Playlist Sessions

The set-up and management for internally controlled and noncontrolled sessions are virtually identical. The following sections explain the workflow for managing an internally controlled or noncontrolled playlist session for an RTSP player or HLS client. These types of sessions do not accept HTTP requests proxied through a third-party Web server.

Initial Playlist Selection

The following is the sequence of events that occurs when a user defines and requests a new playlist.

Requesting a Playlist (Internally Controlled or Noncontrolled Session)

<table>
<thead>
<tr>
<th>Other Feature</th>
<th>Compatibility Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>advanced</td>
<td>You can capture playlist management events using the customized logging</td>
</tr>
<tr>
<td>logging</td>
<td>templates. See “Logging Styles and Variables” on page 55.</td>
</tr>
<tr>
<td>Helix Proxy</td>
<td>Helix Proxy delivers playlist streams to RTSP clients in pass-through mode only.</td>
</tr>
<tr>
<td></td>
<td>Because the proxy does not accept HTTP requests, a Web portal must send all</td>
</tr>
<tr>
<td></td>
<td>playlist directives to the origin server.</td>
</tr>
</tbody>
</table>

To request a playlist in an internally controlled or noncontrolled session:

1. The user constructs the playlist using the Web portal. The portal handles all necessary authentication, financial, and data transactions.
   
   **Note:** If the .hpl playlist already exists on Helix Server, the next action is Step 5.

2. The Web portal formats the playlist as described in the section “Playlist Format” on page 45. It passes the playlist to Helix Server using an authenticated HTTP or HTTPS POST, which the section “Uploading Playlists” on page 54 describes.
3. Helix Server writes the .hpl playlist to disk.

4. On a successful upload, Helix Server responds to the portal with an HTTP 200 message. Otherwise, it issues an HTTP error as described in the section “Error Conditions” on page 55.

5. Using a browser, the viewer requests a playlist:
   - The user of an RTSP media player requests the .hpl file using the SDPgen utility of Helix Server.
   - The HTTP request of an HLS client includes the /m3ugen/ mount point to segment the content and generate the .m3u8 playlists.

   **For More Information:** For background on SDPgen, refer to the online help topic Streaming Basics > Predefined Mount Points > Client Mount Points. The help topic Media Types > HLS Clients > HLS Media Requests explains how HLS clients request content.

6. Helix Server verifies that the playlist has not expired. It then returns an .sdp file to the RTSP player or an .m3u8 playlist to the HLS client. Optionally, the Web portal may proxy the .sdp file or .m3u8 playlist to the client and update the playlist request URL to include the following parameter:

   guid=ID Sets a user-defined session ID that enables logging of the individual clips in the playlist.
   This is optional for RTSP players using an internally controlled session. It is required for HLS clients if the playlist contains a live broadcast. For details, refer to “Session ID” on page 43 and “Logging and Error Codes” on page 55.

   For example, Helix Server may return the following request URL in the SDP file sent to an RTSP player:
   rtsps://helixserver.example.com/playlists/songs.hpl
   The Web portal updates the URL to look like the following:
   rtsps://helixserver.example.com/playlists/songs.hpl?guid=e624acc1-fa91-11e0-f953-6b6ee7b776b

   **For More Information:** The playlist expiration time is defined within the playlist. See “Playlist Expiration” on page 46.

7. The user’s media player initiates the playlist session using the returned playlist URL. For RTSP devices, clips are delivered in sequence within the RTSP session. An HLS client receives a segment playlist (.m3u8) or a series of playlists.

8. Helix Server begins to stream the media resources listed in the playlist.

9. As the playlist resources stream, Helix Server generates separate entries for each clip or broadcast in its access log.

**Playlist Seeking**

The following figure illustrates playlist seeking in an internally controlled session, which defines an overall playlist length. While watching the first clip, the viewer uses the media
player’s seek controls to seek ahead several minutes, skipping over the second clip entirely and resuming playback in the middle of the third clip.

Seeking Through Clips in a Playlist

Tip: Playlist seeking is disabled in noncontrolled sessions, which appear to the user to be live broadcasts with indeterminate playback lengths.

Note: A seek action cannot identify a chapter ID defined in the playlist or provide an external skip directive such as seek=next. These features are available only with externally controlled sessions.

Externally Controlled Playlist Sessions

The following sections explain the workflow in externally controlled playlist sessions available for RTSP media players. These sessions associate an ID with the RTSP session and can accept HTTP directives to modify the session by skipping or seeking to a different part of the playlist timeline.

Initial Playlist Selection

The following events occur when a user defines and requests a new playlist.
To request a playlist in an externally controlled session:

1. The user constructs the playlist using the Web portal. The portal handles all necessary authentication, financial, and data transactions.

   **Note:** If the .hpl playlist already exists on Helix Server, the next action is Step 5.

2. The Web portal formats the playlist as described in the section “Playlist Format” on page 45. It passes the playlist to Helix Server using an authenticated HTTP or HTTPS POST, which the section “Uploading Playlists” on page 54 describes.

3. Helix Server writes the playlist to disk.

4. On a successful upload, Helix Server responds to the portal with an HTTP 200 message. Otherwise, it issues an HTTP error as described in the section “Error Conditions” on page 55.

5. The Web portal requests the playlist as an SDP file (.sdp) using the Helix Server SDPgen utility.

   **For More Information:** For background on SDPgen, refer to the online help topic Streaming Basics > Predefined Mount Points > Client Mount Points.

6. Helix Server verifies that the playlist has not expired. It then returns a standard SDP file to the Web portal.

   **For More Information:** The expiration time is defined within the playlist. See “Playlist Expiration” on page 46.
7. The Web portal updates the playlist request URL within the SDP file to include the following parameters:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>hpl=1</code></td>
<td>Indicates that the RTSP session is subject to playlist control. See “Playlist Request Marker” on page 43.</td>
</tr>
<tr>
<td><code>guid=ID</code></td>
<td>Sets a user-defined session ID. Subsequent requests from the Web portal to modify the playlist session include this ID. Refer to “Session ID” on page 43.</td>
</tr>
<tr>
<td><code>mdp=1</code></td>
<td>Optional parameter that allows Helix Server to use its server-side rate control mechanism when streaming clips. See “Rate Control Flag” on page 45.</td>
</tr>
</tbody>
</table>

For example, Helix Server may return the following request URL in the SDP file:

```
rtsp://helixserver.example.com/playlists/songs.hpl
```

The Web portal updates the URL to look like the following:

```
rtsp://helixserver.example.com/playlists/songs.hpl?hpl=1&guid=e624acc1-fa91-11e0-f953-6b6ee78b776b
```

8. The Web portal delivers the SDP file to the media player.

9. The media player initiates the RTSP session using the playlist URL contained in the SDP file.

10. Helix Server begins to stream the media resources listed in the playlist. It identifies this RTSP session using the `guid` value from the request URL.

11. As the playlist streams, Helix Server generates entries in its access log.

**For More Information:** As explained in the section “Logging Styles and Variables” on page 55, log entries include the user-defined ID and list the sequence of each clip within the playlist.

### Chapter Skipping

To support chapter skipping in an externally controlled session, the playlist organizes content into **chapters**. Each chapter is typically two or more clips that function as a group. For example, an advertisement and a movie preview may make up a chapter. If the viewer issues an HTTP-based skip command while watching the advertisement, Helix Server skips to the next chapter rather than directly to the movie preview.

**Tip:** The playlist may also designate clips as non-skippable. Directives to skip through these clips are not allowed. See “Making a Clip Non-Skippable” on page 50.

**For More Information:** The section “Setting Up Chapters” on page 51 explains how to organize clips into chapters and assign IDs used with skip directives.

### Chapter Skipping Example

In the following figure, the playlist is organized into two chapters. Each chapter groups together two clips that play in sequence. While watching the first clip in chapter 1, the viewer issues a command to skip to the next chapter. This takes the viewer to the start of the third clip.
Skipping Chapters in a Playlist

Chapter Skipping Directives

Signalling for a chapter skip occurs outside of the RTSP session, typically through an HTTP or HTTPS GET requested directed toward the Web portal.

Process for Skipping to a Different Chapter

To skip to a different position in the playlist:

1. Using a browser, the session viewer requests a jump to a different position in the playlist.

   **Tip:** As depicted above, the HTTP or HTTPS command can go through the Web portal, which then forwards the commands to Helix Server. Alternatively, the link commands may go directly to Helix Server.

