



Wowza Streaming Engine™

Dynamic Load Balancing AddOn

Wowza Streaming Engine: Dynamic Load Balancing AddOn



Version: 4.5

<http://www.wowza.com>

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

No Endorsement or Warranty for Third-Party Links and Software

This document contains links to third-party websites ("Linked Sites") that are not under the control of Wowza Media Systems™, LLC ("Wowza™"). Wowza is not responsible for the content on or operation of Linked Sites. If you access Linked Sites, you do so at your own risk and understand that Wowza accepts no responsibility or liability for the content or operation of Linked Sites. Wowza provides these links only as a convenience, and the inclusion of a link does not imply that Wowza endorses such Linked Sites or any content, products, or services available from Linked Sites.

This document also refers to third-party software that is not licensed, sold, or distributed by Wowza (collectively, "Third-Party Software"). Wowza does not endorse, is not responsible for, and accepts no liability related to Third-Party Software. Please ensure that any and all use of Wowza software and third-party software is properly licensed.

Wowza Trademarks

Wowza™, Wowza Streaming Cloud™, Wowza Streaming Engine™, along with other trademarks, logos, trade dress, and other proprietary colors and markings, are each trademarks or registered trademarks of Wowza in the United States and in other countries (collectively, "Wowza Marks"). No right to use Wowza Marks in any way is granted hereunder. Contact sales@wowza.com for information about obtaining the right to use Wowza Marks. Any use of Wowza Marks, authorized or otherwise, shall inure to the sole benefit of Wowza.

Third-Party Trademarks and Copyrights

Trademarks, product names, logos, designs, trade dress, and other proprietary markings of non-Wowza third parties (collectively, "Third-Party Marks") may be trademarks or registered trademarks of their respective owners. Use of Third-Party Marks is for the sole purpose of identifying third-party products and services and does not represent endorsement, sponsorship, partnership, or other affiliation between Wowza and such third parties.

A list of applicable patent and copyright notices related to content in this document is available on the Wowza website at www.wowza.com/legal.

Document History

Version	Description	Date
Doc v4.5	Document initial release of the Dynamic Load Balancing AddOn 4.5 for Wowza Streaming Engine	12-08-2017
Doc v4.5	Update the privateSpace.txt file name throughout for accuracy	02-23-2018
Doc v4.5	Updating the privateAddressMap.txt file name.	10-08-2018
Doc v4.5	Updated several property descriptions and examples.	12-20-2018
Doc v4.5	Added property to set the authentication type for the info page.	11-17-2021
Doc v4.5	Added properties to set redirect and info HTTP interfaces separately	10-07-2022

Table of Contents

Document History	7
Table of Contents	8
Overview	6
Deployment architecture options.....	8
Simple load balancing.....	8
Geographic group load balancing.....	8
Installing the Dynamic Load Balancing module	10
Configuring the load balancing system.....	11
Server listener configuration	11
VHost listener configuration.....	11
Load Balancer property configuration.....	12
Server property configuration	17
HTTP interfaces.....	22
Redirection.....	24
HTTP redirection.....	25
HTTP redirection considerations	27
RTMP and RTSP redirection	28
Host-specific information	29
JSON and XML output.....	29
Developer extension examples	31
Client data collection extension	31
Load Balancer decision processing extension.....	33
HTTP redirection extension	36
Examples.....	38
Example Server.xml file	38
Example privateSpace.txt file (Load Balancer).....	38

Overview

This document describes how to install and configure the Wowza Dynamic Load Balancing AddOn to enable geographic, bandwidth, and connection-based system load balancing between multiple servers running Wowza Streaming Engine software.

Note

Wowza Dynamic Load Balancing AddOn 4.5 is for use with Wowza Streaming Engine 4.7.3 and later. If you're running Wowza Streaming Engine 4.0 to 4.7.2, you must use [Dynamic Load Balancing AddOn 4.0](#). If you're running Wowza Media Server™, you must use [Dynamic Load Balancing AddOn 2.0](#) to enable system load-balancing.

Dynamic Load Balancing AddOn for Wowza Streaming Engine has the following functionality:

- Decision-based chaining based on geography, bandwidth, connections, private address space maps, and published stream locations can be specified in any order:
 - Each criteria can be checked until a destination Server is selected
- Geographic load balancing provided by MaxMind GeoIP2 and GeoIP Legacy:
 - Geographic Server grouping enables multiple Servers in the same geographic region to be further balanced
- Bandwidth usage-based load balancing:
 - Bandwidth usage across the entire Server with a resolution in kilobits per second (kbps)
 - Ability to exclude or include specific applications (case-sensitive)
- Connection-based load balancing:
 - Connections across the entire Server are taken into account
 - Ability to exclude or include specific applications (case-sensitive)
- Published stream location-based load balancing:
 - Ability to exclude or include specific applications (case-sensitive)
 - If not all edge Servers are used, only those to which the stream is published are used to find a valid destination
- Private address space location-based load balancing:
 - Custom address space map file (IPv4 only currently)
 - Ability to support multiple private address space segments per Server

- Client-based exclusions for direct connections to the Load Balancer server:
 - The Load Balancer can accept direct connections from publishers such as Adobe Flash Media Live Encoder, Telestream Wirecast, and so on
 - Local connections, such as live repeaters, are excluded by default from load balancing so that only client connections are used in decision making
- Soft shutdown of load-balancing Servers:
 - Prevent clients from being redirected to specific load-balancing Servers until they're restarted. Allows for connections to be dropped gracefully without needing to shut down a Server
- Basic Web interface for removing load-balancing Servers
 - Ability to change this behavior with the extensible HTTP interface for a custom version
- Automatic re-registration of load-balancing Servers with the Load Balancer:
 - If the Load Balancer server must be restarted, the load-balancing Servers re-register automatically
- Support for specific server name/port, XML and RTMP redirect for live and VOD streams:
 - XML-based redirection for JW Player and Flowplayer
 - HTTP redirection for HTTP-aware clients
 - RTMP redirection for JW Player and other Flash-based clients
 - RTSP redirection for VLC media player and other RTP-based clients
 - MPEG-DASH redirection
 - Specific hostname/port for integration elsewhere
 - Generic JSON/XML server outputs
- Ability to implement custom processing using the extensible API
- Support for Apple HLS, Adobe HDS, Silverlight, MPEG-DASH, RTMP, and RTSP
 - Ability to add HTTP protocols using the extensible API
- Secure communication between Load Balancer and load-balancing Servers
- Support for complete Server or VHost implementation
- Support for multiple load-balancing Server destinations for clients
- Support for private address space communication
- Automatic configuration of all HTTP, RTMP, and RTSP interfaces
 - Ability to disable the automatic configuration of the interfaces

Deployment architecture options

Simple load balancing

The simplest load-balancing configuration is to have one Wowza Streaming Engine server that acts as a load balancer (the "Load Balancer" server) and then any number of other Wowza servers that act as load-balancing edges (the load-balancing "Servers"). Requests to the Load Balancer are redirected to one of the connected Servers. In this scenario, each load-balancing Server has an equal chance of being redirected to.

