Wowza Media Server® 3

Configuration Reference

Copyright © 2006 - 2014 Wowza Media Systems, LLC. All rights reserved.

Wowza Media Server 3: Configuration Reference

Version: 3.6.4

---

http://www.wowza.com

Copyright © 2006 - 2014 Wowza Media Systems, LLC. All rights reserved.

This document is for informational purposes only and in no way shall be interpreted or construed to create any warranties of any kind, either express or implied, regarding the information contained herein.

Third Party Information

This document contains links to third party websites that are not under the control of Wowza Media Systems, LLC ("Wowza") and Wowza is not responsible for the content on any linked site. If you access a third party website mentioned in this document, then you do so at your own risk. Wowza provides these links only as a convenience, and the inclusion of any link does not imply that Wowza endorses or accepts any responsibility for the content on third party sites.

This document refers to third party software that is not licensed, sold, distributed or otherwise endorsed by Wowza. Please ensure that any and all use of Wowza® software and third party software is properly licensed.

Trademarks

Wowza, Wowza Media Systems, Wowza Media Server and related logos are either registered trademarks or trademarks of Wowza Media System, LLC in the United States and/or other countries.

Adobe and Flash are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States and/or other countries.

Microsoft and Silverlight are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.

QuickTime, iPhone, iPad and iPod touch are either registered trademarks or trademarks of Apple, Inc. in the United States and/or other countries.

Other product names, logos, designs, titles, words or phrases mentioned may be third party registered trademarks or trademarks in the United States and/or other countries.

Third party trademarks are used solely to identify and describe third party products as being compatible with Wowza products. Wowza is in no way sponsored, endorsed by or otherwise affiliated with any such third party trademark owners.

Copyright © 2006 - 2014 Wowza Media Systems, LLC. All rights reserved.

Third Party Copyright Notices

Apache Commons Lang: Copyright © 2001-2011, The Apache Software Foundation

Apache Commons Modeler Component: Copyright © 2002-2008, The Apache Software Foundation

Bouncy Castle Crypto APIs: Copyright © 2000 – 2008, Legion of the Bouncy Castle

Jackson JSON Parser: Copyright © 2009, FasterXML, LLC

Java ID3 Tag Library and JLayer 1.0 (classic): Copyright © 1991, 1999, Free Software Foundation, Inc.

Joda-Time version 2.1: Copyright © 2012, Joda.org.

Libgcc_s-4 library and Libstdc++ library: Copyright © 2011, Free Software Foundation, Inc.

LibVA libraries: Copyright © 2007, Intel Corporation. All rights reserved.

Log4j and Mina: Copyright © 2006, The Apache Software Foundation

NVIDIA Video Codec SDK: Copyright © 2013, NVIDIA Corporation

Silver Egg Technology: Copyright © 2001, Silver Egg Technology

Speex Codec: Copyright © 2002-2003, Jean-Marc Valin/Xiph.Org Foundation

VideoEncoderH264VAAPIlmpl: Copyright © 2012, Intel Corporation. All Rights Reserved.

Vorbis/Ogg libraries: Copyright © 2011, Xiph.Org Foundation

WebM VP8 Codec libraries: Copyright © 2010, Google Inc. All rights reserved.

Copyright © 2006 - 2014 Wowza Media Systems, LLC. All rights reserved.

Document History

Version	Description	Release date
Doc v3.5.0	Initial document release for Wowza Media Server 3.5	11-08-2012
Doc v3.6.0	Updated for Wowza Media Server 3.6	05-28-2013
Doc v3.6.1	Updated for Wowza Media Server 3.6.1	05-29-2013
Doc v3.6.2	Updated for Wowza Media Server 3.6.2	06-06-2013
Doc v3.6.3	Updated for Wowza Media Server 3.6.3	11-01-2013
Doc v3.6.4	Updated for Wowza Media Server 3.6.4	01-03-2014

Note

A more recent version of this document may be available online. See the Wowza Media Systems Documentation webpage for the latest updates.

Copyright © 2006 - 2014 Wowza Media Systems, LLC. All rights reserved.

Table of Contents

Copyright © 2006 - 2014 Wowza Media Systems, LLC. All rights reserved.

Introduction

This document is a reference for the configuration files that are located in the **[install-dir]/conf** folder of Wowza Media Server. It covers the most commonly used settings and configuration items. All internal configuration settings are omitted.

Copyright © 2006 - 2014 Wowza Media Systems, LLC. All rights reserved.

Server Configuration File

Server.xml

The **Server.xml** configuration file is used to define the server level configuration.

CommandInterface/HostPort

HostPort definition for the Wowza Media Server command interface. This interface is only used to allow remote shutdown of the server. The entire **CommandInterface** section should be commented-out when a server is deployed into production.

AdminInterface/ObjectList

List of objects made available through the Java Management Extensions (JMX) interface. For more information, see "AdminInterface/ObjectList" in How to use JConsole with Wowza Media Server.

JMXRemoteConfiguration/[*]

Java Management Extensions (JMX) interface configuration. For more information, see "Remote JMX Configuration" in How to use JConsole with Wowza Media Server.

UserAgents

Defines HTTP requests that will be interpreted as RTMPT requests. Any request that includes one of these values in an HTTP request header is considered to be an RTMPT request.

HandlerThreadPool/PoolSize, TransportThreadPool/PoolSize

Defines the maximum size of server-level threads in the handler and transport thread pools. The handler thread pool is used to process incoming messages. The transport thread pool is used to read/write data from the transport sockets. Server-level thread pools are only used if a virtual host's thread pool size is set to **0**. This server-level thread pool is also used to process the shutdown command; therefore, it should never be set to a value less than **10**.