   **Note:** The media player does not receive the playlist from Helix Server. To allow skipping, the Web portal must construct an interactive Web page from the playlist values and allow the user to indicate the desired skip points.
2. The session viewer’s skip request is translated into an HTTP or HTTPS GET command sent to the Helix Server playlist control port. To the original request URL, the HTTP or HTTPS directive adds the following required query string parameters:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>hpl=1</td>
<td>Identifies the request as a playlist session directive. See “Playlist Request Marker” on page 43.</td>
</tr>
<tr>
<td>guid=ID</td>
<td>Provides the user-defined ID assigned to the session when the playlist session was initialized. See “Session ID” on page 43.</td>
</tr>
</tbody>
</table>

The request also includes one or more of the following parameters, which indicate where to jump in the playlist:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>seek=point</td>
<td>Indicates a skip point, such as the next chapter. See “Seek” on page 44.</td>
</tr>
<tr>
<td>plref=ID</td>
<td>Identifies a chapter ID within the playlist as the skip point. See “Playlist Reference” on page 44.</td>
</tr>
<tr>
<td>time=ms</td>
<td>Provides a timing offset for the skip. See “Time” on page 44.</td>
</tr>
</tbody>
</table>

For example, an HTTP request directed toward the Helix Server playlist control port may look like the following:

http://helixserver.example.com:8009/playlists/songs.hpl?hpl=1&guid=e624acc1-fa91-11e0-f953-6b6ee78b776b&seek=next

3. Helix Server returns an HTTP response indicating success or failure.


4. On a success, Helix Server skips to the designated position in the playlist timeline.

Note: The skip typically causes rebuffering in the viewer’s player as the new data streams. Once this preroll requirement is fulfilled, however, streaming continues without rebuffering during normal network conditions.

5. Helix Server records the action in its log file.


**Playlist Seeking**

In an externally controlled session, the playlist always appears to be a live broadcast, preventing the media player from seeking through the timeline using RTSP commands. However, the viewer can interact with the Web portal to indicate a seek request. The Web portal then issues an HTTP or HTTPS directive to Helix Server. The directive includes a time parameter to specify the desired position in the overall playlist timeline. For example:

http://helixserver.example.com:8009/playlists/songs.hpl?hpl=1&guid=e624acc1-fa91-11e0-f953-6b6ee78b776b&time=321400

For More Information: See “Time” on page 44.

**Playlist Control Parameters**

For externally controlled playlists, requests to start the playlist or jump to a different position include query string parameters that are summarized in the following table. The last two table
columns indicate if the parameter is required in the initial RTSP request and any HTTP control directives.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Function</th>
<th>Example</th>
<th>RTSP Request</th>
<th>HTTP Directive</th>
</tr>
</thead>
<tbody>
<tr>
<td>Playlist Request Marker</td>
<td>Indicates that the request pertains to a playlist.</td>
<td>hpl=1</td>
<td>required</td>
<td>required</td>
</tr>
<tr>
<td>Session ID</td>
<td>Identifies the playlist RTSP session.</td>
<td>guid=ID</td>
<td>required</td>
<td>required</td>
</tr>
<tr>
<td>Seek</td>
<td>Specifies a different chapter to play:</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>next – next chapter</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>prev – previous chapter</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>first – first chapter</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Playlist Reference</td>
<td>Provides the ID of a playlist chapter to play.</td>
<td>plref=341</td>
<td>not used</td>
<td>optional</td>
</tr>
<tr>
<td>Time</td>
<td>Indicates the time in milliseconds at which to</td>
<td>time=213400</td>
<td>not used</td>
<td>optional</td>
</tr>
<tr>
<td></td>
<td>start playback.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Rate Control Flag</td>
<td>Enables server-side rate control:</td>
<td>mdp=0</td>
<td>optional</td>
<td>not used</td>
</tr>
<tr>
<td></td>
<td>0 – rate control disabled</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1 – rate control enabled (default)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Tip:** The order of query string parameters in a request URL does not matter.

**For More Information:** You can change the parameter names, such as hpl, through the Helix Server configuration file. See the Helix Administrator online help topic **Configuration File > Content Management Configuration > Server-Side Playlist Configuration.**

**Playlist Request Marker**

The parameter hpl=1 indicates that the HTTP or RTSP request is for a playlist session. It must be included in all requests to start or modify an externally controlled playlist session.

**Session ID**

The guid parameter provides a user-defined value that identifies the playlist session. The Web portal is responsible for generating a unique value for each user. It must include this parameter and value pair in each playlist request to enable Helix Server to handle the session. The guid value should be 8 to 32 characters in length, and may contain any of the following characters:

- a-z
- A-Z
- 0-9
- - (hyphen)
Seek

You can include the seek parameter with HTTP-based skip commands to instruct Helix Server to skip to a different clip within a playlist. Clips do not need to have IDs explicitly defined in the playlist for the seek parameter to work. Valid values are the following:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>first</td>
<td>Skip to the first clip in the playlist.</td>
</tr>
<tr>
<td>next</td>
<td>Skip to the next clip or defined skip-point in the playlist.</td>
</tr>
<tr>
<td>prev</td>
<td>Skip to the previous clip or the beginning of the current clip. (See below.)</td>
</tr>
</tbody>
</table>

**Note:** If either the plref or time parameter is also present in the URL, Helix Server ignores the seek value.

**For More Information:** See the section “Setting the Skip Point” on page 50 for more about skip-point markers.

Seeking to the Previous Clip

By default, seek=prev skips back to the start of the current clip if that clip is playing at greater than the 10-second mark in its timeline. If the clip is playing at less than its 10-second mark when Helix Server receives the seek=prev directive, Helix Server skips to the start of the preceding clip.

To summarize:

- Current clip timeline < 10 seconds = skip to the previous clip
- Current clip timeline > 10 seconds = skip to the start of the current clip

**Tip:** You can change the default value from 10 seconds by editing the PreviousSeekTimeout variable in the Helix Server configuration file. See the Helix Administrator online help topic *Configuration File > Content Management Configuration > Server-Side Playlist Configuration.*

Playlist Reference

The plref=ID value instructs Helix Server to skip to a different clip or chapter in the playlist. The parameter value matches the case-sensitive ID assigned in the clip tag or the sequence tag. You can pair this parameter with a time=ms parameter to fast-forward to a specific position in the new chapter’s timeline.

**For More Information:** For more on chapter IDs, see “Defining a Chapter or Clip ID” on page 51.

Time

The time=ms parameter indicates the number of milliseconds into the timeline at which to start playing. When the HTTP or HTTPS directive also contains a plref=ID value, the time parameter indicates the amount of time into the specified chapter to start playback. If the directive does not contain a plref=ID value, the time parameter sets an absolute time from the start of the playlist.

**Tip:** The playlist clipBegin attribute can set a clip’s start position at somewhere other than the clip’s normal beginning. In this case, the clipBegin value in the
playlist and the time offset in the HTTP command are cumulative. See “Defining the Clip Beginning” on page 49.

Rate Control Flag

For the rate control flag, the default value is mdp=1. This allows server-side rate control to modify the streaming rate and accommodate fluctuating network conditions. The value mdp=0 prevents server-side rate shifting. This means that Helix Server delivers each clip or broadcast at a single streaming rate unless the media player uses a client-side rate control method such as SureStream.

Note: Once the session starts, the rate control state (enabled or disabled) stays in effect for the entire RTSP session.

For More Information: Server-side rate control must be enabled and configured for each type of client. For details about rate control, refer to the Helix Administrator online help.

Playlist Format

The Helix Server playlist format utilizes SMIL 2.0, a case-sensitive, XML-based language for creating multimedia presentations. Typically, a media player interprets the SMIL mark-up. With playlists, however, only Helix Server reads the playlist and interprets the mark-up. This makes playlists compatible with media players that do not support the SMIL 2.0 standard.

A playlist file uses an .hpl file extension. It must contain a <body> section, and it typically includes the optional <head> section. The opening <smil> tag must declare the SMIL 2.0 namespace, as shown in the following example:

```xml
<smil xmlns="http://www.w3.org/2001/SMIL20/Language">
  <head>
    ...header section for defining metadata...
  </head>
  <body>
    ...body section for defining clips...
  </body>
</smil>
```

Tip: If the .hpl file extension causes an error for a media player, you can use the extension .hpl.rm, hpl.mp4, or .hpl.3gp.