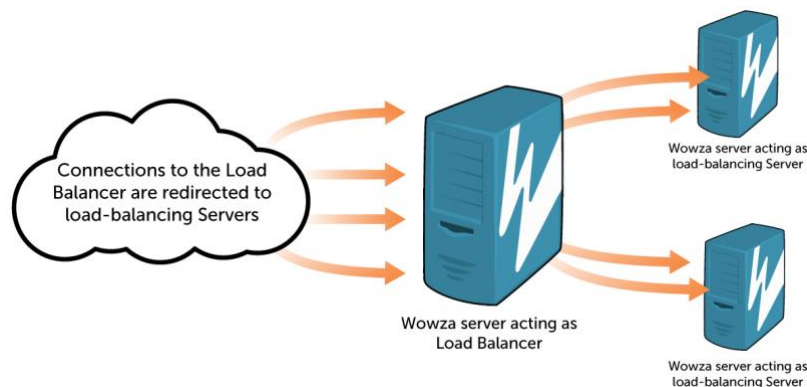


Figure 1 Diagram of simple load-balancing configuration.

Note that the Wowza Streaming Engine server that's configured as the Load Balancer can also act as a Server.

Geographic group load balancing

A more advanced load-balancing configuration option is to provide geographic load balancing with geographic grouping. This enables you to load balance multiple Wowza Streaming Engine servers in different geographic regions.

The following diagram shows two geographic groups of load-balancing Servers, a group in the United States ([loadbalanceClientCountryList](#) or [loadbalanceClientGeoIP2CountryList](#) = **US**) and another group in United Kingdom and other countries ([loadbalanceClientCountryList](#) or [loadbalanceClientGeoIP2CountryList](#) = **GB,***). A client in the United States that connects to the Load Balancer is redirected to the group of Servers in the US geographical region for load balancing, and then a specific Server is selected to fulfill the client request depending on the configuration. For example, if the configuration is based on the number of connections, the Server with the most available connections is selected to fulfill the client requests. Or if the configuration is based on available bandwidth, the Server that has the most available bandwidth is selected.

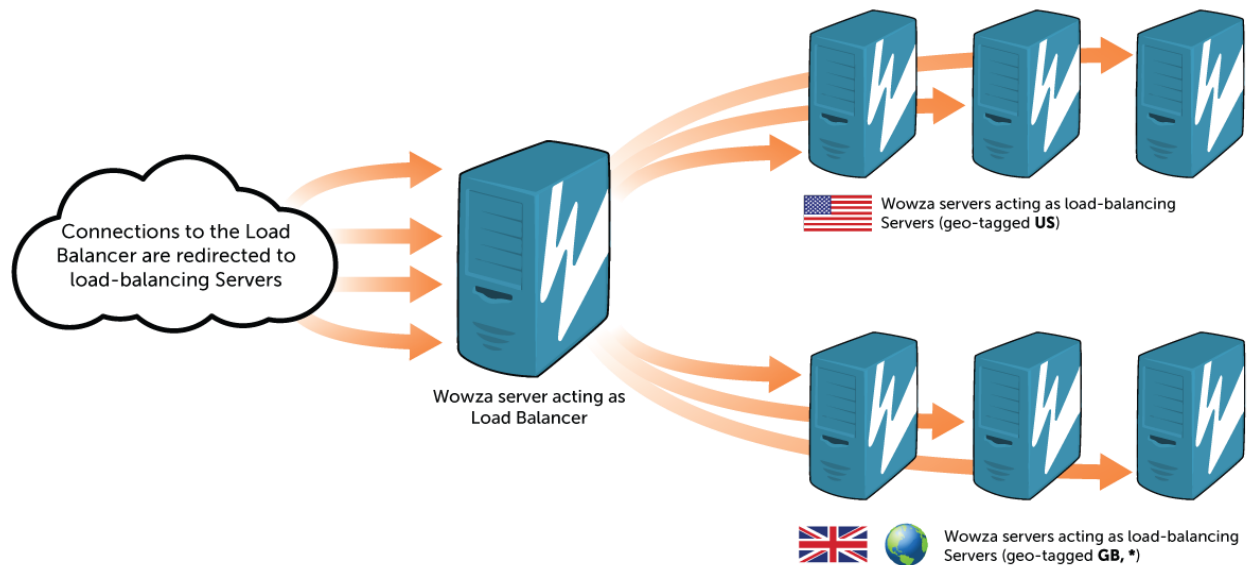


Figure 2 Diagram of geographic group load-balancing configuration.

All of the load-balancing Servers are registered with the Load Balancer and can be removed individually. The Wowza Streaming Engine server that's configured as the Load Balancer can also act as a Server and can't be removed from the configuration.

Note

To use geographic load balancing, you must download the GeoIP Legacy or GeoIP2 database(s) from [MaxMind, Inc.](#), and then configure the [loadbalanceClientCountryList](#), [loadbalanceClientGeoIP2CountryList](#), and/or [loadbalanceClientGeoIP2CityList](#) properties on the Load Balancer. The [legacy GeoIP Country](#) and GeoIP2 [Country](#) and [City](#) databases are available for one-time purchase or paid subscription. Less accurate versions of the legacy GeoIP and GeoIP2 databases (the [GeoLite Country](#) and [GeoLite2](#) databases) are available free-of-charge.

Installing the Dynamic Load Balancing module

Before you can install the Dynamic Load Balancing module, you must change your Wowza Streaming Engine configuration. The changes are very simple. The instructions in this section assume that the server software is installed at the following location (depending on your operating system):

Linux

/usr/local/WowzaStreamingEngine-[version]

Windows

/Program Files (x86)/Wowza Media Systems/Wowza Streaming Engine [version]

OS X

/Library/WowzaStreamingEngine-[version]

The above Wowza Streaming Engine installation paths for all operating systems are referred to as **[install-dir]** throughout this document.

Installation files

The installation files for the module are contained in a compressed (zipped) folder (**WowzaStreamingEngineLoadBalancer-[version].zip**). Download the file to each Wowza Streaming Engine server that you plan to use in the load-balancing configuration, and then use the following instructions to install:

1. Extract the files from the downloaded **WowzaStreamingEngineLoadBalancer-[version].zip** file.
2. Copy the contents of the unzipped folders to the corresponding folders in **[install-dir]**.

Configuring the load balancing system

A load-balancing system running on Wowza Streaming Engine software has the following configuration:

- [Server listener](#) or [VHost listener](#) (required for Load Balancer and all Servers)
- [Load Balancer Properties](#) (required for Load Balancer)
- [Server Properties](#) (required for all Servers)
- [HTTP interfaces](#) (required for Load Balancer)

Server listener configuration

To configure the server listener for the Dynamic Load Balancing AddOn, open the **[install-dir]/conf/Server.xml** file in a text editor and add the following server listener to the **<ServerListeners>** section in the file. This must be done on the Load Balancer and all Servers in the load-balancing topology:

```
<ServerListener>
  <BaseClass>com.wowza.wms.plugin.loadbalancer.listeners.ServerListener</BaseClass>
</ServerListener>
```

Note

The [Examples](#) section of this document shows sample **Server.xml** configurations for the Load Balancer in a load-balancing deployment.