Copyright © 2006 - 2014 Wowza Media Systems, LLC. All rights reserved.

RTP/DatagramStartingPort

Lowest UDP port value assigned to incoming UDP streams. Ports are assigned starting with this value and incremented by 1. The most common value for RTSP/RTP-based servers is **6970**. If you plan to support RTSP/RTP, native RTP, or MPEG-TS streams, you should open UDP ports 6970-9999.

RTP/DatagramPortSharing

If set to **true**, then UDP ports (both unicast and multicast) can be shared between Session Description Protocol (SDP) files and MPEG-TS streams or shared between application instances. This enables two SDP files to share RTP ports. For example, if you have a single video stream with multiple audio streams (each in a different language), then you can create two SDP files that share the single RTP video stream and refer to a unique audio stream. This also enables the same SDP file or MPEG-TS stream to be loaded by different application instances.

If set to **false**, UDP ports aren't allowed to be shared and an error is generated if an attempt is made to reuse a UDP port.

Properties

Properties are typed name/value pairs. These properties are available in the server-side API through the **IServer.getProperties()** interface.

VHosts.xml

The **VHosts.xml** configuration file is used to define virtual host environments. By default, Wowza Media Server ships with a single virtual host environment named **_defaultVHost_**. For a complete description of this configuration file, see "Virtual Hosting" in the Wowza Media Server User's Guide.

log4j.properties

The **log4j.properties** file is used to configure Wowza Media Server logging. The server uses the Java-based log4j logging system. By default, the server is configured to log basic information to the console window and detailed information in W3C Extended Common Log

Copyright © 2006 - 2014 Wowza Media Systems, LLC. All rights reserved.

Format (ECLF) to log files. For more information about how to configure the logging system, see "Logging" in the Wowza Media Server User's Guide.

Copyright © 2006 - 2014 Wowza Media Systems, LLC. All rights reserved.

Virtual Host Configuration Files

Authentication.xml

The **Authentication.xml** file is used to configure authentication for RTSP and HTTP sessions.

Method/[Name, Description, Class, Properties]

Definition for an authentication method. **Name** is the name of the method. **Class** is the full path to the class that provides the underlying implementation. **Properties** are additional configuration properties.

CEACaptionConversion.xml

The **CEACaptionConversion.xml** file contains settings for conversion of closed captions to CEA-608 captions.

UTF8CharacterMapping/[UTFKey, UTFMapping]

A mapping of a UTF8 character that isn't supported by CEA-608 captions to a reasonable substitute that's supported. The **UTFKey** value is the character that will be replaced by the **UTFMapping** value.

DVR.xml

The **DVR.xml** file contains definitions for Wowza nDVR recording and playback.

DVRRecorder/[Name, Description, BaseClass, Properties]

Definition for a DVRRecorder item. **Name** is the name of the recorder. **BaseClass** is the full path to the class that provides the underlying implementation. **Properties** are additional configuration properties.

Copyright © 2006 - 2014 Wowza Media Systems, LLC. All rights reserved.

DVRStore/[Name, Description, Properties]

Definition for a DVRStore item. **Name** is the name of the store. **Properties** are additional configuration properties.

HTTPStreamers.xml

The **HTTPStreamers.xml** file contains definitions for HTTP streaming such as Cupertino streaming (Apple® HTTP Live Streaming), San Jose streaming (Adobe® HTTP Dynamic Streaming), and Microsoft® Smooth Streaming.

HTTPStreamer/[Name, Description, BaseClass, Properties]

Definition for an HTTPStreamer item. **Name** is the name of the HTTPStreamer. **BaseClass** is the full path to the class that provides the underlying implementation. **Properties** are additional configuration properties.

HTTPStreamer/ApplicationContextClass

Full path to the class that's used to provide the application instance context.

HTTPStreamer/RequestFilters

Used to determine if a given HTTP request should be processed by this HTTPStreamer.

HTTPStreamer/IdleFrequency

Time (in milliseconds) between idle events.

HTTPStreamer/LiveStreamPacketizer

Packetizer used by this HTTPStreamer.

HTTPStreamer/LiveStreamRepeater

Repeater used by this HTTPStreamer.

Copyright © 2006 - 2014 Wowza Media Systems, LLC. All rights reserved.

LiveStreamPacketizers.xml

The **LiveStreamPacketizers.xml** file contains definitions for HTTP live stream packetizers that are used for HTTP streaming technology, such as Cupertino streaming (Apple HLS) and Smooth Streaming.

LiveStreamPacketizer/[Name, Description, BaseClass, Properties]

Definition for a LiveStreamPacketizer item. **Name** is the name of the LiveStreamPacketizer. **BaseClass** is the full path to the class that provides the underlying implementation. **Properties** are additional configuration properties.

LiveStreamPacketizer/IsRepeater

Set to **true** if this live stream packetizer is a live stream repeater packetizer.

LiveStreamPacketizer/LiveRepeaterPlayerClass

Full path to the class used on the player side for live stream repeater.

LiveStreamPacketizer/LiveRepeaterReceiverClass

Full path to the class used on the receiver side for live stream repeater.

LiveStreamTranscoders.xml

The **LiveStreamTranscoders.xml** file contains the base Wowza Transcoder AddOn configuration.

MaximumConcurrentTranscodes

Defines the maximum number of concurrent incoming streams that are transcoded at any point in time. This setting is useful when you have a Daily or Monthly license and you want to limit the number of concurrent transcodes for billing purposes. A value of **0** allows an unlimited number of concurrent transcodes.

Copyright © 2006 - 2014 Wowza Media Systems, LLC. All rights reserved.