Note: Playlists support the use of HTML-style comments within the file. For example: <!-- This is a comment -->


SMIL Timing Values

Playlist attributes that specify relative timing values use the following syntax:

hh:mm:ss.xy
where:

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>hh</td>
<td>hours</td>
</tr>
<tr>
<td>mm</td>
<td>minutes</td>
</tr>
<tr>
<td>ss</td>
<td>seconds</td>
</tr>
<tr>
<td>x</td>
<td>tenths of seconds</td>
</tr>
<tr>
<td>y</td>
<td>hundredths of seconds</td>
</tr>
</tbody>
</table>

Only the ss field is required. When the time value does not include a decimal point, the last field is read as the seconds. For example, 1:30 means 1 minute and 30 seconds whereas 1:30:00 means 1 hour and 30 minutes. Note that all of the following values are equivalent to 90 minutes:

- `clipBegin="1:30:00.0"`
- `clipBegin="90:00"`
- `clipBegin="5400"`

### Setting Metadata Values

The playlist `<head>` section defines overall metadata for the presentation:

```xml
<smil xmlns="http://www.w3.org/2001/SMIL20/Language">
  <head>
    <meta name="Expires" content="time"/>
    <meta name="title" content="title"/>
    <meta name="author" content="author"/>
    <meta name="copyright" content="copyright"/>
    <meta name="chapter-skip" content="0|1"/>
  </head>
  <body>...body section for defining clips...</body>
</smil>
```

#### Presentation Title, Author, and Copyright

Most RTSP media players display metadata values in the user interface. The values are ignored by HLS clients. The metadata attributes of title, author, and copyright allow you to assign title, author, and copyright values to the entire playlist. For example:

```xml
<meta name="title" content="Year's Best Music"/>
<meta name="author" content="Exciting Media, Inc."/>
<meta name="copyright" content="(c)2012"/>
```

#### Playlist Expiration

The `name="Expires"` element functions like an HTTP header that provides an absolute time after which the playlist is no longer valid. To set an expiration time, add the following metadata tag to the playlist:

```xml
<meta name="Expires" content="time"/>
```
The time format is `dd-Mon-yyyy hh:mm:ss`. All time fields are required:

<table>
<thead>
<tr>
<th>dd</th>
<th>day numeric value, as in 04</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mon</td>
<td>first three letters of the month, as in Sep</td>
</tr>
<tr>
<td>yyyy</td>
<td>year, as in 2012</td>
</tr>
<tr>
<td>hh</td>
<td>hour in 24-hour format, as in 14 for 2 p.m.</td>
</tr>
<tr>
<td>mm</td>
<td>minute (00 to 59)</td>
</tr>
<tr>
<td>ss</td>
<td>second (00 to 59)</td>
</tr>
</tbody>
</table>

For example, the following element causes the playlist to expire on July 24, 2012 at 4:56.13 p.m:

```xml
<meta name="Expires" content="24-Jul-2012 16:56:13" />
```

The `Expires` attribute is useful for time-sensitive material such as a news show that is updated hourly. When a media player requests a playlist, Helix Server verifies the expiration timestamp against its system clock. It returns a `404 Not Found` error if a media player requests an expired playlist.

**Note:** If a playlist expires while a media player is playing the content, Helix Server continues to stream the content until the session stops.

**Chapter Skipping**

Chapter skipping works with externally controlled playlists streamed to RTSP media players. If `chapter-skip` is set to 1, a user-issued skip command causes Helix Server to jump to the next defined chapter in the playlist:

```xml
<meta name="chapter-skip" content="1" />
```

**For More Information:** Chapters are groups of clips, as explained in the section “Setting Up Chapters” on page 51.

**Chapter Skipping Enabled Example**

As an example of chapter skipping, suppose that the playlist defines a sequence of three clips as Chapter 1. Three additional clips make up Chapter 2. To skip between chapters, the session viewer uses a browser-based interface that contacts Helix Server with a `seek=next` or `seek=prev` directive (see “Seek” on page 44).

If Helix Server receives a `seek=next` directive while it is streaming *any* of the three clips in Chapter 1, it skips to the beginning of Chapter 2 (the fourth clip). Conversely, if Helix Server receives a `seek=prev` directive while the viewer is watching *any* of the three clips in Chapter 2, Helix Server jumps back to the beginning of Chapter 2 or skips back to the start of Chapter 1.

**For More Information:** See “Seeking to the Previous Clip” on page 44. The section “Chapter Skipping Example” on page 40 illustrates the process of skipping chapters.

**Chapter Skipping Disabled Example**

The default value for `chapter-skip` is 0, which disables chapter skipping. In this case, if Helix Server receives a `seek=next` while streaming the second clip in Chapter 1, the skip takes the viewer to the start of the third clip in Chapter 1. In other words, skip requests step the viewer through the clips in the order in which they are listed in the playlist, not in the order of defined chapters.
Adding Clips to the Playlist

Within the playlist <body> section, individual <video> elements identify each clip or broadcast. If the <video> element contains only simple attributes, it can be a unary tag ending with a closing slash. For example:

```html
<video src="/media/clip_1.mp4" dur="20000"/>
```

You can also write the <video> element as a binary tag. This is required if the element uses complex attributes expressed as param values. In this case, the <video> element does not include a closing slash. Instead, a </video> tag follows the param values. For example:

```html
<video src="/media/clip_1.mp4" dur="20000">
  <param name="skip-point" value="sequence_6"/>
</video>
```

**Tip:** Because the <video> element generically identifies a media source rather than a media type, you can use it for audio clips as well. Helix Server identifies the actual media type using the file extension or MIME type of the media stream.

Clip Attribute Summary

The following table summarizes the attributes that you can add to a <video/> element. Only the src attribute is required.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
<th>Function</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>clipBegin</td>
<td>time</td>
<td>Sets the point where the clip starts playback. Can be used with all types of sessions and media clients.</td>
<td>page 49</td>
</tr>
<tr>
<td>dur</td>
<td>time</td>
<td>Determines how long the clip plays. Can be used with all types of media clients.</td>
<td>page 49</td>
</tr>
<tr>
<td>id</td>
<td>string</td>
<td>Sets an ID used with skipping. This is useful only with externally controlled playlists.</td>
<td>page 51</td>
</tr>
<tr>
<td>skippable</td>
<td>0</td>
<td>1</td>
<td>Indicates if the viewer can seek through the clip or skip over the clip.</td>
</tr>
<tr>
<td>skip-point</td>
<td>ID</td>
<td>Sets a target for a seek-next action. This is supported only with externally controlled playlists.</td>
<td>page 50</td>
</tr>
<tr>
<td>src</td>
<td>path</td>
<td>Provides the relative path to the clip or playlist.</td>
<td>page 48</td>
</tr>
<tr>
<td>src-dur</td>
<td>time</td>
<td>Gives a clip duration for playlist timing purposes.</td>
<td>page 49</td>
</tr>
</tbody>
</table>

Specifying the Media Source

The src attribute provides the relative URL to the location of the clip or live broadcast on Helix Server. The value starts with the content or broadcast mount point under which the media resides. For example, the following clip is stored under the Helix Server default mount point for content (/):

```html
<video src="/clip_1.mp4" .../>
```

The next clip is stored under a user-defined mount point named /media/:

```html
<video src="/media/clip_2.mp4" .../>
```
Defining the Clip Beginning

The optional clipBegin attribute defines the amount of time into the clip’s internal timeline at which playback begins. If you omit it, the clip starts to play at its encoded beginning. The following example starts the clip at its 20-second mark:

```html
<video...clipBegin="20"/>
```

A time value that falls outside of the clip’s timeline is ignored. For example, setting clipBegin="60" on a clip that lasts 30 seconds has no effect. The clipBegin attribute is ignored for live broadcasts.

**Note:** If the clipBegin value used with a video clip does not fall on a keyframe, Helix Server starts playback at the preceding keyframe. If the first preceding keyframe occurs five seconds before the clipBegin value, for example, the actual clip timeline will be five seconds longer than calculated by Helix Server and will be reported to the media player when the playlist initializes.