VHost listener configuration

To configure the VHost listener for the Dynamic Load Balancing AddOn, open the **[install-dir]/conf/Server.xml** file in a text editor and add the following server listener to the **<VHostListener>** section in the file. This must be done on the Load Balancer and all servers in the load-balancing topology:

```
<VHostListener>
  <BaseClass>com.wowza.wms.plugin.loadbalancer.listeners.VHostListener</BaseClass>
</VHostListener>
```

Note

The [Examples](#) section of this document shows sample **Server.xml** configurations for the Load Balancer in a load-balancing deployment.

Load Balancer property configuration

You must specify properties in the **Server.xml** or **VHost.xml** file on the Load Balancer and all servers to configure the desired load-balancing functionality. If you are using the server listener, configure the properties in the **Server.xml** file. If you are using the VHost listener, we recommend configuring the properties in the **VHost.xml** file; if you use the **Server.xml** file, any identical properties in **VHost.xml** will override the **Server.xml** values.

To configure the properties, open the **.xml** file in a text editor and add the desired properties to the **<Properties>** section at the end of the file. The properties that can be set on the Load Balancer are described in the following table:

Notes:

- The [Examples](#) section of this document shows sample **Server.xml** configurations for the Load Balancer and Servers in a load-balancing deployment.
- Some of the properties in the following table require corresponding properties to be configured on the Servers.

Property Name	Description
loadbalanceType	Specifies the server type in the load-balancing configuration. The Load Balancer has the value Server or, if it will perform both roles, Server,Client . The load-balancing Servers have the value Client .
loadbalanceKey	Enables encrypted communication between the Load Balancer and Servers. The value must be 16 hcharacters long and must be the same on all servers in the configuration.
loadbalanceServerIP	IP address used for the Communications HTTP interface. The default value is [any] .
loadbalanceServerPort	Port used for the Communications HTTP interface. The default value is 1935
loadbalanceServerPath	Path used for the Communications HTTP interface. The default value is loadbalancerinterface .
loadbalanceServerRedirectIP	IP address used for the Redirection HTTP interface. The default value is [any]
loadbalanceServerRedirectPort	Port used for the Redirection HTTP interface. The default value is 1935
loadbalanceServerRedirectPath	Path used for the Redirection HTTP interface. The default value is redirect .

loadbalanceServerInfoIP	IP address used for the Information HTTP interface. The default value is [any]
loadbalanceServerInfoPort	Port used for the Information HTTP interface. The default value is 1935
loadbalanceServerInfoPath	Path used for the Information HTTP interface. The default value is loadbalanceinfo .
loadbalanceAdminAuthenticationMethod	Sets the authentication method to use with the loadbalanceinfo HTTP interface. Valid values are admin-basic , admin-digest , and admin-file-digest . The default value is admin-basic .
loadbalanceServerListenApplicationNames	A comma-separated list of applications that are automatically configured for RTMP/RTSP redirection when they start. Application redirection is used by default. Set the value to an empty string to turn off auto-configuration. Note: Listing an application name in this value doesn't create the application. You must create the applications separately.
loadbalanceServerIgnoreClients	A pipe-separated list of client identifiers that are used to determine if an RTMP connection is a publisher. Connections determined to be publishers aren't redirected. The default value for Wowza Streaming Engine 4.5 and later is: Wirecast/ FME/ FMLE/ Wowza GoCoder* Lavf/ UA Teradek/ KulaByte/ VidBlaster/ XSplit/ PES A. Note: Updates to Wowza Streaming Engine media server software may result in changes to this list. For more information, see Flash Version String .
loadbalanceServerDecisionOrder	Controls the order in which decision-making is processed for load balancing. Some or all of the following values can be entered as a comma-separated list: Geographic – Select load-balancing Servers based on the geographic selector using the legacy MaxMind GeoIP database format.

	<p>GeoIP2Country – Select load-balancing Servers based on the geographic country selector using the MaxMind GeoIP2 country database format.</p> <p>GeoIP2City – Select load-balancing Servers based on the geographic city selector using the MaxMind GeoIP2 city database format.</p> <p>PrivateAddressSpace – Select load-balancing Servers based on a custom address space map selector using a text file address or tag format.</p> <p>Connection – Select load-balancing Servers based on the bandwidth selector.</p> <p>Bandwidth – Select load-balancing Servers based on the connection selector.</p> <p>PublishedStreams – Select load-balancing Servers based on the published stream location selector.</p> <p>For example, to specify bandwidth-based and then connection-based load balancing, the value is: Bandwidth,Connection</p>
loadbalanceServerMaxMindDataFile	<p>The full path to the legacy MaxMind GeoIP Country database (GeoIP.dat) that's used for Geographic balancing. The default value is [install-dir]/conf/geoip.dat.</p> <p>Note: You must download the legacy GeoIP country database file from MaxMind to use geographic load balancing.</p>
loadbalanceServerMaxMindGeoIP2CountryFile	<p>The full path to the MaxMind GeoIP2 Country database (GeoIP2-Country.mmdb or GeoLite2-Country.mmdb) that's used for country balancing (GeoIP2Country). The default value is [install-dir]/conf/[filename].mmdb.</p> <p>Note: You must download the GeoIP2 or GeoLite2 country database file from MaxMind, Inc. to use country balancing. For more information, see Geographic group load balancing.</p>

loadbalanceServerMaxMindGeoIP2CityFile	<p>The full path to the MaxMind GeoIP2 City database (GeoIP2-City.mmdb or GeoLite2-City.mmdb) that's used for city balancing (GeoIP2City). The default value is [install-dir]/conf/[filename].mmdb.</p> <p>Note: You must download the GeoIP2 or GeoLite2 city database file from MaxMind, Inc. to use city balancing. For more information, see Geographic group load balancing.</p>
loadbalanceServerPrivateAddressMapFile	<p>The full path to the private address map file (privateSpace.txt) that's used for private address space balancing. The default location is [install-dir]/privateSpace.txt.</p> <p>Note: You must create a map file named privateSpace.txt file to use private address space balancing. For more information, see Example privateSpace.txt.</p>
loadbalanceServerProcessClasses	<p>A pipe-separated list of classes to load for server decision-making. When this property is set, it overrides all default decision classes. If any of the classes are not found, it is reported but won't stop the Load Balancer from starting.</p>
loadbalanceServerProcessAddClasses	<p>A pipe-separated list of classes to be loaded in addition to the default list for decision making. If any of the classes are not found, it is reported by won't stop the Load Balancer from starting.</p>
loadbalanceHTTPOutputClasses	<p>A pipe-separated list of classes that are loaded in addition to the default HTTP handling classes. This allows you to change the way different HTTP requests are handled.</p>
loadbalanceDebug	<p>(Optional) Controls all debug logging. This should be set to False unless you're having problems and can't identify a narrower scope. Whenever possible, we recommend using the more specific debug properties.</p>
loadbalanceDebugDecrypt	<p>(Optional) Controls decryption debug logging, which helps you to see how content was decrypted. This should be set to False unless you're having problems.</p>