LiveStreamTranscoder/[Name, Description, BaseClass, Properties]

Definition for a transcoder item. **Name** is the name of the transcoder. **BaseClass** is the full path to the class that provides the underlying implementation. **Properties** are additional configuration properties.

Properties

Property name	Description
disableTranscoderBehindFilter	Controls the overflow protection mechanism. If the property value is set to **false** and the transcoder can't keep up with the transcoding demand, it will drop audio and video frames based on the remaining properties defined here. For example, if the transcoder is behind by more than 1500 milliseconds (1.5 seconds), it will try to catch up by dropping every other frame; if it's behind by more than 2500 milliseconds (2.5 seconds) it will drop 2 out of every 3 frames; and so on. To turn off this protection mechanism, set the **disableTranscoderBehindFilter** property value to **true**.

MediaCasters.xml

The **MediaCasters.xml** file contains MediaCaster definitions.

Connections/[*]

Defines the socket configuration for MediaCaster connections.

MediaCaster/[Name, Description, BaseClass, Properties]

Definition for a MediaCaster item. **Name** is the name of the MediaCaster. **BaseClass** is the full path to the class that provides the underlying implementation. **Properties** are additional configuration properties.

Copyright © 2006 - 2014 Wowza Media Systems, LLC. All rights reserved.

MediaCaster/StreamType

The stream type in **Streams.xml** that's used when using this MediaCaster.

MediaCaster/ConnectionTimeout

The time (in milliseconds) that this MediaCaster will wait when connecting to a remote service.

MediaCaster/KeepAliveTime

The time (in milliseconds) that this MediaCaster will continue to run after the last user of the MediaCaster disconnects before the MediaCaster is shut down.

MediaReaders.xml

The **MediaReaders.xml** file contains file format reader definitions.

MediaReader/[Name, Description, ClassBase, Properties]

Definition for a MediaReader item. **Name** is the name of the MediaReader. The **Name** doubles as the stream name prefix that's used to invoke this MediaReader. **ClassBase** is the full path to the class that provides the underlying implementation. **Properties** are additional configuration properties.

MediaReader/FileExtension

The default file extension.

MediaReader/ContentType

The content type for the media reader. The supported values are:

- **media**. Media file (such as FLV or MP4).
- **medialist**. Media playlist for describing adaptive bitrate video streams.
- **timedtext**. Timed text file format (such as TTML) used for closed captioning data.

Copyright © 2006 - 2014 Wowza Media Systems, LLC. All rights reserved.

MediaWriters.xml

The **MediaWriters.xml** file contains file format writer definitions.

MediaWriter/[Name, Description, ClassBase]

Definition for a MediaWriter item. **Name** is the name of the MediaWriter. The **Name** doubles as the stream name prefix that's used to invoke this MediaWriter. **ClassBase** is the full path to the class that provides the underlying implementation.

MediaWriter/FileExtension

The default file extension.

MP3Tags.xml

The **MP3Tags.xml** file contains a mapping of MP3 ID3 tags to metadata property names.

MP3Tag/[ID, Name]

Maps an internal MP3 ID3 tag ID (**ID**) to a metadata property name (**Name**).

RTP.xml

The **RTP.xml** file contains a list of depacketizers that handle incoming native RTP and MPEG-TS streams.

DePacketizer/[Name, Description, Class, Properties]

Definition for a DePacketizer item. **Name** is the name of the DePacketizer. **Class** is the full path to the class that provides the underlying implementation. **Properties** are additional configuration properties.

StartupStreams.xml

The **StartupStreams.xml** file contains a list of streams to startup at virtual host startup.

Copyright © 2006 - 2014 Wowza Media Systems, LLC. All rights reserved.

StartupStream/Application

Application name and instance name to use when starting the stream in the form **[application]/[appInstance]**. If **[appInstance]** is omitted, the default application name **_definst_** is used.

StartupStream/MediaCasterType

MediaCaster type name from **MediaCasters.xml** to use to start the stream.

StartupStream/ StreamName

Name of the stream to startup. If needed, this value should include a stream name prefix.

Streams.xml

The **Streams.xml** file contains a list of stream type definitions.

Stream/[Name, Description, ClassBase, ClassPlay, Properties]

Definition for a Stream item. **Name** is the name of the Stream. **ClassBase** is the full path to the class that provides the underlying implementation for the **IMediaStream** interface. **ClassPlay** is the full path to the class that provides the underlying implementation for the **IMediaStreamPlay** interface. **Properties** are additional configuration properties.

TimedTextProviders.xml

The **TimedTextProviders.xml** file contains definitions for providers of timed text data. Wowza Media Server contains timed text providers for live and video on demand (VOD) closed captioning.

TimedTextTypes

Defines the set of timed text types that are available to Wowza Media Server.

TimedTextType/[Name]

Name is the name of the TimeTextType and is referred to by timed text providers and OutputTypes.

Copyright © 2006 - 2014 Wowza Media Systems, LLC. All rights reserved.

OutputTypes

Defines the set of available timed text output types. Timed text providers must refer to one or more of these output types. For example, a **OutputType/Name** value of **CEA608** is specified because one or more caption providers can create CEA-608 captions.

OutputType/[Name, Interface, TimedTextType]

Definition for a timed text output type. **Name** is the name of the output type and is referred to by a timed text provider if it can provide that output type. **Interface** is the interface that's associated with the output type. A timed text provider must implement the interface for each declared output type. **TimedTextType** must refer to one of the declared **TimedTextTypes**.

LiveIngestTypes

Defines the set of available live ingestion types. Live timed text providers must refer to one or more of these ingestion types. For example, a **LiveIngestType/Name** value of **onTextData** is specified because one or more live caption providers can ingest Action Message Format (AMF) **onTextData** events.