Defining the Clip Duration

The optional dur attribute controls the amount of time that a clip plays once it starts to stream. The following example ends the clip after 85 seconds, regardless of the length of the clip’s internal timeline:

```html
<video...dur="85"/>
```

If you omit the dur value, the clip plays to its encoded end point. If the specified duration exceeds the actual clip length, Helix Server ignores the dur value. The dur attribute has no effect on live broadcasts.

Helix Server takes the clipBegin value into account when calculating the clip’s end point. For example, if the `<video/>` element specifies both clipBegin="20" and dur="40", the clip stops playing at the 60-second mark of its internal timeline. When it stops, it will have played for 40 seconds.

Specifying the Clip Duration for Timing Purposes

The optional src-dur attribute indicates the total length of the clip. Helix Server uses this value only to calculate playlist lengths, not to shorten a clip’s playback duration. If a dur attribute is also present in the video tag, Helix Server uses the dur value and ignores the src-dur value. You specify src-dur as a param element embedded in a binary `<video>` element:

```html
<video...>
  <param name="src-dur" value="122.3"/>
</video>
```

**Warning!** The src-dur value should always reflect the actual clip length to an accuracy of at least one-tenth of a second. Because Helix Server uses src-dur values to calculate the overall playlist duration, inaccurate src-dur values may lower the quality of playback, adversely affect the timing of seek actions, or cause a media player to end the presentation before all media sources have fully played.
Clip Durations and Presentation Seeking

Helix Server can calculate an overall presentation timeline only if each clip contains either a `dur` or a `src-dur` value. If an on-demand clip lacks either value, Helix Server streams the entire playlist as if it were a live broadcast. This has the following ramifications:

- In a noncontrolled playlist session, the viewer cannot fast-forward or rewind through the playlist.

  **Note:** An internally controlled session automatically fails over to a noncontrolled session when clip durations are not defined.

- In an externally controlled playlist session, an HTTP or HTTPS directive using the `time` parameter is ignored (see “Time” on page 44). However, directives to seek to other chapters within the playlist are honored.

Setting the Skip Point

The `skip-point` element lists as its value an existing chapter ID in the primary playlist or any secondary playlists. If the user issues a `skip=next` directive, Helix Server skips to the `skip-point` target, rather than the next chapter. If the `skip-point` does not exist in the playlist, Helix Server skips to the next clip or chapter in the defined sequence.

You specify `skip-point` as a `param` element embedded in a binary `<video>` element:

```
<video...>
  <param name="skip-point" value="sequence_6"/>
</video>
```

**Note:** The `skip-point` value is ignored if `chapter-skip` is enabled in the playlist metadata (see “Chapter Skipping” on page 47), or if the `skippable` value for the current clip is false.

**For More Information:** See “Defining a Chapter or Clip ID” on page 51.

Making a Clip Non-Skippable

The `skippable` parameter with the default value of 1 allows the user to skip the clip or seek through it. You can disable skipping or seeking for a clip by setting the `skippable` attribute to 0 within a `param` element embedded in a binary `<video>` element:

```
<video...>
  <param name="skippable" value="0"/>
</video>
```

The configuration variable `AllowInbandforNonskippablePL` determines if seeking is available in an internally controlled playlist that contains a non-skippable clip. By default, making a clip non-skippable automatically turns an internally controlled playlist into a noncontrolled playlist. That is, it disables seeking, making the entire playlist appear to be a live broadcast.

**For More Information:** See the Helix Administrator online help topic `Configuration File > Content Management Configuration > Server-Side Playlist Configuration`. 
Setting Up Chapters

An externally controlled session plays media resources in the sequence defined by the playlist. Within the overall playlist sequence, however, you use the SMIL `<seq>` element to combine clips into chapters. Consider the following example:

```xml
...earlier chapters...
  <seq id="chapter_3">
    <video id="clip_8" src="clip_8.mp4"/>
    <video id="clip_9" src="clip_9.mp4"/>
    <video id="clip_10" src="clip_10.mp4"/>
  </seq>
...later chapters...
```

Because the three clips are enclosed within `<seq>` and `</seq>` tags, the media player treats the clips as a single chapter. Skipping to the next chapter (if allowed) takes the viewer to the clip or sequence that starts after `chapter_3`.

Tip: If a clip is not contained with a `<seq>` group, the clip functions as a single chapter. For example, if a playlist has no defined chapters or `<seq>` groups, each `skip=next` directive takes the viewer to the next clip defined in the playlist.

Defining a Chapter or Clip ID

The optional `id` attribute identifies a chapter or any target for a skip directive. You can add an `id` value to a `<seq>` tag or to individual `<video/>` or `<ref/>` tags (see “Specifying the Media Source” on page 48). The ID provides a unique, user-defined name:

```xml
<video id="clip5" .../>
```

The ID is used with `plref=ID` directives (see “Playlist Reference” on page 44), as well as to define `skip-point` locations (see “Setting the Skip Point” on page 50) in externally controlled playlist sessions. The following rules apply to the use of IDs:

- Each ID must be unique within a single playlist.
- If you use two or more words for an ID, combine the words, or separate the words with an underscore or hyphen, as in `clip1`, `clip-1`, or `clip_1`.
- The first character for an ID can be a letter, a colon, or an underscore. It cannot be a number or a special character such as an ampersand (&). You can use numbers and special characters after the first character, however. For example, you can use `id="video3"` as an ID but not `id="3video"`.
- There is no minimum or maximum length for IDs.
- IDs are case-sensitive.

Streaming a Secondary Playlist

A playlist can refer to another playlist. When Helix Server reaches the reference to the secondary playlist, it begins to stream that playlist’s media resources. When that playlist finishes, Helix Server returns to the original playlist.
To set up a secondary playlist, add a `<ref>` element to the primary playlist. This element functions like a source clip `<video>` element. For the `<ref>` element’s `src` attribute, indicate the secondary playlist located on Helix Server:

```html
<ref src="/playlists/playlist2.hpl"/>
```

### Notes on Streaming Secondary Playlists

Note the following about secondary playlists:

- Secondary playlists function only with RTSP media clients. Do not include them in playlists used with HLS clients.
- Clips in all secondary playlists must be compatible with the first clip streamed from the primary playlist.

**For More Information:** See “Stream Requirements for RTSP Players” on page 32.

- Most metadata values listed in the `<head>` section of a secondary playlist are ignored. Metadata values for `title`, `author`, and `copyright` in the first playlist stay in effect through the presentation.
- The `chapter-skip` value of the original playlist is observed throughout the secondary playlist session.
- Helix Server honors the `Expires` date for a secondary playlist. If the expiration time has passed, Helix Server ignores the playlist, issues an error message, and continues to the next clip in the primary playlist.

**For More Information:** See “Playlist Expiration” on page 46.

- A `skip-point` in the primary playlist can target a chapter in a secondary playlist. As well, a `plref` directive can skip to a chapter in the secondary playlist from the primary playlist. For these functions to work, all chapters within the primary and secondary playlists must have unique IDs.

**For More Information:** For background on these functions, refer to “Setting Up Chapters” on page 51, “Setting the Skip Point” on page 50, and “Playlist Reference” on page 44.

### Playlist Examples

The following sections provide examples of playlist features.

#### Simple Playlist

The following is a simple playlist of three clips that play in sequence:

```html
<smil xmlns="http://www.w3.org/2001/SMIL20/Language">
  <head>
    <meta name="title" content="Coming Attractions"/>
    <meta name="author" content="Exciting Media, Inc."/>
    <meta name="copyright" content="(c)2012"/>
  </head>
  <body>
    <video id="clip_1" src="/media/clip1.mp4">
```
Note the following about this playlist:

- The body consists of a sequence of clips without a `<seq>` grouping. In this case, each clip functions as a separate chapter.
- Each clip includes an `id` value, which allows the viewer to skip to different clips using an HTTP or HTTPS request in an externally controlled playlist session.
- All clips include `src-dur` values that indicate the clip's playing time. This allows Helix Server to calculate the session's total playing time, as well as to seek forward or backward across the three clips in either an externally controlled or internally controlled playlist session.