loadbalanceDebugEncrypt	(Optional) Controls encryption debug logging, which helps you to see how content was encrypted. This should be set to False unless you're having problems.
loadbalanceDebugClientCommunications	(Optional) Controls debug logging of all Server communications with the Load Balancer. This should be set to False unless you're having problems.
loadbalanceDebugHTTPExtensions	(Optional) Controls debug logging of HTTP requests through extensions. This should be set to False unless you're having problems.
loadbalanceDebugClientExtension	(Optional) Controls Server extension activity debug logging. This should be set to False unless you're having problems.
loadbalanceDebugServerExtension	(Optional) Controls Load Balancer extension activity debug logging. This should be set to False unless you're having problems.
loadbalanceDebugServerDecision	(Optional) Controls decision-making debug logging, which helps you to see how the Load Balancer obtained a result. Enabling this property can produce a lot of log entries, which can create very large log files. This should be set to False unless you're having problems.
loadbalanceDebugClientProcess	(Optional) Controls client-processing debug logging. This should be set to False unless you're having problems.
loadbalanceDebugURLRequest	(Optional) Controls debug logging of HTTP requests made from clients to the Load Balancer. This should be set to False unless you're having problems.

Server property configuration

You must specify properties in the **Server.xml** or **VHost.xml** file on the Load Balancer and all Servers to configure the desired load-balancing functionality. If you are using the server listener, configure the properties in the **Server.xml** file. If you are using the VHost listener, we recommend configuring the properties in the **VHost.xml** file; if you use the **Server.xml** file, any identical properties in the **VHost.xml** file will override the **Server.xml** values.

To configure the properties, open the **.xml** file in a text editor and add the desired properties to the **<Properties>** section at the end of the file. The properties that can be set on the Servers are described in the following table:

Note

Some of the properties in the following tables require corresponding properties to be configured on the Load Balancer.

Property Name	Description
loadbalanceType	Specifies the server type in the load-balancing configuration. The load-balancing server has the value Client , or if it will perform both roles, Server,Client .
loadbalanceServerIP	IP address of the Load Balancer. To allow the Server to connect to multiple Load Balancers, use a comma-separated list. If you're using SSL, you must specify a hostname; an IP address won't work with an SSL certificate.
loadbalanceServerPath	URL path used for communication between the Load Balancer and the Servers. The value must be the same on all load-balancing Servers and must match the loadbalanceServerListenPath on the Load Balancer. To allow a Server to connect to multiple Load Balancers, use a comma-separated list.
loadbalanceServerPort	The port the Server uses to connect to the Load Balancer. The default value is 1935 . To allow a Server to connect to multiple ports, use a comma-separated list. If you're using SSL, you must use port 443 .
loadbalanceClientName	Name of the Server. If a name isn't specified, a random name is created each time the Server starts up.

loadbalanceKey	Enables encrypted communication between the Load Balancer and Servers. The value must be 16 hexadecimal characters long and must be the same on all servers in the configuration.
loadbalanceClientForceIP	Controls the IP address (or fully qualified domain name) to which playback clients are redirected when this load-balancing Server is used. This can be useful when there are multiple interfaces on a server and communications occurs over an interface that should not be used for playback traffic. If you're using SSL, you must set this to the fully qualified domain name that is specified as wildcard information in the SSL certificate installed on the load-balancing Server.
loadbalanceClientForcePort	Controls the port to which playback clients are redirected when this load-balancing Server is used. This can be useful if the load-balancing Servers don't use the same port as the Load Balancer. If you're using SSL, you must configure this property.
loadbalanceClientForceScheme	Controls the scheme that is returned to playback clients. The default scheme is http . If you're using SSL, you must set this to https .
loadbalanceClientCommunicationScheme	Controls the scheme that is used for communication between the load-balancing Servers and the Load Balancer. The default scheme is http . If you're using SSL, you must set this to https .
For Geographic balancing:	
loadbalanceClientCountryEnable	Controls legacy MaxMind GeoIP country balancing . The default value is off . To enable country balancing, set this value to on and configure the loadbalanceCountryList property on the load-balancing Servers.
loadbalanceClientCountryList	A comma-separated list of the MaxMind ISO 3166 Country Codes that this Server supports for legacy GeoIP country balancing. At least one country code must be specified. Use a * to specify that all country codes are supported.

For GeoIP2Country balancing:	
loadbalanceClientGeoIP2CountryEnable	Controls MaxMind GeoIP2 country load balancing . The default value is off . To enable country load balancing, set this value to on and configure the loadbalanceClientGeoIP2CountryList property on the load-balancing Servers.
loadbalanceClientGeoIP2CountryList	A comma-separated list of the MaxMind ISO 3166 Country Codes that this Server supports for GeoIP2 country balancing. At least one country code must be specified. Use a * to specify that all country codes are supported.
For GeoIP2City balancing:	
loadbalanceClientGeoIP2CityEnable	Controls MaxMind GeoIP2 city load balancing . The default value is off . To enable city load balancing, set this value to on and configure the loadbalanceClientGeoIP2CityList on the load-balancing Servers.
loadbalanceClientGeoIP2CityList	<p>A comma-separated list of the cities that this Server supports for GeoIP2 city load balancing. At least one city must be specified. Use a * to specify that all cities are supported.</p> <p>Note: When using GeoIP2City load balancing, we recommend that you also use GeoIP2Country balancing because multiple countries can contain cities with the same name.</p>
For PrivateAddressSpace balancing:	
loadbalanceClientPrivateAddressMapEnable	Controls private address space load balancing . The default value is off . To enable private address space load balancing, set this value to on and configure the loadbalanceClientPrivateAddressTagList property on the load-balancing Servers.
loadbalanceClientPrivateAddressTagList	<p>A comma-separated list of tags that the Server supports. At least one tag must be specified.</p> <p>Note: Tags are defined in the privateSpace.txt file. See Example privateSpace.txt for more information.</p>

For Connection and Bandwidth balancing:	
loadbalanceApplicationsExclude	A comma-separated list of applications to exclude from bandwidth and connection load-balancing calculations. The application names must match exactly; wildcards and partial names aren't supported. Applications listed here override the same applications listed in loadbalanceApplicationsInclude .
loadbalanceApplicationsInclude	A comma-separated list of applications to include in bandwidth and connection load-balancing calculations. Leaving this value blank includes all applications on the Server automatically. The application names must match exactly; wildcards and partial names aren't supported. Any applications specified in loadbalanceApplicationsExclude override this setting.
For Connection balancing:	
loadbalanceClientConnectionEnable	Controls connection balancing . The default value is off . To enable connection balancing, set this value to on and configure the loadbalanceClientConnectionLimit property on the load-balancing Servers.
loadbalanceClientConnectionLimit	<p>The limit for outgoing connections on this Server when using connection load balancing. To allow an unlimited number of connections, set to 0.</p> <p>If all Servers in the connection load-balancing deployment are servicing the maximum number of connections, redirections will fail and you must add an additional load-balancing Server to increase capacity.</p>
loadbalanceClientCountSourceClient	Enables you to include connections made by the Load Balancer, such as receiving streams from an origin server or an IP camera, in the connection count for connection load balancing. The default value is False , which excludes all remote connections from the count. To include remote connections in the count, set to True .