LiveIngestType/[Name, TimedTextType]

Definition for a timed text live ingest type. **Name** is the name of the live ingest type and is referred to by a live timed text provider if it can support that type of ingestion. **TimedTextType** must refer to one of the declared **TimedTextTypes**.

VODTimedTextProviders

Contains a list of VODTimedTextProvider items.

VODTimedTextProvider/[Name, Description, TimedTextType, OutputTypes, BaseClass, FileExtensions, Properties]

A declared timed text provider for video on demand.

- **Name**. The provider name. The **TimedText/VODTimedTextProviders** setting in **Application.xml** references this provider in order to enable it.
- **TimedTextType**. Must refer to one of the declared **TimedTextTypes**.

Copyright © 2006 - 2014 Wowza Media Systems, LLC. All rights reserved.

- **OutputTypes**. A comma-separated list of supported **OutputTypes**.
- **BaseClass**. The full path to the class that provides the underlying implementation.
- **FileExtensions** (optional). A comma-separated list of supported file extensions used for this provider.
- **Properties**. Additional configuration properties.

LiveTimedTextProviders

Contains a list of LiveTimedTextProvider items.

LiveTimedTextProvider/[Name, Description, TimedTextType, LiveIngestType, OutputTypes, BaseClass, Properties]

A declared timed text provider for live streams.

- **Name**. The provider name.
- **TimedTextType**. Must refer to one of the declared **TimedTextTypes**.
- **LiveIngestType**. Must refer to one of the declared **LiveIngestTypes.**
- **OutputTypes**. A comma-separated list of supported **OutputTypes**.
- **BaseClass**. The full path to the class that provides the underlying implementation.
- **Properties**. Additional configuration properties.

VHost.xml

The **VHost.xml** file defines the virtual host configuration.

HostPortList/HostPort

A list of IP addresses and TCP ports that Wowza Media Server will bind to for incoming and outgoing streaming connections. Wowza Media Server can be configured for any number of TCP ports. **HostPorts** are used to stream RTMP, RTSP, and HTTP. A **HostPort** can be configured to use Secure Sockets Layer (SSL) encryption. HTTPProviders configuration is done on a per-HostPort basis.

Copyright © 2006 - 2014 Wowza Media Systems, LLC. All rights reserved.

HostPortList/HostPort/ProcessorCount

The number of threads to use to service connections. For more information about recommended values to use based on server resources, see Performance Tuning.

HostPortList/HostPort/[IpAddress, Port]

IpAddress is the IP address or domain name of the address that Wowza Media Server will listen to for incoming requests. If **IpAddress** is set to the wildcard (*****) character, the server will try to listen for incoming connections on all available network interfaces. **Port** is a comma-separated list of ports.

HostPortList/HostPort/SocketConfiguration/[*]

The detailed socket connection configuration that's created by this **HostPort** definition at runtime. You can use these settings to tune the performance of the socket connections that are used to send data into and out of Wowza Media Server. **SendBufferSize**, **ReceiveBufferSize**, and **ReadBufferSize** are the most important settings in this group. They define the size of the memory buffers that are used during data transfer over the socket connection. Values of **0** for **SendBufferSize** and **ReceiveBufferSize** instruct Wowza Media Server to use the operating system default values for these settings. For operating systems that support it, TCP auto-tuning at the kernel level may be used to set the buffer sizes dynamically for individual connections. For more information about recommended values to use based on server resources, see Performance Tuning.

The **ReuseAddress** and **KeepAlive** settings should both be set to **true**. They are only provided for completeness.

The **AcceptorBackLog** setting controls the maximum number of TCP connection requests that can be pending before new connection requests are refused. Wowza Media Server will respond to TCP connection requests as fast as possible. This value shouldn't be set to a value of less than **50**. Setting this value to **-1** allows the operating system to control the maximum number of pending TCP connection requests. (Note: This isn't always the best setting. Some platforms will then use a very small value that can greatly increase connection times.)

Copyright © 2006 - 2014 Wowza Media Systems, LLC. All rights reserved.

HostPortList/HostPort/HTTPStreamerAdapterIDs

A comma-separated list of HTTPStreamers that this **HostPort** will include when processing HTTP requests. **HTTPStreamerAdapterIDs** can contain none, one, or more of the following values (separated by commas): **cupertinostreaming** (Apple HLS), **smoothstreaming**, **sanjosestreaming** (Adobe HDS), **dvrchunkstreaming**, **mpegdashstreaming**.

HostPortList/HostPort/HTTPProviders

A list of HTTPProviders that this **HostPort** will include when processing HTTP requests. For more information, see "HTTPProviders" in the Wowza Media Server User's Guide.

HostPortList/HostPort/SSLConfig/[*]

The Secure Sockets Layer (SSL) configuration for a given **HostPort**.

- **KeyStorePath**. The full path to the key-store file.
- **KeyStorePassword**. The key-store password.
- **KeyStoreType**. The key-store type. The default value is **JKS** for Sun Java JRE.
- **SSLProtocol**. The cryptographic protocol. The default value is **TLS** (Transport Layer Security).
- **Algorithm**. The encryption algorithm. The default value is **SunX509** for Sun Java JRE.

HTTPStreamerAdapters

Defines a list of HTTPStreamers that are enabled for **HostPort** entries. All HTTPStreamers listed here can be streamed using the **HostPort** IP address and port combination. If you want to disable a particular HTTP streaming protocol, you can remove it from this list.