Advertisements Preceding Video Clips

The following sample playlist contains three video clips. Each clip is preceded by a short advertisement. In the header, the `chapter-skip` attribute is enabled. In this case, a user-initiated `skip` command takes the viewer to the next defined sequence. This effectively groups each ad clip with its subsequent content clip:

```xml
<smil xmlns="http://www.w3.org/2001/SMIL20/Language">
  <head>
    <meta name="title" content="Coming Attractions"/>
    <meta name="chapter-skip" content="1"/>
  </head>
  <body>
    <seq id="chapter_1">
      <video id="ad_1" src="/ads/ad1.mp4">
        <param name="src-dur" value="31.3"/>
      </video>
      <video id="preview_1" src="/media/preview1.mp4">
        <param name="src-dur" value="185.4"/>
      </video>
    </seq>
    <seq id="chapter_2">
      <video id="ad_2" src="/ads/ad2.mp4">
        <param name="src-dur" value="30.8"/>
      </video>
      <video id="preview_2" src="/media/preview2.mp4">
        <param name="src-dur" value="201.4"/>
      </video>
    </seq>
    <seq id="chapter_3">
      <video id="ad_3" src="/ads/ad3.mp4">
        <param name="src-dur" value="29.9"/>
      </video>
    </seq>
  </body>
</smil>
```
Uploading Playlists

After the Web portal creates a playlist, it uploads it to Helix Server. If the portal has write access to the Helix Server file system, it can copy playlists to the proper directory. Otherwise, the portal uses the HTTP methods described in the following sections.

**Note:** For the Web portal to upload a playlist, a mount point must have a Yes value set for its Allow File Creation option. A delete action requires a Yes value for the Allow File Deletion option. For an update action, both permissions must be set to Yes. For details, refer to the Helix Server online help topic Helix Administrator > Server Setup > Mount Points > Creating an On-Demand Mount Point.

Adding a Playlist

To upload a playlist, the Web portal issues an HTTP or HTTPS XML POST command directed at the Helix Server file system control port. The request URL includes an add command along with a path query string parameter that indicates the intended mount point and playlist name. For example:

```
http://helixserver.example.com:8010/add?path=playlists/songs.hpl
```

The preceding example writes the posted playlist data to the file `songs.hpl` under the Helix Server mount point `/playlists/`. Note that a forward slash preceding the mount point name is not used in the actual command.

**Tip:** The path can also include subdirectories under the mount point, such as `playlists/music/songs.hpl`. If the specified subdirectory does not exist already, Helix Server creates it, assigning it the same permissions as the parent directory.

**Warning!** Do not enclose the mount point and playlist string in single or double quotation marks. If the string includes spaces, URL-encode each space character as `%20`.

**For More Information:** If authentication is enabled, Helix Server prompts for a user name and password. For more about authentication, refer to “Secure the Playlist Management System” on page 58.

Updating a Playlist

To update a playlist, the Web portal adds a new version of the file to the same playlist mount point, overwriting the existing playlist. If viewers are playing the older playlist when the newer
playlist arrives, they continue to receive the older content until their session finishes. Requesting the playlist again provides the updated playlist.

**Warning!** The Web portal must ensure unique naming of playlists to avoid inadvertently overwriting one playlist with another.

### Deleting a Playlist

To delete a playlist, use an HTTP or HTTPS GET command directed at the file system control port. The URL includes a delete command and a path query string parameter:

http://helixserver.example.com:8010/delete?path=playlist/songs.hpl

### Logging and Error Codes

The following sections explain how Helix Server logs clips streamed during a playlist session, as well as how it reports HTTP errors to the Web portal or user browser.

#### Logging Styles and Variables

Helix Server adds a log record for each clip or broadcast when the content streams. To record information about playlist sessions, set the logging style to 8. This logging style includes three fields that contain playlist information for each logged clip. These fields provide the guid value, list the order that clips played, and indicate the last clip in the playlist.

**Custom Logging Variables**

You can also define custom logging reports. The following are the Helix Server registry variables related to playlist logging:

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>%Client.<em>.Session.</em>.SessionIDName%</td>
<td>Session ID assigned by the guid parameter. See “Session ID” on page 43.</td>
</tr>
<tr>
<td>%Client.<em>.Session.</em>.Clip.*.ClipCount%</td>
<td>Counter that starts at 1 and increments by 1 for each successive clip.</td>
</tr>
<tr>
<td>%Client.<em>.Session.</em>.Clip.*.FinalClip%</td>
<td>Counter set to 0 for all clips in the playlist, except the final clip, which has a value of 1.</td>
</tr>
</tbody>
</table>

**For More Information:** Refer to the Helix Administrator online help topic **Registry Properties > Client Session and Clip Properties** for details about logging variables.

#### Error Conditions

Helix Server does not verify the contents of a playlist before streaming the resources. If a clip is incompatible with the preceding clip in a playlist streamed to an RTSP player, Helix Server omits that clip and streams the subsequent clip. Any of the following changes in format or encoding may cause a clip to be omitted:

- Switching from one file format to another, such as from MPEG-4 (.mp4) to RealMedia (.rm).
- Changes in the codec used to encode the streams, such as from H.264 to H.263.
- Switching the media bit rate when streaming to RTSP players. For multi-rate clips, all bit rate encodings must be identical to support playback with RTSP players.
- Changes in maximum video frame rate or audio sampling rate when streaming to RTSP players.
- Switching from the RDT packet format to RTP (or vice versa) when streaming to RealNetworks media players.
- Streaming content located on a different server.

Timeline Error from Omitted Clips
In an internally controlled playlist, an omitted clip can result in a timeline error. With this type of playlist, Helix Server adds the clip duration values to calculate an overall playback time that it sends to the media player. If Helix Server must omit one or more clips, the actual playback time will be shorter than the total playback time reported to the media player.

**Note:** Some media players do not tear down the RTSP session once the last clip has played. Instead, these players wait for the advertised playback time to elapse, or for the viewer to stop the stream. This results in blank playback equal to the playing time of the omitted clips.

HTTP Status Codes
The following are the HTTP status codes that Helix Server may return for a playlist request.

200 OK
Action successful.

204 No Content
Playlist switch request, such as a chapter skip, was successful.

273 No Session
Requested change to the playlist session failed because the session identified by the guid value is no longer active. See “Session ID” on page 43.

274 Content Not Compatible
Clip is not compatible with previous clips in the playlist.

403 Forbidden
1. Requested playlist session uses a guid value already in use. See “Session ID” on page 43.
2. Request to add or delete the playlist failed because of one of the following reasons:
   - Helix Server file permissions do not allow the requested action.
   - Control port value in the URL is incorrect.
   - The Allow File Creation or Allow File Deletion option is set to False for the playlist mount point.
404 Not Found

1. Requested playlist does not exist under the specified mount point. See “Adding a Playlist” on page 54.
2. Next clip in the playlist was not found.
3. Playlist expiration time has been exceeded. See “Playlist Expiration” on page 46.
4. Request to modify the playlist session failed because the a guid value could not be found. See “Session ID” on page 43.

Configuring Helix Server for Playlist Control

The following sections explain the components of Helix Server that must be configured to enable server-side playlist management.

Set Playlist HTTP Ports

The server-side playlist system uses two Helix Server ports for HTTP or HTTPS communications with the Web portal. These ports are assigned during system installation. Navigate to the Ports page in Helix Administrator (Server Setup > Ports) to ensure that the ports are set to the values you want.

Tip: The setting for the Enable Control Port Security field on the Helix Administrator ports page determines if only HTTP (security disabled) or HTTPS (security enabled) requests are accepted on the control port.

Note: If a firewall separates the Web portal and Helix Server, ensure that the firewall allows HTTP or HTTPS traffic on these ports.

For More Information: See Helix Media Delivery Platform Quick Start Guide for more about these port values. For instructions about changing port values, refer to the Helix Administrator online help topic Helix Administrator > Server Setup > Ports > Port Assignments. The online help also explains how to set up access rules that limit the use of certain ports to specific IP addresses.

Define Playlist Mount Points

RealNetworks recommends that you store playlists under a dedicated Helix Server mount point that does not store content clips. In Helix Administrator (Server Setup > Mount Points), create a mount point such as /playlists/. Create one or more playlist mount points depending on your needs.

Note: For each playlist mount point, set Yes values for the Allow File Creation and Allow File Deletion options. This allows the Web portal to write and delete playlists as described in the section “Uploading Playlists” on page 54.

Tip: The base path that corresponds to a mount point may have subdirectories. You can create subdirectories under the mount point’s base path directly. Or, the Web portal can create a subdirectory when it uploads a playlist.
Secure the Playlist Management System

User name and password authentication is used with Web portals that add or delete playlists to Helix Server mount points. The section “Uploading Playlists” on page 54 explains the process for adding or deleting a playlist.

