

# Black Book

ixia

---

Edition 10

Video over IP



**Your feedback is welcome**

Our goal in the preparation of this Black Book was to create high-value, high-quality content. Your feedback is an important ingredient that will help guide our future books.

If you have any comments regarding how we could improve the quality of this book, or suggestions for topics to be included in future Black Books, please contact us at [ProductMgmtBooklets@ixiacom.com](mailto:ProductMgmtBooklets@ixiacom.com).

Your feedback is greatly appreciated!

Copyright © 2014 Ixia. All rights reserved.

This publication may not be copied, in whole or in part, without Ixia's consent.

RESTRICTED RIGHTS LEGEND: Use, duplication, or disclosure by the U.S. Government is subject to the restrictions set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.227-7013 and FAR 52.227-19.

Ixia, the Ixia logo, and all Ixia brand names and product names in this document are either trademarks or registered trademarks of Ixia in the United States and/or other countries. All other trademarks belong to their respective owners. The information herein is furnished for informational use only, is subject to change by Ixia without notice, and should not be construed as a commitment by Ixia. Ixia assumes no responsibility or liability for any errors or inaccuracies contained in this publication.



## Contents

How to Read this Book.....	vii
Dear Reader .....	viii
Video over IP .....	1
Getting Started Guide .....	13
Test Case: RTSP Media Server Testing.....	17
Test Case: Basic VoD Server Interoperability .....	29
Test Case: Large Scale RTP Video - 600,000 Streams.....	41
Test Case: Large Scale RTP Video - 4.8 Million Packets per Second .....	53
Test Case: Basic Flash™ Player Emulation .....	65
Test Case: Flash Player User Actions - PAUSE, RESUME.....	75
Test Case: Flash Player User Actions - PLAY and SEEK .....	79
Test Case: Flash Player User Actions - Playback of Multiple Media Files .....	83
Test Case: Flash™ Player Emulation - Secure streaming using RTMPE .....	85
Test Case: Flash™ Player Emulation - Secure streaming using RTMPTE .....	99
Test Case: Basic Adobe® HTTP Dynamic Streaming Client .....	111
Test Case: Adobe® HTTP Dynamic Streaming with Pre-selected or Automatic Adaptive Bitrate Shifting.....	123
Test Case: Basic Silverlight® Smooth Streaming Client.....	137
Test Case: Silverlight® Smooth Streaming with Pre-selected Bitrate Shifts .....	147
Test Case: Basic HTTP Live Streaming (HLS) Client.....	157
Test Case: HTTP Live Streaming with Pre-Selected Adaptive Bitrate Shifting .....	165
Annex A: Configuring IP and Network Settings .....	173
Annex B: Configuring TCP Parameters.....	175
Annex C: Configuring HTTP Servers.....	177
Annex D: Configuring HTTP Clients.....	179
Annex E: Setting the Test Load Profile and Objective .....	181
Annex F: Adding Test Ports and Running Tests.....	183
Contact Ixia.....	185



## How to Read this Book

The book is structured as several standalone sections that discuss test methodologies by type. Every section starts by introducing the reader to relevant information from a technology and testing perspective.

Each test case has the following organization structure:

<b>Overview</b>	Provides background information specific to the test case.
<b>Objective</b>	Describes the goal of the test.
<b>Setup</b>	An illustration of the test configuration highlighting the test ports, simulated elements and other details.
<b>Step-by-Step Instructions</b>	Detailed configuration procedures using Ixia test equipment and applications.
<b>Test Variables</b>	A summary of the key test parameters that affect the test's performance and scale. These can be modified to construct other tests.
<b>Results Analysis</b>	Provides the background useful for test result analysis, explaining the metrics and providing examples of expected results.
<b>Troubleshooting and Diagnostics</b>	Provides guidance on how to troubleshoot common issues.
<b>Conclusions</b>	Summarizes the result of the test.

## Typographic Conventions

In this document, the following conventions are used to indicate items that are selected or typed by you:

- **Bold** items are those that you select or click on. It is also used to indicate text found on the current GUI screen.
- *Italicized* items are those that you type.

## Dear Reader

Ixia's Black Books include a number of IP and wireless test methodologies that will help you become familiar with new technologies and the key testing issues associated with them.

The Black Books can be considered primers on technology and testing. They include test methodologies that can be used to verify device and system functionality and performance. The methodologies are universally applicable to any test equipment. Step-by-step Instructions using Ixia's test platform and applications are used to demonstrate the test methodology.