For Bandwidth balancing:	
loadbalanceClientBandwidthEnable	Controls bandwidth monitoring . The default value is off . To enable bandwidth monitoring, set this value to on and configure the loadbalanceClientBandwidthLimit property on load-balancing Servers.
loadbalanceClientBandwidthLimit	The bandwidth limit for the Server, in kilobits per second (Kbps). For example, if you want this Server to provide only 50 megabits per second (Mbps) of throughput, set the value to 50000 . To allow unlimited bandwidth, set to 0 . If all Servers in the bandwidth load-balancing deployment are saturated, redirection will fail and you must add an additional load-balancing Server to increase capacity.
loadbalanceClientBandwidthSourceClient	Enables you to include connections made by the Load Balancer, such as receiving streams from an origin server or an IP camera, in the bandwidth count for bandwidth load balancing. The default value is False , which excludes all remote connections from the count. To include remote connection in the bandwidth count, set to True .
For PublishedStreams balancing:	
loadbalanceClientPublishStreamsEnable	Controls published stream load balancing . The default value is off . To enable published stream balancing, set this value to on .
loadbalanceApplicationInstanceNameInclude	Enables the Server to include the application instance name in the published stream data sent to the Load Balancer. The default value is False . To include the application instance name, set this value to True .
loadbalanceServerPublishPrefixPassThrough	Enables requests for stream names with the SMIL, NGRP, or AMLST prefixes to be excluded from published stream decision-making. The default value is True . Setting the value to False might cause responses to fail when the appropriately-named stream same group isn't considered a published stream.

HTTP interfaces

To enable communication between the Load Balancer and Servers, and to view Server statistics in a web-based administration interface, HTTP Providers are automatically created:

- [Statistics and control HTTP interface](#)
- [Redirection interface for XML and specific host information](#)

Statistics and control HTTP interface

A statistics interface is configured automatically and enables you to see connected Servers and the bandwidth and countries allocated for them. It also enables you to shut down Servers, removing them from the configuration while leaving other connected Servers unaffected.

When the Load Balancer is on, you can view the connected Servers and their status in a web browser using the **loadbalanceServerListenerIP**, **loadbalanceServerListenerPort**, and the **loadbalanceServerListenInfoPath** properties in the following format:

```
http://[loadbalanceServerListenerIP]:[loadbalanceServerListenerPort]/[loadbalanceServerListenInfoPath]
```

For example, if the Load Balancer has the IP address of **192.168.1.1**, the port **1935**, and the path **loadbalanceinfo**, you can open the following address in the browser:

```
http://192.168.1.1:1935/loadbalanceinfo
```

The web-based interface includes **Pause Client** and **Remove Client** buttons for each Server entry.

Pause Client

When you click **Pause Client**, the Server is temporarily excluded from load-balancing decisions but it will continue to check in. The following is an example of the entry log created when you click **Pause Client**:

```
INFO server comment - loadBalancer: Client: Pause: fa5bb359-8e1f-4fed-8e8c-d50d4180d88d
```

After you click **Pause Client**, the button changes to **Unpause**, which immediately re-includes the client in load balancing decision-making when clicked. The following is an example of a log entry that is created when you click **Unpause**:

```
INFO server comment - loadBalancer: Client: Unpause: fa5bb359-8e1f-4fed-8e8c-d50d4180d88d
```

Remove Client

When you click **Remove Client**, the Server is removed from the load-balancing configuration without shutting it down. The following is an example of a log entry that is created when you click **Remove Client**:

```
INFO server comment - loadBalancer: Client: Stopped: eabdc97-3989-4622-a5df-04d34193558a
```

Clicking **Remove Client** immediately removed the client from load balancing decision-making and stops the client from checking in, which eventually causes the Server to expire. The following is an example of a log entry that is created when a Server expires:

```
INFO server comment - loadBalancer: Remove client: Expire Timeout: eabdc97-3989-4622-a5df-04d34193558a
```

Redirection interface for XML and specific host information

A redirection interface enables you to return load-balancing information that can be integrated into well-known Flash clients (such as JW Player) or with your own application returning only the IP/port that's used. This interface is configured automatically and uses the `loadbalanceServerListenRedirectPath` property.

Redirection

The load-balancing redirection interfaces on the Load Balancer enable you to provide a single URL to clients in the following basic form:

```
[protocol]://[load-balancer-ip-address]:1935/redirect/[application-name]/[stream-name]?type=[http-transmission-type]
```

The **[load-balancer-ip-address]** is the IP address of the Wowza Load Balancer. The *type* URL query is used to specify the manifest file format in the redirect URL that's returned to clients and is only used for HTTP redirection.

A client is then redirected to an available load-balancing Server via a URL that has the following basic form:

```
[protocol]://[load-balanced-server-ip-address]:1935/[application-name]/[stream-name]/[manifest]
```

Notice that the **/redirect/** part of the request URL is removed in the resulting redirect URL. You can configure your filter to be any name and it too will be removed appropriately. The **[load-balanced-server-ip-address]** is either the IP address of the Wowza Load Balancer or the IP address of a load-balancing Server, depending on the load on the system.

If you use `wget` and use the `--server-response` and set `max-redirect` to 0 you can see the headers returned. For example, this command:

```
wget --max-redirect 0 --server-response http://[wowza-ip-address]:1935/redirect/vod/stream.mp4?type=m3u8
```

returns the following information:

```
HTTP/1.1 302 Found
Access-Control-Expose-Headers: Date, Server, Content-Type, Content-Length
Access-Control-Allow-Headers: Content-Type, User-Agent, If-Modified-Since, Cache-Control, Range
Access-Control-Allow-Origin: *
Location: http://[wowza-client-ip-address]:1935/vod/stream.mp4/playlist.m3u8
Access-Control-Allow-Methods: OPTIONS, GET, POST, HEAD
Access-Control-Allow-Credentials: true
Content-Type: text/html
Connection: Keep-Alive
Server: WowzaStreamingEngine/9.9.9.99
Cache-Control: no-cache
Content-Length: 0
```

You can see the request for the application **vod** and stream name **stream.mp4** with the type **m3u8** this then generated a redirect of

```
http://[wowza-client-ip-address]:1935/vod/stream.mp4/playlist.m3u8
```

The address changes based on the load reported by each server and the configuration of the decisions set.

Notes

- When the [loadbalanceType](#) property value is set to **Server,Client** on the Load Balancer, it's also a load-balancing Server that can fulfill the client request if not overloaded. Setting this property value to **Server** means that the Load Balancer redirects all client requests to connected load-balancing Servers.
- When using a load-balancing system to deliver relative playlists, particularly with Apple HLS, some playback clients may not follow the redirection correctly. Therefore, it is important to configure absolute playlists for all load-balancing edge servers. For more information, see [Switch between absolute and relative URLs in HLS playlists in Wowza Streaming Engine](#).