**Warning!** Failure to secure playlist mount points may allow outside processes to add content to, and delete content from, any unsecured Helix Server mount point.

**Playlist Authentication**

The following procedure summarizes the process for setting up authentication for playlist addition and deletion commands.

**To implement authentication for playlist management:**

1. For the authentication realm (**Security > Authentication**), select the predefined realm SecurePlaylistManager. Then indicate if the Web portal credentials will be submitted using the Basic or Digest authentication protocol.

   **Warning!** Using a realm other than SecurePlaylistManager causes playlist authentication to fail. You can select a different realm only if you manually update the FileSystemControl list in the Helix Server configuration file. For details, refer to the Helix Administrator online help topic **Configuration File > Content Management Configuration > Server-Side Playlist Configuration**.

   For More Information: For background on realms as well as the Basic and Digest protocols, refer to the Helix Administrator online help topic **Helix Administrator > Security > Authentication**.

2. Under the SecurePlaylistManager realm, click **Add a User to Realm**. In the pop-up dialog, enter the user name and password that the Web portal must submit when adding or deleting a playlist.

   **Tip:** The user name and password used to access Helix Administrator is automatically added to this realm when the system is installed.

**User Authentication**

Securing playlist mount points against the SecurePlaylistManager realm causes Helix Server to authenticate only playlist additions and deletions. When a user requests a playlist stored under the playlist mount point, Helix Server does not require a user name and password, however. You can implement user name and password authentication for desktop media players requesting playlist content. To do so, place the playlist clips under a secure content mount point (such as the default security mount point, /secure/). Then, set up user names and passwords for individual users within a content authentication realm, such as SecureContent.

**Warning!** When Helix Server authenticates a playlist request, it validates only the request for the first clip in the playlist. It does not require authentication for subsequent clips in the playlist. If the first clip does not reside under a secure mount point, no authentication occurs for any content in the playlist.
Enable HLS Segmentation

HLS segmentation must be enabled to support playlist management for HLS clients. All content included in playlists must fall under a mount point enabled for segmentation, such as /Segments/. You must also enable segmentation for applicable broadcasts, such as those that are streamed under the default /broadcast/ mount point.

Enable Playlist Control Requests

Once you have set up and secured the necessary mount points and HTTP ports, you must enable Helix Server to listen for playlist control requests. In Helix Administrator, navigate to Content Management > Serverside Playlist. In the drop-down box for Enable Serverside Playlist, select Yes and click Apply.

Warning! Setting this option to No prevents Helix Server from honoring requests to skip to different parts of the playlist. However, it does not prevent users from requesting and receiving playlist content.
Chapter 5: CHANNEL SWITCHING

Channel switching allows Helix Server to switch the media input for a continuous stream, eliminating the need to set up a separate RTSP session for each stream. Channel switching works with on-demand clips, live streams, and simulated-live streams broadcast by SLTA.

Note: Fast-channel switching requires the use of RTSP. HLS, DASH, and Flash clients are not supported for fast-channel switching.

Understanding Channel Switching

A channel is a stream that can be switched to a different media input (a new channel). This switching capability eliminates the need to set up and tear down a user’s RTSP session when the user selects a new stream. As a result, a user can “change channels” without experiencing disruption in playback when a new channel begins. To the user, the succession of streams appears to be a broadcast of user-selected content that ends when a channel finishes streaming and no subsequent channel has been queued.

The following figure illustrates a simple channel switching scenario. Here, the user selects the first channel, “Pop Hits.” The user does not make this choice directly through Helix Server in an RTSP session. Rather, the user interacts with a third-party Web portal that supplies the user’s browser with links to the available channels.

Once the user selects the channel, the Web portal sends the channel choice information to Helix Server as a specially-formatted HTTP or HTTPS URL directed toward a secure port. Helix Server then delivers the media player instructions in an SDP file that enable it to connect to Helix Server and receive the channel. The channel content may be an on-demand clip, a live broadcast, or a simulated-live broadcast.

First Channel Playback

The next figure shows a user changing the streaming content to a second channel, “Jazz.” Again, the user interacts with the Web portal to do this. In this scenario, the change request
results from user feedback, such as clicking a link in an on-line guide. However, change requests can occur without user interaction. For example, change requests may result from a predefined playlist that the Web portal manages. Once the Web portal sends Helix Server an HTTP or HTTPS command to change the channel, Helix Server queues the new source. It can switch to the new source immediately, or after the current source has finished playing. When the new channel starts, it replaces the existing stream within the media player’s RTSP session, providing a seamless switchover between the channel streams.

Second Channel Playback

Third-Party Components

The following sections describe the components that are necessary for channel switching but that are not part of Helix Server. Each section provides details about the requirements that the component must meet to enable channel switching.

Web Portal

A Web portal is a third-party application that provides the user with information about available channels. The portal carries out any billing duties, and submits channel requests to Helix Server according to the workflow described in the section “Channel Switching Workflows” on page 67.

User Browser

Each user must have available a browser that allows the user to select channels by interacting with the Web portal. There are no specific requirements for the browser required by the channel switching system.

Note: A rich media client may fulfill browser functions depending on the player’s capabilities and the means used to deliver channel information.

User Media Players

The user must have an RTSP-based media player that supports the codec used to encode the stream. Because non-standard RTSP commands are not used, most media players compliant with the RTSP standard can participate in channel switching.
Codexs and File Formats

Because a media player cannot switch its decoding method within a media stream, all channel streams must be encoded using the same streaming rates and the same codec (same profile and level).

Tip: Because stream quality can differ across encoders, creating content using the same hardware or software encoder helps to prevent streaming errors.

Note: Clips used for channel switching are subject to the maximum size imposed by 32-bit operating systems. On 64-bit operating systems, files can be any size.

Supported Codecs

Channel switching works with the following codecs:

Video Codecs
- H.264 (recommended)
- H.263
- MPEG-4
- RealVideo

Audio Codecs
- AAC or AAC+ (recommended)
- Enhanced AAC+
- AMR-NB or AMR-WB
- RealAudio

Supported File Formats

Supported file formats are the following:
- MPEG-4 (.mp4 and variants)
- 3GPP (.3gp)
- RealMedia (.ra, .rv, .rm)
- QuickTime (.mov)
- F4V (.f4v)

Note: Channel switching does not function with Flash clients, which use the RTMP protocol. However, you can use the F4V format with RTSP clients.

Unsupported Formats

The following formats are not supported:
- Windows Media
- FLV
- any format that use digital rights management (DRM) protection
Protocols and Transports

Channel switching works only with RTSP-based streams. This includes RTSP streams cloaked as HTTP. MMS streaming, RTMP streaming, and HTTP download are not supported. The network transport can be either UDP or TCP. You can use the RealNetworks RDT data packet format or the standards-based RTP format.

User Authentication

Channel switching is compatible with security features such as username and password authentication. When Helix Server authenticates a user, it validates only the initial RTSP request made by the media player. HTTP or HTTPS commands requesting a switch to a different channel are not authenticated by Helix Server. It is the responsibility of the Web portal to verify that channel switch requests made through the portal have come from the actual users.

Note: User name and password authentication is not viable in all situations, especially for mobile players that do not pass usernames and passwords. Contact your RealNetworks representative for information about additional ways to generate and authenticate secure URLs.

Error Conditions

Attempting any of the following changes when switching from one channel to another causes an error condition that closes the channel:

- Switching from one file format to another, such as from RealMedia (.rm) to MPEG-4 (.mp4).
- Change in the codec used to encode the channel, such as from H.263 to H.264.
- Switching the streaming bit rate. For multi-rate clips, all bit rate encodings must be identical.
- Switching from the RDT packet format to RTP, or vice versa.
- Streaming content located on different servers.
- Switching to a non-existing source.

Tip: If an error occurs, Helix Server can still stream the new channel (see the section “SDP File on Error” on page 67). However, this requires tearing down the user’s existing RTSP session and setting up a new one, thereby eliminating the seamless nature of channel switching.

For More Information: For more about error messages, refer to “HTTP Status Codes” on page 67.
Compatibility with Other Features

The following table summarizes the compatibility of channel switching with other Helix Server features.