HandlerThreadPool/PoolSize, TransportThreadPool/PoolSize

Defines the maximum size of the virtual host level threads in the handler and transport thread pools. The handler thread pool is used to process incoming messages while the transport thread pool is used to read/write data from the transport sockets. If the pool size is set to **0** for a given thread pool type, the server-level thread pool of the same type is used for this virtual host. For more information about recommended values to use based on server resources, see Performance Tuning.

Copyright © 2006 - 2014 Wowza Media Systems, LLC. All rights reserved.

IdleWorkers/[WorkerCount, CheckFrequency]

WorkerCount controls the number of threads being used to generate idle events. **CheckFrequency** is the time (in milliseconds) between checking to see if a client has been idle for **Client/IdleFrequency**. The **CheckFrequency** value should be at least four times smaller than the **Client/IdleFrequency** value. For more information about recommended values to use based on server resources, see Performance Tuning.

NetConnections/[ProcessorCount, IdleFrequency, SocketConfiguration]

The settings in this section are used to tune connections made between computers running Wowza Media Server (for example, when using the live stream repeater).

HTTPTunnel/KeepAliveTimeout

The keepalive time (in milliseconds) for RTMPT, RTMPTE, and RTMPS connections.

Client/[ClientTimeout, IdleFrequency]

ClientTimeout is the time (in milliseconds) that the server will wait before shutting down an unresponsive client connection. **IdleFrequency** is the time (in milliseconds) between idle events. For basic video on demand (VOD) streaming, a value of **250** provides the best reliability versus performance ratio. For live streaming, a value between **125** and **250** is more desirable. It will increase the frequency at which media data is sent to Adobe Flash clients. If you adjust this value, be sure to also adjust the **IdleWorkers/CheckFrequency** value to a value that's at least four times smaller.

RTP/IdleFrequency

The time (in milliseconds) between idle events for RTP connections.

RTP/DatagramConfiguration/[Incoming, Outgoing]/[*]

The datagram socket configuration for incoming and outgoing RTP connections.

- **ReceiveBufferSize**. The size (in bytes) of the incoming UDP buffer.
- **SendBufferSize**. The size (in bytes) of the outgoing UDP buffer.

Copyright © 2006 - 2014 Wowza Media Systems, LLC. All rights reserved.

- **MulticastTimeout**. The timeout value (in milliseconds) for multicast polling.
- **DatagramMaximumPacketSize**. The maximum size (in bytes) for a single incoming UDP packet.

RTP/[*]/ProcessorCount

The number of threads allocated to incoming and outgoing unicast and multicast UDP streams.

Application/ApplicationTimeout

The time (in milliseconds) that the server will wait before shutting down an application to which no clients are connected. A value of **0** keeps applications running until the virtual host is shut down.

Application/PingTimeout

The time (in milliseconds) that the server will wait for a ping response from the client. The ping mechanism is an RTMP internal ping (not an ICMP ping) that's used to validate a client connection. If set to **0**, the server will wait indefinitely.

Application/ValidationFrequency

The time (in milliseconds) that the server will wait during server-to-client validation. Validation only occurs if the client stops sending data to the server. Validation is done by sending a ping request from the server to the client. If set to **0**, the server won't validate client connections.

Application/MaximumPendingWriteBytes

The maximum number of bytes that are allowed to be queued and waiting to be sent. If this value is exceeded for an individual client connection, the connection is terminated. If set to **0**, there is no maximum value.

Application/MaximumSetBufferTime

The maximum allowed client-side buffer time (in milliseconds). A value of **0** removes the buffer time limit. This property is useful to deter stream rippers that might connect to a

Copyright © 2006 - 2014 Wowza Media Systems, LLC. All rights reserved.

Wowza Media Server and set a very large buffer time in order to steal content more quickly. This can lead to server-side memory run-up.

StartStartupStreams

When set to **true**, streams that are defined in **StartupStreams.xml** are instantiated at server startup. When set to **false**, streams aren't started at server startup.

Properties

Properties are typed name/value pairs. All virtual host properties are copied to virtual hosts upon creation. These properties are available in the server-side API through the **IVHost.getProperties()** interface.

Copyright © 2006 - 2014 Wowza Media Systems, LLC. All rights reserved.

Application Configuration Files

Application.xml

The **Application.xml** file is used to configure an application.

ApplicationTimeout

The time (in milliseconds) that the server will wait before shutting down an application to which no clients are connected. A value of **0** will keep applications running until the virtual host is shut down. If this value isn't provided (section commented-out), the value set in the **VHost.xml** file is used.

PingTimeout

The time (in milliseconds) that the server will wait for a ping response from the client. The ping mechanism is an RTMP internal ping (not an ICMP ping) that's used to validate a client connection. If set to **0**, the server will wait indefinitely. If this value isn't provided (section commented-out), the value set in the **VHost.xml** file is used.

ValidationFrequency

The time (in milliseconds) that the server will wait during server-to-client validation. Validation only occurs if the client stops sending data to the server. Validation is done by sending a ping request from the server to the client. If set to **0**, the server won't validate client connections. If this value isn't provided (section commented-out), the value set in the **VHost.xml** file is used.

MaximumPendingWriteBytes

The maximum number of bytes that are allowed to be queued and waiting to be sent. If this value is exceeded for an individual client connection, then the connection is terminated. If set to **0**, there is no maximum value. If this value isn't provided (section commented-out), the value set in the **VHost.xml** file is used.

Copyright © 2006 - 2014 Wowza Media Systems, LLC. All rights reserved.