This tenth edition of the Black Books includes twenty-two volumes covering some key technologies and test methodologies:

**Volume 1** – Higher Speed Ethernet

**Volume 2** – QoS Validation

**Volume 3** – Advanced MPLS

**Volume 4** – LTE Evolved Packet Core

**Volume 5** – Application Delivery

**Volume 6** – Voice over IP

**Volume 7** – Converged Data Center

**Volume 8** – Test Automation

**Volume 9** – Converged Network Adapters

**Volume 10** – Carrier Ethernet

**Volume 11** – Ethernet Synchronization

**Volume 12** – IPv6 Transition Technologies

**Volume 13** – Video over IP

**Volume 14** – Network Security

**Volume 15** – MPLS-TP

**Volume 16** – Ultra Low Latency (ULL) Testing

**Volume 17** – Impairments

**Volume 18** – LTE Access

**Volume 19** – 802.11ac Wi-Fi Benchmarking

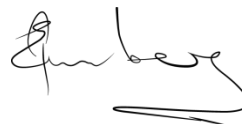
**Volume 20** – SDN/OpenFlow

**Volume 21** – Network Convergence Testing

**Volume 22** – Testing Contact Centers

A soft copy of each of the chapters of the books and the associated test configurations are available on Ixia's Black Book website at <http://www.ixiacom.com/blackbook>. Registration is required to access this section of the Web site.

At Ixia, we know that the networking industry is constantly moving; we aim to be your technology partner through these ebbs and flows. We hope this Black Book series provides valuable insight into the evolution of our industry as it applies to test and measurement. Keep testing hard.



Errol Ginsberg, Acting CEO



# Video over IP

## Test Methodologies

This Black Book series focuses on testing video delivery platforms used in IPTV and OTT services. The test cases outline how to use Ixia's IxLoad™ solution with several core streaming protocols, including RTSP, RTP, IGMP, multicast, Flash™ Player technology, Silverlight® Player technology, Apple® HLS, with advanced MPEG2, H.264, AAC, and related audio/video CODECs. The types of video services being tested include linear broadcast (LTV), video on demand (VOD), and HTTP-based adaptive bitrate technologies for Over the Top (OTT) on-demand and live streaming.

## Video over IP

Video delivery over IP can be broadly divided into the following two sets of technology:

- **Internet TV (IPTV).** IPTV uses a converged IP service delivery network to deliver television services, such as broadcast TV and video on demand to consumers. The service delivery network is dedicated for video traffic with a walled garden approach for delivering channels, content, and services over this infrastructure. The major IPTV service providers include AT&T, France-Telecom, and PCCW.

IPTV is traditionally delivered over the managed network into a subscriber's home. Increasingly, IPTV service providers are leveraging over the top (OTT) technologies as a means to keep their subscribers engaged and allowing content to be viewed online.

- **Over the top (OTT) video.** OTT, or streaming video, uses the Internet to deliver a wide variety of video content, usually offered on Web sites. Whereas IPTV commonly delivers studio grade video content such as sitcoms, original series and broadcast news, OTT includes user generated content (UGC), semi-professional, and samplings of studio content that is typically re-broadcast. Increasingly, live events and studio content is also being made available on the Web as a means to increase audience and brand advertising.

OTT content can be viewed on a number of devices, such as TV sets equipped with Internet access, or a hybrid set top box with Internet capability, personal computers, tablets, smart phones, netbooks, and similar devices.

## IPTV Delivery

### How is IPTV Delivered?

Delivery of video to the consumer has undergone rapid change in recent years and is guaranteed to continue to do so in the future. Cable TV networks deliver a large range of content, and the ability to provide user interactive features, including VoD.

Carrying the most promise for the future is delivery of video over multi-service IP networks. This is commonly referred to as IPTV. It is delivered as a Triple-Play service to consumers that includes high speed Internet (HSI) and Voice over IP (VoIP).

### Video over IP Information Flow

The major components and data flow in IPTV networks consists of media and control flowing between content servers and home networks.