<table>
<thead>
<tr>
<th>Other Feature</th>
<th>Channel Switching Used with Other Features</th>
</tr>
</thead>
<tbody>
<tr>
<td>reduced start-up delay</td>
<td>Reduced start-up delay is compatible with channel switching.</td>
</tr>
<tr>
<td>splitting</td>
<td>Channel switching is compatible with splitting live streams, which Chapter 3 describes. However, there will</td>
</tr>
<tr>
<td></td>
<td>be a delay if a channel switch request is for a pull-split stream that has not yet been pulled on the receiver</td>
</tr>
<tr>
<td></td>
<td>that manages the user’s RTSP session.</td>
</tr>
<tr>
<td>live rate adaptation</td>
<td>Live rate adaptation is compatible with channel switching, as long as all channels are encoded at the same</td>
</tr>
<tr>
<td></td>
<td>streaming rates.</td>
</tr>
<tr>
<td>basic logging</td>
<td>To record channel switching events in the access log file, use basic logging style 7.</td>
</tr>
<tr>
<td>advanced logging</td>
<td>You can capture channel switching events using the customized logging templates. Refer to the client</td>
</tr>
<tr>
<td></td>
<td>properties section of the Helix Administrator online help for information about the FinalSwitch, SessionControlID, and SwitchCount properties.</td>
</tr>
<tr>
<td>Helix Proxy</td>
<td>Helix Proxy delivers switchable streams only in pass-through mode. Because the proxy does not accept HTTP</td>
</tr>
<tr>
<td></td>
<td>requests, a Web portal must send all channel switch commands to the origin server.</td>
</tr>
<tr>
<td>HTTP proxies</td>
<td>Third-party HTTP proxies can be used to proxy the HTTP commands for the Web portal.</td>
</tr>
</tbody>
</table>

Channel Controller

A Web portal issues channel switch commands to the channel controller on Helix Server. Commands consist of HTTP or HTTPS requests directed to the channel controller port. They include parameter and value pairs separated from the main URL by standard query string syntax (“?” and “&”).

The parameter values instruct Helix Server on the channel switching actions to take. The following example illustrates the command syntax:

http://helixserver:controller_port/stream.ext?param1=value1&param2=value2…

Channel Switching Control Port

The channel controller listens for HTTP or HTTPS requests on a user-configured port. The control port is initially set to 8008, but you can change this to any free port using Helix Administrator (Server Setup > Ports). RealNetworks does not recommend using the main HTTP port (typically port 80) as the control port, however.

Tip: The setting for the Enable Control Port Security field on the Helix Administrator ports page determines if only HTTP (security disabled) or HTTPS (security enabled) requests are accepted on the control port.
Control Port Security

To prevent unauthorized channel switching, ensure that only authorized Web portals and HTTP proxies can issue HTTP commands to Helix Server on its control port. Media users should not have direct access to the control port.

**Tip:** You can restrict port access by configuring your firewall appropriately or by defining an access rule. The Helix Server online help section Helix Administrator > Security > Access Control describes access rules.

HTTP Request Parameters

The following table summarizes the parameters included in an HTTP or HTTPS GET request directed toward the Helix Server controller port. The last two columns indicate if the attribute is required in the initial request that sets up the channel session, as well as a channel switch request.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Function</th>
<th>Example</th>
<th>Initial Request</th>
<th>Channel Switch</th>
</tr>
</thead>
<tbody>
<tr>
<td>Switching Allowed Marker</td>
<td>Indicates that the requested stream is a channel that can be switched.</td>
<td>hxs=1</td>
<td>required</td>
<td>required</td>
</tr>
<tr>
<td>Session ID</td>
<td>Identifies the user’s RTSP session.</td>
<td>guid=ID</td>
<td>required</td>
<td>required</td>
</tr>
<tr>
<td>Rate Control Flag</td>
<td>Enables rate control: 0 – rate control disabled (default) 1 – rate control enabled</td>
<td>mdp=1</td>
<td>optional</td>
<td>not used</td>
</tr>
<tr>
<td>Queueing Indicator</td>
<td>Specifies when the next channel starts to play: 0 – switch immediately (default) 1 – switch at end of playback</td>
<td>queue=1</td>
<td>not used</td>
<td>optional</td>
</tr>
<tr>
<td>SDP File on Error</td>
<td>Generates an SDP file on an error: 0 – no new SDP on error 1 – new SDP on error (default)</td>
<td>sdp=0</td>
<td>optional</td>
<td>optional</td>
</tr>
</tbody>
</table>

**Note:** The order of query string parameters in a controller URL does not matter.

**Tip:** You can change the query string parameter names, such as hxs, through the Helix Server configuration file. See the Helix Administrator online help topic Configuration File > Content Management Configuration > Channel Switching Configuration.

Switching Allowed Marker

The switching marker is the query string parameter hxs. The value for this parameter is 0 (stream is not switchable) or 1 (stream is switchable). The parameter and value hxs=1 must be included in each request URL to instruct Helix Server to allow the stream in a channel switching session.
Note: The presence of this query string parameter in the initial channel request causes Helix Server to generate an SDP file automatically. You therefore do not need to include the SDPgen mount point in the request URL.

Session ID
The session ID name is the value of the guid query string parameter. Helix Server identifies the stream-switching session using this parameter’s value. The Web portal is responsible for generating a unique value for each user, and including the parameter and value pair in each request to switch a user’s channel. The value should be 8 to 32 characters in length, and may contain any of the following characters:

• a-z
• A-Z
• 0-9
• - (hyphen)

Rate Control Flag
The rate control flag is the query string parameter mdp. The default value of mdp=0 prevents server-side rate shifting. This means that, unless the media player uses a client-side rate control method, Helix Server delivers the channel streams at a single streaming rate. Including the parameter mdp=1 in the initial channel request enables Helix Server to use its server-side rate control feature.

Note: To enable rate control, add the mdp parameter to the request for the first channel. Once rate control is enabled or disabled, it stays in effect for the entire RTSP session. Adding this parameter to a request to switch to a subsequent channel has no effect.

For More Information: For server-side rate control to function, the feature must be enabled and configured for the client. For details about rate control, refer to the Helix Administrator online help.

Queuing Indicator
Adding the parameter queue=1 to the query string informs Helix Server to queue the new channel but not switch to the new channel until the current channel has ended. The default value of queue=0 stops the current channel and switches to the new channel immediately.

Note: Use the queue=1 parameter and value only when the current and requested channels have defined endpoints. This includes on-demand clips and finite SLTA playlists. For broadcasts and looping SLTA playlists, the channel switch must occur immediately. Otherwise, it may not occur at all.

Tip: If a channel change is pending, and the user issues another channel change request, the new request overrides the queued request. This means that a user can have only one pending request at a time.
**SDP File on Error**

The default value of `sdp=1` causes Helix Server to return an SDP file for the requested channel if the channel switch request fails. If you add `sdp=0` to the query string, Helix Server returns only an HTTP code if the channel switch fails.

As an example of channel switch failure, suppose that the requested channel is encoded in a media format that differs from the current channel. The request causes an error that tears down the user’s RTSP session. However, if Helix Server returns a new SDP file to the Web portal, the portal can forward the SDP file to the user’s media player. This sets up a new channel session.

**HTTP Status Codes**

The following are the HTTP status codes that Helix Server may return for an HTTP request sent to its session controller port.

<table>
<thead>
<tr>
<th>Code</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>204</td>
<td>No Content Initial SDP request or a channel switch request succeeded.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found Switch request failed because the hxs or guid parameter was not included.</td>
</tr>
<tr>
<td>273</td>
<td>No Session Switch request failed because the RTSP channel session has already been closed, or the requested content does not exist.</td>
</tr>
<tr>
<td>274</td>
<td>Content Not Compatible Switch request failed because the current channel stream and requested channel stream are incompatible. (See “Error Conditions” on page 63.)</td>
</tr>
</tbody>
</table>

**Channel Switching Workflows**

The following sections explain how the channel switching system carries out its essential tasks.

**Initial Channel Playback**

This section explains the sequence of events that occurs when a user initiates a new channel session. The following figure illustrates the process described in the subsequent steps.

**Request First Channel**
To start a new session by requesting a channel:

1. The user selects a channel using information supplied by the Web portal. The channel information may be delivered to the Web browser or to a rich media client.