MaximumSetBufferTime

The maximum number of milliseconds honored server-side for client-side calls to **NetStream.setBufferTime(seconds)**. To turn off this check, set this value to **0**. The default value is **60000** (60 seconds). This setting is used to protect against spoofing threats (such as those posed by Replay Media Catcher and Grab Pro), which can set a very large client-side buffer to trick a server into sending all the media data at once. This can cause the server to consume a large amount of Java heap memory.

MaximumStorageDirDepth

The maximum number of subfolders that are allowed in any storage path. This setting helps to protect against symbolic link loops.

Connections/AutoAccept

This setting determines if the application automatically accepts incoming Adobe Flash client connection requests. If set to **true**, all incoming connection requests are accepted automatically. If set to **false**, the application must make a server-side call to **client.acceptConnection()** to accept an incoming connection request.

Connections/AllowDomains

A comma-delimited list of domain names or IP addresses for which client connections are accepted. The domain names or IP addresses that are specified represent the domain name or IP address of an Adobe Flash SWF file connecting to Wowza Media Server or the IP address of a client connecting to Wowza Media Server. If no value is specified, then connections from all domains or IP addresses are accepted.

For example, if you have the SWF file *http://www.mycompany.com/flash/myflashmovie.swf*, you can set **AllowDomains** to **www.mycompany.com** in order to configure Wowza Media Server so that only clients from the **mycompany.com** domain can access the server. You can also add an IP address (or IP address wildcard) to accept all connections from a particular IP address. You might filter based on IP address when you're working with a client-side encoder that doesn't provide a valid referrer.

Copyright © 2006 - 2014 Wowza Media Systems, LLC. All rights reserved.

You can use the wildcard character (*****) to match partial domain names or IP addresses. For example, if you want to match all domain names that end with **mycompany.com**, you would specify the domain name ***.mycompany.com**.

Allow-domains processing occurs just before the **onConnect** event method. So if you want to provide finer-grained access control to your server, you can override the **onConnect** event handler in a custom module and provide your own filtering mechanism.

Streams/StreamType

The name (as defined in the **Streams.xml** file) of the default stream type for this application. For more information about stream types, see "Stream Types" in the Wowza Media Server User's Guide.

Streams/StorageDir, Streams/KeyDir, SharedObjects/StorageDir

Streams/StorageDir is the full path to the directory where the application reads and writes media files. **KeyDir** is the directory where the Cupertino streaming (Apple HLS) AES-128 encryption keys are stored. **SharedObjects/StorageDir** is the full path to the directory where the application reads and writes remote stored object data. If no values are specified, an application uses the following directories:

```
%WMSCONFIG_HOME%/applications/[application]/streams/[appinstance]
%WMSCONFIG_HOME%/applications/[application]/sharedobjects/[appinstance]
%WMSCONFIG_HOME%/applications/[application]/keys/[appinstance]

%WMSCONFIG_HOME%            the value of the environment variable WMSCONFIG_HOME
[application]               the name of the application
[appinstance]               the name of the application instance
```

The following variables are supported:

```
${com.wowza.wms.AppHome}                          - Application home directory
${com.wowza.wms.ConfigHome}                       - Configuration home directory
${com.wowza.wms.context.VHost}                    - Virtual host name
${com.wowza.wms.context.VHostConfigHome}          - Virtual host config directory
${com.wowza.wms.context.Application}              - Application name
${com.wowza.wms.context.ApplicationInstance}      - Application instance name
```

Streams/LiveStreamPacketizers

The HTTP streaming packetization schemes to use for incoming live streams. Live stream packetization is done to make a stream available for HTTP streaming to the Apple® iPhone®

Copyright © 2006 - 2014 Wowza Media Systems, LLC. All rights reserved.

and iPod touch®, Adobe Flash, and Microsoft® Silverlight®. It's is also done to enable DVR for a live stream. **LiveStreamPacketizers** can contain none, one, or more of the following values (separated by commas):

LiveStreamPacketizers	Description
cupertinostreamingpacketizer	Cupertino: iOS devices
smoothstreamingpacketizer	Smooth: Microsoft Silverlight
sanjosestreamingpacketizer	San Jose: Adobe Flash
mpegdashstreamingpacketizer	MPEG-DASH
cupertinostreamingrepeater	Cupertino: Live stream repeater for iOS devices
smoothstreamingrepeater	Cupertino: Live stream repeater for Microsoft Silverlight
sanjosestreamingrepeater	San Jose: Live stream repeater for Adobe Flash
dvrstreamingpacketizer	Wowza nDVR: Streaming
dvrstreamingrepeater	Wowza nDVR: Live stream repeater

Streams/Properties

Defined properties for this application will override properties defined in **Streams.xml**.

Transcoder/LiveStreamTranscoder

The name (as defined in the **LiveStreamTranscoders.xml** file) of the default transcoder handler for this application. If set to **transcoder**, the live stream transcoder is enabled for this application. If no value is specified, incoming streams aren't transcoded.

Transcoder/Templates

A comma-separated list of template names to search when matching an incoming stream to a live stream transcoder template. If the live stream transcoder is enabled, each new live stream that's published to the application can be transcoded. Wowza Media Server will search the **Templates** list for the first template file that exists. After it finds an existing transcoder template, it uses that template to transcode the stream. If no matches are found, the stream won't be transcoded. The default value for this setting is:

```
${SourceStreamName}.xml,transrate.xml
```

Copyright © 2006 - 2014 Wowza Media Systems, LLC. All rights reserved.

The first item in the list uses the variable **${SourceStreamName}**. If there's a transcoder template in the template directory with the name **[stream-name].xml** (where **[stream-name]** is the name of the incoming stream), then that transcoder template is used. If this file doesn't exist, the **transrate.xml** template is used (if it exists). If neither template exists, then the stream won't be transcoded. Only the variable **${SourceStreamName}** is supported.