A Wowza Streaming Engine load-balancing system supports redirection over the following protocols:

- [HTTP](#)
- [RTMP and RTSP](#)

HTTP redirection

The HTTP redirection interface enables you to provide a single URL that redirects client requests to an available load-balancing Server. The HTTP redirection supports most clients by enabling [M3U8](#), [F4M](#), [Manifest](#), [MPEG-DASH](#), and [RTMP](#) XML outputs to be created. All of the redirection types work for live and VOD applications configured in Wowza Streaming Engine software.

HTTP clients are redirected using the HTTP 302 response status code. Your client must support the HTTP 302 code for load balancing to work. RTMP clients that use HTTP requests must support the returned XML (see [RTMP XML](#)).

Note

Cross-origin resource sharing (CORS) headers are now supported by default on all HTTP Providers used by Wowza Streaming Engine, and can't be configured through the load balancer.

Apple HLS (M3U8)

The following is an example for generating an HTTP 302 (redirection) response with a redirection URL in the Location header field for use in some iOS-based devices:

```
http://[load-balancer-ip-address]:1935/redirect/[application-  
name]/[stream-name]?type=m3u8
```

In this example, Apple HLS clients are redirected to:

```
http://[load-balanced-server-ip-address]:1935/[application-  
name]/[stream-name]/playlist.m3u8
```

Adobe HDS (F4M)

The following is an example for generating an HTTP 302 (redirection) response with a redirection URL in the Location header field for use in some Flash-based players:

```
http://[load-balancer-ip-address]:1935/redirect/[application-  
name]/[stream-name]?type=F4M
```

In this example, Adobe HDS clients are redirected to:

```
http://[load-balanced-server-ip-address]:1935/[application-  
name]/[stream-name]/manifest.f4m
```

Microsoft Smooth Streaming (Manifest)

The following is an example for generating an HTTP 302 (redirection) response with a redirection URL in the Location header field for use in some Silverlight clients:

```
http://[load-balancer-ip-address]:1935/redirect/[application-  
name]/[stream-name]?type=Manifest
```

In this example, Silverlight clients are redirected to:

```
http://[load-balanced-server-ip-address]:1935/[application-  
name]/[stream-name]/Manifest
```

MPEG-DASH (MPD)

The following is an example for generating an HTTP 302 (redirection) response with a redirection URL in the Location header field for use in some MPEG-DASH clients:

```
http://[load-balancer-ip-address]:1935/redirect/[application-  
name]/[stream-name]?type=dash
```

In this example, MPEG-DASH clients are redirected to:

```
http://[load-balanced-server-ip-address]:1935/[application-  
name]/[stream-name]/manifest.mpd
```

You can also change the manifest type by using the *manifestType* query parameter. The default manifest type is **manifest.mpd**, but the the following types are also valid:

- manifest_mpm4sav_mvlist.mpd
- manifest_mvlist.mpd
- manifest_mpm4sav_mvtime.mpd
- manifest_mvtime.mpd
- manifest_mpm4sav_mvnumber.mpd

- manifest_mvnumber.mpd.

In this example, MPEG-DASH clients are redirected with a segment list manifest (manifest_mvlist.mpd):

```
http://[load-balancer-ip-address]:1935/redirect/[application-name]/[stream-name]?type=dash&manifestType=manifest_mvlist.mpd
```

HTTP redirection considerations

When configuring the load balancer for Apple HLS playback it may be required to enable absolute URLs.

When the load balancer redirects a playback client to the appropriate server in some cases it must remain an absolute URL, rather than relative, to enable successful playback.

The following Wowza tutorial outlines how to enable absolute URLs and to avoid confusion should be configured on all the servers in use for the load balancing implementation.

<https://www.wowza.com/docs/how-to-switch-between-absolute-and-relative-urls-in-apple-http-live-streaming-playlists>

RTMP XML (RTMP)

The following is an example for generating XML output for use in some Flash-based players:

```
http://[load-balancer-ip-address]:1935/redirect/[application-name]/[stream-name]/loadbalancersmil
```

In this example, an RTMP client using HTTP requests gets XML returned similar to:

```
<?xml version="1.0"?>
<smil>
  <head>
    <meta base="rtmp://[load-balanced-server-ip-address]:1935/[application-name]/" />
  </head>
  <body>
    <switch>
      <video src="[stream-name]" />
    </switch>
  </body>
</smil>
```

In the above example, a client connects to the Load Balancer and the original URL parameters are used to construct the application name and stream to be used. If you specify a longer directory structure, this is added to the **src** element in the XML file.

You can also generate a multi-bitrate version of this output to allow redirection for adaptive bitrate (ABR) streams. To do this, the URL requires additional query parameters that describe the assets and bitrates. These should contain a comma-separated list of

assets and their corresponding bitrates. If the number of assets and bitrates doesn't match, no output is returned.

The following is an example for generating XML output for use in some Flash-based players:

```
http://[load-balancer-ip-address]:1935/redirect/[application-  
name]/loadbalancer.smil?assets=file1.mp4,file2.mp4,file3.mp4&bitrates=25  
0000,350000,500000
```

In this example, an RTMP client using HTTP requests gets XML returned similar to:

```
<?xml version="1.0"?>  
<smil>  
  <head>  
    <meta base="rtmp://54.194.186.4:80/vod/_definst_" />  
  </head>  
  <body>  
    <switch>  
      <video src="file1.mp4" system-bitrate="250000" />  
      <video src="file2.mp4" system-bitrate="350000" />  
      <video src="file3.mp4" system-bitrate="500000" />  
    </switch>  
  </body>  
</smil>
```

Any additional query string parameters added to the HTTP request are also appended to the file names presented in the XML file. This can be useful when using the [SecureToken](#) feature in Wowza Streaming Engine (version 4.1 and later).

RTMP and RTSP redirection

To enable RTMP redirection functionality to the Load Balancer, [configure the load-balancing system](#) and add an application named **redirect**. By default, the Load Balancer enables this application for RTMP and RTSP redirection. You can use the [loadbalanceServerListenApplicationNames](#) property to disable the auto-configuration or change the application(s).

When auto-configuration is enabled, no changes are made to the **Application.xml** file. However, if you disable auto-configuration of RTMP and RTSP redirection, you must add the **Redirect** module to the **[install-dir]/conf/[application-name]/Application.xml** file as the last module in the **<Modules>** section. The following example shows the **Redirect** module:

```
<Module>  
  <Name>Redirect</Name>  
  <Description>Redirect</Description>  
  <Class>com.wowza.wms.plugin.loadbalancer.module.ClientConnections</Cla  
ss>  
</Module>
```

This module returns an RTMP or RTSP redirect to clients that connect to the Load Balancer, except for those that are configured to be ignored in the

[loadbalanceServerIgnoreClients](#) property in the **Server.xml** file. Wowza edge servers that connect to a Load Balancer that's also an origin server in a live stream repeater (origin/edge) configuration are also ignored.