2. The Web portal requests a channel SDP file from Helix Server using an HTTP or HTTPS GET request directed toward the server’s channel controller port. To the base URL, the Web portal adds the switching allowed and session ID parameters. For example, the URL to an on-demand channel that the Web portal requests from Helix Server might look like the following:

   http://helixserver.example.com:8008/station5.mp4?hxs=1&mdp=1&guid=a5230cj321

3. Helix Server checks for an active channel that uses the submitted session ID value. If no such channel exists, it returns an SDP file for the requested channel to the Web portal. That SDP file encapsulates an RTSP channel URL that includes the same query string parameters and values that the Web portal submitted in the request URL.

4. The Web portal delivers the SDP file to the user, identifying the stream with the SDP MIME type, application/sdp.

   Note: The Web portal may pre-generate SDP files and link them to a Web page dynamically generated for the user. This means that the actions described in Step 2 occur before the user selects a channel. Once the user selects a channel as described in Step 1, the media player contacts Helix Server as described in Step 5.

5. The user’s media player initiates the RTSP session by contacting Helix Server using the channel URL.

6. As Helix Server begins to stream the channel, the channel controller registers the channel under the session ID supplied in the URL.

Switch to a New Channel

As the initial channel plays, the stream can be changed to a different channel. The following figure illustrates the process described in the subsequent steps.

Switch to New Channel
To switch to a new channel:

1. The user selects a new channel, typically by clicking on an HTTP link supplied by the Web portal. Alternatively, the channel change request may result from a playlist that the Web portal manages. In this case, this first step of user interaction is not required.

2. The Web portal directs an HTTP or HTTPS GET request to the Helix Server channel controller port. The URL does not request an SDP file. Rather, it requests a channel switch, and includes the hxs and guid query string parameters that were included in the initial channel request. To these parameters, the Web portal may add the queue or sdp parameter. For example, the URL request that the Web portal sends to Helix Server might look like the following:

   http://helixserver.example.com:8008/station9.mp4?hxs=1&queue=1&sdp=0&guid=a5230cgj321

3. When Helix Server receives the new channel URL, it does the following:
   • Checks that the session identified by the session ID value (guid value) is still active.
   • Verifies that the newly requested channel, whether an on-demand clip or a broadcast, exists.
   • Determines if the current channel and the new channel are compatible.
   If any of these verifications fail, Helix Server returns an HTTP error code and, optionally, an SDP file to the new channel.

4. Optionally, the Web portal delivers a response to the user. This might be a message indicating that the channel switch succeeded or failed. On a failure, the Web portal may simply deliver the SDP file for the new channel, if one was generated.

5. At the point indicated by the queue value (immediately, or after the current channel finishes), Helix Server switches the stream within the RTSP session to the new channel. The user’s media player begins to play the new channel stream as soon as all buffered data for the previous channel is consumed. The channel switch then occurs without any pause or rebuffering.

   Tip: The rate control feature allows you to specify the maximum amount of playback time (such as three seconds) the media player buffers for each stream.

End Session

If the current stream for a channel ends playback and Helix Server has not received a channel switch URL, Helix Server closes the channel.

To end a channel session:

1. Helix Server issues the media player an RTSP BYE packet.

2. The media player may then issue an RTSP TEARDOWN command and close its TCP connection.

3. Helix Server records the final stream in its access log entry with a final_switch value of 1.
**For More Information:** For more about how streams within a channel session are recorded in the access log, refer to the Helix Server online help topic *Helix Administrator > Logging and Monitoring > Basic Logging > Basic Access Log Fields.*

## Configuring Channel Switching

Do the following in Helix Administrator to enable channel switching.

- Click **Server Setup > Ports** and verify that the channel control port is set to the port value you wish to use.

  **Note:** If Enable Control Port Security is set to Yes, the controller port accepts only HTTPS connections.

- Click **Content Management > Channel Switching**, to enable channel switching and select the SDPgen mount point.

  **For More Information:** For configuration details, refer to the Helix Administrator online help topic *Helix Administrator > Content Management > Channel Switching.*
INDEX

A
ASXgen
NAT firewall workaround, 3
virtual IP addresses, 3

C
channel negotiation, 5
channel switching
browser requirements, 61
channel controller
HTTP requests, 65
listen port, 64
queue parameter, 66
security, 65
session ID, 66
switching allowed marker, 65
codec support, 62
configuration, 70
error conditions
causes of, 63
HTTP status codes, 67
SDP generation, 67
file format support, 62
media player requirements, 61
overview, 60
protocol support, 63
Web portal requirements, 61
workflows
initial channel playback, 67
select new channel, 68
session end, 69
conventions in this manual, 2

D
DES encryption for SNMP, 10

E
encoder firewall issues, 7

F
firewalls
media player issues, 5
NAT address specification, 3
proxies, 8
receivers, 7
server placement, 3
Windows Media Encoder, 7
Flash broadcast splitting, 24

H
Helix broadcast
splitting, 22
Helix Server

P
playlist format
chapter skipping
chapter-skip attribute, 47
example, 47
overview, 40
chapters, 51
clipBegin attribute, 49
comments, 45
dur attribute, 49
examples, 52
expiration time, 46
file extension
.hpl, 45
.hpl.3gp, 45
.hpl.mp4, 45
.hpl.rm, 45
playback error, 45
id attributes
case-sensitivity, 51
first characters, 51
length, 51
spaces in, 51
uniqueness, 51
media sources, 48
metadata
presentation, 46
secondary playlists, 51
seq attribute, 51
skippable attribute, 50
skip-point attribute, 50
SMIL background, 45
src-dur attribute, 49
timing values, 45
video source tags, 48
playlist management
authentication
playlist upload, 58
user access, 58
browser requirements, 32
codec support
HLS clients, 34
RTSP players, 32
deleting a playlist, 55
enabling, 59
error conditions
causes of, 55
effect on timeline, 56
HTTP status codes, 56
externally controlled session
chapter skipping, 40
overview, 31
parameters
guid, 43
hpl, 43
mdp, 45
plref, 44
seek, 44
time, 44
playlist selection, 38
rate control, 45
seeking
implementing, 42
seeking to previous clip, 44
session ID, 43
skipping, 41
workflows, 38
feature compatibility, 35
file format support
RTSP players, 32
HLS client requirements, 34
HLS clients
live broadcasts, 34
playlist naming, 35
segment creation, 35
segmentation configuration, 59
internally controlled session
overview, 31
seeking, 37
workflows, 36
logging
style, 55
variables, 55
mount point requirements, 57
noncontrolled session
overview, 31
playlist selection, 36
workflows, 36
overview, 30
protocol support, 33
RTSP media player requirements, 32
RTSP seeking, 37
security requirements, 58
updating a playlist, 54
uploading a playlist, 54
Web portal
requirements, 32
SDP modifications, 37, 40
ports
control channel, 64
hinting, 6
proxy firewall issues, 8
Q
QuickTime splitting, 23
R
Ramgen
NAT firewall workaround, 3
virtual IP addresses, 3
RealProducer
sending streams to multiple receivers, 22
splitting technology, 19
receiver firewall issues, 7
redundant encoders with splitting, 27
S
SDPgen
NAT firewall workaround, 3
virtual IP addresses, 3
server-side playlists, see playlist management
SHA for SNMP, 10
SNMP
AgentX protocol and port, 12
authentication, 10
DES encryption, 10
management system, 16
master agent
address and port, 12
configuration, 11
security model, 12
UNIX startup, 16
VACM setup, 13
Windows startup, 15
MD5, 10
MIB file, 10
overview, 9
plug-in configuration, 11
privacy, 10
security, 12
SHA, 10
traps, 11
trees
configuration, 17, 18
control, 18
monitor, 16
VACM setup, 13
version support, 10
splitting
  encoder-to-server, 19
  multicast distribution to players, 26
  multiple splitting arrangements, 25
  one-to-many, 21
  one-to-one, 22
  overview, 19
  pull splitting
    bandwidth efficiency, 21
    configuring, 23
    overview, 20
    with push encoding, 21
  push splitting
    configuring, 22
    overview, 20
    transmitter setup, 20
    with pull encoding, 20
  simultaneous unicast and multicast, 27, 29
  split stream direction, 25
  terminology definitions, 19
  transmitter redundancy, 28

T  terminology, 2
  transmitter
    pull splitting, 23
    push splitting, 20
    redundancy, 28
    typographical conventions, 2

U  UDP resend requests, 5

V  VIPs, 3
  virtual IP addressing, 4

W  Windows Media firewall issues, 7