Transcoder/ProfileDir

Reserved for future use.

Transcoder/TemplateDir

The full path to the directory where the application looks for transcoder templates that are used to control the live stream transcoder. By default, Wowza Media Server looks for transcoder templates in the **[install-dir]/transcoder/templates** folder. You can also define a per-application templates folder by using path variables. For example, if you want to set up a per-application templates folder, specify the following path:

```
${com.wowza.wms.context.VHostConfigHome}/conf/${com.wowza.wms.context.Application}/
profiles
```

The following variables are supported:

```
${com.wowza.wms.AppHome}                          - Application home directory
${com.wowza.wms.ConfigHome}                       - Configuration home directory
${com.wowza.wms.context.VHost}                    - Virtual host name
${com.wowza.wms.context.VHostConfigHome}          - Virtual host config directory
${com.wowza.wms.context.Application}              - Application name
${com.wowza.wms.context.ApplicationInstance}      - Application instance name
```

DVR/Recorders

The name (as defined in the **DVR.xml** file) of the default DVR recorder for this application.

DVR/Store

The name (as defined in the **DVR.xml** file) of the default DVR store for this application.

DVR/WindowDuration

The duration (in seconds) of material available for DVR playback. The default value of **0** denotes that there's no DVR window (all data is available for DVR playback).

Copyright © 2006 - 2014 Wowza Media Systems, LLC. All rights reserved.

DVR/StorageDir

The full path to the directory where the application reads and writes Wowza nDVR data.

The following variables are supported:

```
${com.wowza.wms.AppHome}                          - Application home directory
${com.wowza.wms.ConfigHome}                       - Configuration home directory
${com.wowza.wms.context.VHost}                    - Virtual host name
${com.wowza.wms.context.VHostConfigHome}          - Virtual host config directory
${com.wowza.wms.context.Application}              - Application name
${com.wowza.wms.context.ApplicationInstance}      - Application instance name
```

DVR/ArchiveStrategy

This value tells the DVR store what to do with an old stream when a new stream of the same app instance and stream name starts. The default value is **append**. **ArchiveStrategy** can contain only one of the following values:

ArchiveStrategy	Description
append	Append new stream information to the end of previous store.
delete	Delete the old DVR store and start a new one.
version	Create a new version of the DVR store.

DVR/Properties

Defined properties for this application will override properties defined in **DVR.xml**.

TimedText/[VODTimedTextProviders, Properties]

VODTimedTextProviders contains a comma-separated list of video on demand (VOD) caption providers, which are declared in **TimedTextProviders.xml**. **Properties** are additional configuration properties.

HTTPStreamers

The available HTTP streaming types. HTTP is done to make a stream available for HTTP streaming to the Apple iPhone and iPod touch, Adobe Flash, and Microsoft Silverlight. It also allows the DVR streaming to repeat DVR audio and video content from the origin to the

Copyright © 2006 - 2014 Wowza Media Systems, LLC. All rights reserved.

edge. **HTTPStreamers** can contain none, one, or more of the following values (separated by commas):

HTTPStreamers	Description
cupertinostreaming	Cupertino: HTTP streaming to iPhone and iPod touch
smoothstreaming	Smooth: HTTP streaming to Microsoft Silverlight
sanjosestreaming	San Jose: HTTP streaming to Adobe Flash
mpegdashstreaming	MPEG-DASH
dvrchunkstreaming	DVR: Enable streaming from origin to edge

Client/IdleFrequency

The time (in milliseconds) between idle events. Idle events represent the heartbeat of streaming for Adobe Flash and RTSP/RTP streaming. It represents how often new data is sent to the player. If this value is set to **-1**, then the value specified in the **VHost.xml** file is used.

Client/Access/[*]

Controls the default access that an Adobe Flash client connection has to assets associated with a particular Wowza Media Server application. An individual client's access can be modified through the server-side API. This is most commonly done in the **onConnect** or **onConnectAccept** event handler. Each of these settings is a comma-delimited list of names that are matched against the asset name (stream name or shared object name) to control access. If any part of the asset name matches one of the elements in the list, then the given access is granted. The values are case-sensitive. If no parameter value is specified, then access is denied to all clients. If the parameter value is set to the wildcard (*****) character, then access is granted to all clients. For example if **StreamReadAccess** is set to **testa/testb;testc**, then the following stream name would be granted the following access:

```
testc                                Granted Access
testc/test                           Granted Access
testC/test                           Denied Access (incorrect case)
testa/testb                          Granted Access
testa/testb123                       Granted Access
testa/testb/file123                  Granted Access
testa/test                           Denied Access (incomplete match)
```

Copyright © 2006 - 2014 Wowza Media Systems, LLC. All rights reserved.

- **StreamReadAccess:** Controls access to view or listen to a **NetStream** object.
- **StreamWriteAccess:** Controls access to write or publish to a **NetStream** object.
- **StreamAudioSampleAccess:** Controls access to call **SoundMixer.computeSpectrum()** to grab the waveform data of a **NetStream** object.
- **StreamVideoSampleAccess:** Controls access to call **BitmapData.draw()** to take a snapshot of a **NetStream** object.
- **SharedObjectReadAccess:** Controls access to read values from a RemoteSharedObject.
- **SharedObjectWriteAccess:** Controls access to write values to a RemoteSharedObject.

RTP/Authentication/[PublishMethod, PlayMethod]

The authentication method used to secure RTSP connections to Wowza Media Server. **PublishMethod** is for incoming or publishing connections and **PlayMethod** is for outgoing or play connections. Authentication methods are defined and configured in **Authentication.xml**. By default, there are three authentication methods:

- **none**. No authentication.
- **basic** . User name and password are sent in cleartext.
- **digest**. Password is hashed using MD5 and is never sent in cleartext over the network.