The following are examples of the URLs used to redirect:

RTMP clients:

```
rtmp://[load-balancer-ip-address]:1935/redirect/[application-name]/
```

In this example, RTMP clients are redirected to:

```
rtmp://[load-balanced-server-ip-address]:1935/[application-name]/
```

RTSP clients:

```
rtsp://[load-balancer-ip-address]/redirect/[application-name]/[stream-name]
```

In this example, RTSP clients are redirected to:

```
rtsp://[load-balanced-server-ip-address]:1935/[application-name]/[stream-name]
```

Host-specific information

To determine the IP address and port of a load-balancing Server to use for any client request, you can request this information by using the following URL:

```
http://[load-balancer-ip-address]:1935/redirect/?type=client&IP=A.B.C.D
```

You can use the *IP* query parameter to specify the IP address of a potential client to see which load-balancing Server it would be redirected to. If you omit the query parameter, the IP address of the local client is used.

This will return a text string with the *IP address:port* combination, for example:

```
[load-balanced-server-ip-address]:1935
```

You can also use the *streamName* query parameter to specify the name of a stream you are looking for.

JSON and XML output

Additionally, you can request JSON and XML output when you query the load balancer interface by adding the *type* query parameter to the queries.

JSON

To request JSON output, include the *type* query parameter with a value of **json**. For example, the following query:

```
http://[load-balancer-ip-address]:1935/redirect/vod/sample.mp4?type=json
```

returns JSON output similar to the following:

```
{
  "loadbalanceVersion": "4.5.0.0",
  "loadbalanceBuild": "14598",
  "loadbalanceScheme": "http",
  "loadbalanceHost": "192.168.1.5:1935",
  "loadbalanceAssetPath": "/vod/sample.mp4",
  "loadbalanceClientQueryString": "",
  "loadbalanceClientIP": "192.168.1.10"
}
```

The output includes the load balancer version, build number, and the load-balancing Server. It also returns the IP address determined from the client request. However, you can override the IP address by using the *IP* query parameter; see [Host-specific information](#) for more information.

XML

To request XML output, include the *type* query parameter with a value of **xml**. For example, the following query:

```
http://[load-balancer-ip-address]:1935/redirect/vod/sample.mp4?type=xml
```

returns XML output similar to the following:

```
<?xml version="1.0" encoding="UTF-8"?>
<loadbalance>
  <server>
    <version>4.5.0.0</version>
    <build>14598</build>
  </server>
  <client>
    <host>192.168.1.5:1935</host>
    <queryString></queryString>
    <clientIP>192.168.1.10</clientIP>
    <assetPath>/vod/sample.mp4</assetPath>
  </client>
</loadbalance>
```

The output includes the load balancer version, build number, and the load-balancing Server. It also returns the IP address determined from the client request. However, you can override the IP address by using the *IP* query parameter; see [Host-specific information](#) for more information.

Developer extension

examples

The Dynamic Load Balancing AddOn 4.5 enables customers to develop and deploy their own extensions to provide functionality specific to their environment. There are four types of extensions: client data collection, server decision processing, communication encryption, and HTTP redirection.

Client data collection extension

The client data collection extension is defined by the **ILoadBalancerClientDataExtension**, which is found in the **com.wowza.wms.plugin.loadbalancer.client** package. This interface enables you to collect information you need, send this information to the load-balancing Server, which in turn sends the information to the Load Balancer. The following example shows a client data collection extension:

```
package my.domain.test.client.extension;

import java.util.HashMap;
import java.util.Iterator;
import java.util.List;
import java.util.Map;

import com.wowza.wms.application.WMSProperties;
import com.wowza.wms.logging.WMSLoggerFactory;
import com.wowza.wms.plugin.loadbalancer.client.ILoadBalancerClientDataExtension;
import com.wowza.wms.plugin.loadbalancer.utils.Logger;
import com.wowza.wms.server.IServer;
import com.wowza.wms.vhost.IVHost;

public class LoadBalancerClientDummyExtension implements
ILoadBalancerClientDataExtension
{
    // The COMPONENET_NAME must be unique, if an extension with the same
    // name is already loaded
    // the extension is not loaded
    private static final Class<LoadBalancerClientDummyExtension> CLASS =
LoadBalancerClientDummyExtension.class;
    private final String COMPONENT_NAME = "ClientExtensionDummy";
    private boolean debug = false;
    private boolean active = false;
    private Logger logger = null;

    public void init(WMSProperties props, IServer iServer, List<IVHost>
vHosts)
    {
        // The props contain the properties from the Server or VHost. This
        // allows you to use your own
        // custom properties for your own extensions.
    }
}
```

```
// The iServer value is null if running as a VHost listener

// A list of VHosts is passed into the List VHosts. If the server is
configured as
// a VHost listener then only that VHost is present in the list.

// The init is called when the extension is loaded

Iterator<IVHost> useableVHosts = vHosts.iterator();
while (useableVHosts.hasNext())
{
    IVHost thisVhost = useableVHosts.next();
    if (getDebug() == true)
    {
        WMSLoggerFactory.getLogger(CLASS).info(CLASS + " Doing Vhost " +
thisVhost.getName());
    }
}

public void setDebug(boolean debug)
{
    this.debug = debug;
}

public boolean getDebug()
{
    return this.debug;
}

public void setActive(boolean active)
{
    this.active = active;
}

public boolean getActive()
{
    return this.active;
}

public void setLogger(Logger log)
{
    this.logger = log;
}

public Logger getLogger()
{
    return this.logger;
}

public String getComponentName()
{
    return COMPONENT_NAME;
}

public Map<String, String> processClientInformation(WMSProperties
props, IServer iServer, List<IVHost> vHosts)
{

```

```
// This function is called each time the client is polls for
information. It must return a name/value map

Map<String, String> returnMap = new HashMap<String, String>();
// Do something here
returnMap.put("dummyName", "dummyInfo");

return returnMap;
}
}
```

The `COMPONENT_NAME` is used as a prefix for all of the names provided in the map as they pass through the Server to the Load Balancer. This maintains unique entries for all client extensions.

To use this extension, the `loadbalanceClientProcessAddClasses` property should be added to the `Server.xml` or `VHost.xml` file. The following is an example for configuring the property:

```
<Property>
  <Name>loadbalanceClientProcessAddClasses</Name>
  <Value>my.domain.test.client.extension.LoadBalancerClientDummyExtensio
n</Value>
</Property>
```

If you need to add multiple classes, use a pipe-separated (`|`) list.

Load Balancer decision processing extension

The Load Balancer decision processes extension is defined by the **`ILoadBalancerServerDataExtension`**, which is found in the **`com.wowza.wms.plugin.loadbalancer.server`** package. This interface enables you to process any information you need as a decision parameter and select registered Servers. The following example shows a Load Balancer decision process extension:

```
package my.domain.test.server.extension;

import java.util.HashMap;
import java.util.Iterator;
import java.util.Map;

import com.wowza.wms.application.WMSProperties;
import com.wowza.wms.logging.WMSLoggerFactory;
import com.wowza.wms.plugin.loadbalancer.utils.Logger;

public class LoadBalancerServerDummyExtension
implements ILoadBalancerServerDataExtension
{
    private boolean debug = false;
    private boolean active = false;
    private final String COMPONENT_NAME =
"LoadBalancerServerDummyExtension";
    private Logger logger = null;
```

```
public void init(WMSProperties props)
{
}

public void setDebug(boolean debug)
{
    this.debug = debug;
}

public boolean getDebug()
{
    return this.debug;
}

public void setLogger(Logger log)
{
    this.logger = log;
}

public Logger getLogger()
{
    return this.logger;
}

public void updateComponent(HashMap<String, String> data)
{
}

public void setActive(boolean active)
{
    this.active = active;
}

public boolean getActive()
{
    return this.active;
}

public String getComponentName()
{
    return COMPONENT_NAME;
}

public List<ILoadBalancerServerClient> processClientInformation(List
<ILoadBalancerServerClient> clientList, Map<String, String> params)
{
    // The processClientInformation function is run each time a client
    request is made for
    // load balancing.

    // clientList contains the clients available to be checked

    // params contains any custom parameters added by client requests
    when a request is
    // received.

    if (!getActive())
        return clientList;
}
```

```
List<ILoadBalancerServerClient> outputClients = new
ArrayList<ILoadBalancerServerClient>();

    Iterator<ILoadBalancerServerClient> clientI =
clientList.iterator();
    while (clientI.hasNext())
    {
        ILoadBalancerServerClient thisClient = clientI.next();
        HashMap<String, String> thisClientAtt =
thisClient.getAttributes();
        // The attribute list from the client contains ALL attributes
passed by ALL the client data collection extensions.
        // Each client extension prefixes the name of the extension

        if (thisClientAtt.containsKey("ClientExtensionDummy-dummyName"))
        {
            // Determine if this client matches criteria you are looking for
and add to the
            // return list of clients. This list is passed to the next
extension for processing.
            //
            // outputClients.put(thisClient, clientList.get(thisClient));
        }
    }

    return outputClients;
}
```

The COMPONENT_NAME must be unique for each extension loaded.

To use this extension, the **loadbalanceServerProcessAddClasses** property should be added to the **Server.xml** or **VHost.xml** file. The following is an example for configuring the property:

```
<Property>
    <Name>loadbalanceServerProcessAddClasses</Name>

    <Value>my.domain.test.server.extension.LoadBalancerServerDummyExtensio
n </Value>
</Property>
```

If you need to add multiple classes, use a pipe-separated (|) list.

HTTP redirection extension

The HTTP redirection extensions enable you to write your own extension to handle any type of request you wish to redirect. The following is an example of an HTTP redirection extension:

```
package my.domain.test.http.extension;

import java.util.HashMap;
import java.util.List;
import java.util.Map;

import com.wowza.wms.http.IHTTPRequest;
import com.wowza.wms.http.IHTTPResponse;
import
com.wowza.wms.plugin.loadbalancer.http.ILoadBalancerHTTPExtension;
import com.wowza.wms.plugin.loadbalancer.server.ILoadBalancerServer;
import
com.wowza.wms.plugin.loadbalancer.server.ILoadBalancerServerClient;
import com.wowza.wms.plugin.loadbalancer.utils.Logger;
import com.wowza.wms.vhost.IVHost;

public class LoadBalancerHTTPEasy implements ILoadBalancerHTTPExtension
{

    private boolean debug = false;
    private boolean active = true;
    private final String COMPONENT_NAME = "Easy";

    public void setDebug(boolean debug)
    {
        this.debug = debug;
    }

    public boolean getDebug()
    {
        return this.debug;
    }

    public void setActive(boolean active)
    {
        this.active = active;
    }

    public boolean getActive()
    {
        return this.active;
    }

    public String getComponentName()
    {
        return COMPONENT_NAME;
    }

    public String processHTTPRequest(IVHost inVhost, IHTTPRequest req,
    IHTTPResponse resp, String assetPath, ILoadBalancerServer
    loadBalancerServer)
    {
        Map<String, List<String>> requestParams = req.getParameterMap();
```

```
String IP = req.getRemoteAddr();
HashMap<String, String> params = new HashMap<String, String>();
// If a decision is needed based on the source IP , add the IP to
the params list being sent
// to the decision logic. As this is for M3U8 type we do not need to
override the request.
params.put("ClientIP", IP);
ILoadBalancerServerClient foundClient =
loadBalancerServer.processServerDataExtensions(params);
String output = "";
if (foundClient != null)
{
    // foundClient.getClientHost() does all the work for client IP or
forced IP and also port
    // removing the : if port 80 is provided
    output = req.getType() + "://" + foundClient.getClientHost() +
assetPath + "/playlist.m3u8";

    // You could set return headers for a redirect
    // resp.setHeader("Location", output);
    //resp.setResponseCode(302);
}
// If you want to return an output you do so here, or return null.
If redirecting returning null is important for some clients.
return output;
}
}
```

To use this extension, the **loadbalanceHTTPOutputClasses** should be added to the **Server.xml** or **VHost.xml** file:

```
<Property>
  <Name>loadbalanceHTTPOutputClasses</Name>
  <Value>my.domain.test.http.extension.LoadBalancerHTTPDummy</Value>
</Property>
```

If you need to add multiple classes, use a pipe-separated (|) list.

After creating the extension, there are two ways to access it:

You can use a type parameter. For example:

```
http://[load-balancer-ip-address]:1935/redirect/[application-
name]/[stream-name]?type=dummy
```

Or you can append a **.[suffix]** to the end of the URL. For example:

```
http://[load-balancer-ip-address]:1935/redirect/[application-
name]/[stream-name].dummy
```

Note

If Wowza Streaming Engine matches against an extension, such as m3u8, you must use the type parameter.

All the Java examples shown can be found in the `documentation/exampleConfigurations/JavaAPIExamples` folder.

Examples

Example Server.xml file

Server.xml files must be configured on the Load Balancer and all load-balancing Servers. Examples are available in `[install-dir]/documentation/exampleConfigurations` for each type of load balancing decision. Each example includes two files: **Server.txt** demonstrates the Load Balancer configuration and **Client.txt** demonstrates the load-balancing Server configuration.

Note

To use multiple types of load balancing, enter a comma-separated list of values for the **loadbalanceServerDecisionOrder** in the Load Balancer **Server.xml** file, and then make sure the properties for those types of load balancing are added to the **Server.xml** files on the Load Balancer and load-balancing Servers. For more information, see [Load Balancer property configuration](#) and [Server property configuration](#).

Example privateSpace.txt file (Load Balancer)

To balance load based on a custom address space map, you must create a new text file named **privateSpace.txt** that lists the private addresses and tags the Server should be used for. Usually, this file is located in the `[install-dir]`, but you can use the [loadbalanceServerFile](#) property to specify an alternate path. This map file should contain one entry per line and include a tag and the IP addresses.

Note

IPv6 addresses are not supported. You must use IPv4 or CIDR notation.

To use IPv4 notation, each entry must be formatted as a comma-separated list and contain a beginning IP address, an ending IP address, and a tag. For example:

```
192.168.0.0,192.168.0.63,HOME  
192.168.1.0,192.168.1.255,WORK
```

To use the CIDR notation, each entry must contain the CIDR address and a tag, separated by a comma. For example:

```
192.168.0.0/26,HOME  
192.168.1.0/24,WORK
```

The tags listed in the **privateSpace.txt** file must correspond to the value of the [loadbalanceClientPrivateAddressTagList](#) property on the Server.