User names and passwords are stored in the **conf/publish.password** file. The **publish.password** file contains one line per user formatted as `[user name][space][password]`. The authentication method can also be set at the virtual host level in **VHost.xml**.

RTP/[AVSyncMethod, MaxRTCPWaitTime]

Controls how Wowza Media Server synchronizes audio and video channels when receiving an RTP stream. **AVSyncMethod** configures the methodology used to synchronize the audio and video channels. There are three possible values:

Copyright © 2006 - 2014 Wowza Media Systems, LLC. All rights reserved.

- **senderreport**. Use the Sender Report (SR) packets that are sent over the Real-time Control Protocol (RTCP) channel. This is the default value.
- **rtptimecode**. Assume that RTP timecodes are absolute timecode values.
- **systemclock**. Synchronize based on the system clock.

MaxRTCPWaitTime is the maximum time (in milliseconds) that Wowza Media Server will wait to receive an SR packet over the RTCP channel. If no SR packets are received within this time period, the server will default to using the **rtptimecode** method.

RTP/IdleFrequency

The time in (milliseconds) between RTP idle events. Idle events are used to send new media data and events to an RTP or MPEG-TS session.

RTP/RTSPSessionTimeout

The elapsed time (in milliseconds) after which an idle RTSP session is determined to be stale and is disconnected. To turn off idle disconnect, set this value to **0**. Wowza Media Server monitors all RTSP sessions and looks for periodic RTSP messages or RTCP receiver packets over RTP. If the server hasn't received messages during the session timeout period, the session is disconnected.

RTP/RTSPMaximumPendingWriteBytes

The maximum number of bytes that can wait to be written to an RTSP session. If an RTSP session has more bytes waiting to be written then what's specified by this setting, the session is disconnected. Wowza Media Server monitors the RTSP TCP connection and watches the number of bytes that are waiting to be delivered. If the number of pending bytes exceeds the specified value, then the session is determined to be inactive and the session is disconnected. To turn off maximum pending write byte monitoring, set this value to **0**.

RTP/[RTSPBindIpAddress, RTSPConnectionIPAddress, RTSPOriginIpAddress]

Controls the IP addresses that are exchanged and used during RTSP/RTP port and IP address negotiation when the RTP portion of the stream is delivered over UDP. The **RTSPBindIpAddress** property is the IP address that Wowza Media Server binds to when delivering RTP packets over UDP. If the server is using network address translation routing

Copyright © 2006 - 2014 Wowza Media Systems, LLC. All rights reserved.

(NAT), then this address should be set to the internal IP address of the network interface that's mapped to the external IP address. If NAT isn't being used, then this value should be set to the external IP address of the server. The **RTSPConnectionIpAddress** and **RTSPOriginIpAddress** are the IP addresses that are exchanged as part of the Session Description Protocol (SDP) data. The **RTSPConnectionIpAddress** value is the IP address specified in the `(c=)` line of the SDP data and the **RTSPOriginIpAddress** value is the IP address specified in the `(o-)` line of the SDP data. These two values should be set to the external IP address of the server.

RTP/IncomingDatagramPortRanges

Defines UDP port ranges that this application can use for stream ingestion. Single ports and port ranges can be defined. For example, the following entry enables port **10000** and ports **20000** through **20004**:

```
<IncomingDatagramPortRanges>10000, 20000-20004</IncomingDatagramPortRanges>
```

This setting only affects stream ingestion when pulling streams using Session Description Protocol (SDP) files or MPEG-TS udp:// URLs. It doesn't affect UDP ports that are dynamically assigned during RTSP port negotiation.

RTP/Properties

Defined properties for this application will override properties defined in **RTP.xml**.

MediaCaster/RTP/RTSP/[RTPTransportMode]

Defines the RTSP flavor that's used to re-stream an RTSP/RTP source, such as an IP camera. If set to **interleave**, Wowza Media Server uses RTSP/RTP interleaved (RTP over TCP) to connect to RTSP sources. If set to **udp**, Wowza Media Server uses RTSP/RTP in non-interleaved mode (RTP over UDP) to connect to RTSP sources.

MediaCaster/Properties

Defined properties for this application will override properties defined in **MediaCasters.xml**.

MediaReader/Properties

Defined properties for this application will override properties defined in **MediaReaders.xml**.

Copyright © 2006 - 2014 Wowza Media Systems, LLC. All rights reserved.

MediaWriter/Properties

Defined properties for this application will override properties defined in **MediaWriters.xml**.

LiveStreamPacketizer/Properties

Defined properties for this application will override properties defined in **LiveStreamPacketizers.xml**.

HTTPStreamer/Properties

Defined properties for this application will override properties defined in **HTTPStreamers.xml**.

Repeater/[OriginURL, QueryString]

The origin URL and query string to use when using the application as a live stream repeater. If the application is a DVR repeater, the **OriginURL** is the URL of the DVR origin server from which to retrieve audio and video chunks.

Modules/Module/[Name, Description, Class]

A list of modules that are available to this application. The **Modules** list must contain a unique **Name** element to identify the module. The **Description** field isn't used. The **Class** field is the full package name and class name of the module. For more information, see "Server-side Modules" in the Wowza Media Server User's Guide.

Properties

Properties are typed name/value pairs. All application properties are copied to child application instances during instance creation. These properties are available in the server-side API through the **IApplicationInstance.getProperties()** interface.

Copyright © 2006 - 2014 Wowza Media Systems, LLC. All rights reserved.