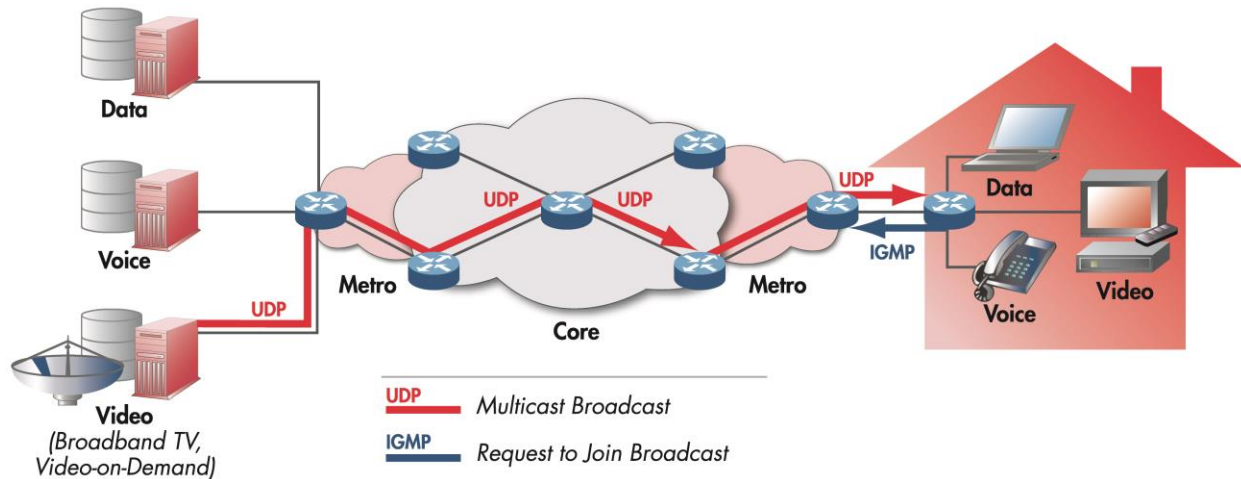
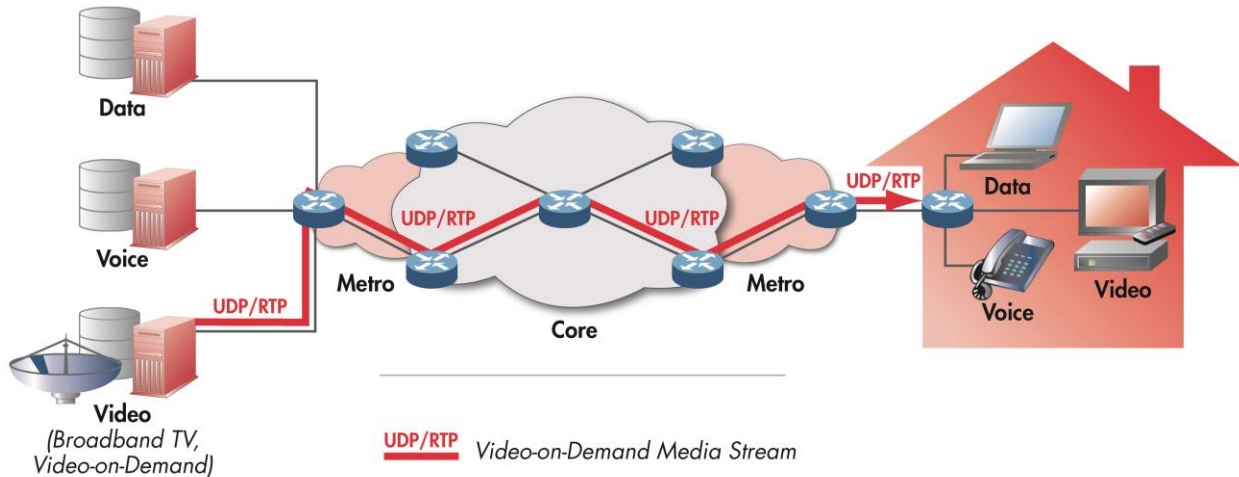


Figure 1. Broadcast TV Information Flow

The two types of video services delivered are linear broadcast and VoD. Both have dramatically different characteristics that affect the networks that handle them. Broadcasts are regularly scheduled programs sent to large numbers of subscribers. It is sent efficiently over multicast IP routes.

## Video over IP

VoD service delivery exhibits an entirely different behavior from linear broadcast service. Stored videos are sent to the subscriber on demand. Each subscriber receives their own video flow, which they can control with VCR-like controls.



**Figure 2. Video on Demand Information Flow**

The differences are responsible for the complexity of the delivery network. Broadcast TV over IP is primarily a one-way channel, using well understood multicast protocols. The home network is responsible for multicast messages and image display.

VoD adds another level of complexity. Requests for and control of video content are transmitted upstream from the subscriber to the service provider using the Real-Time Streaming Protocol (RTSP). Video content is returned to the subscriber through the Real-time Transport Protocol (RTP).

## **IPTV Delivery Challenges**

Until differentiating services are developed for IP-based video-voice-data networks, IPTV services will continue to be compared with traditional TV, cable, and satellite service. As such, the IP delivery network must remain transparent to customers. Customers expect video quality and service availability to be on par or better to make the switch.

With multiple choices available to consumers, there is little tolerance for poor quality and operational problems. A poorly engineered network can lead to substantial customer churn.

To successfully deploy IPTV, the following end-user requirements must be addressed:

- Video quality: Subscribers' perception of quality must be the same or better than other alternatives
- Minimal channel change delay: Because instant response is expected
- Assured service delivery and availability for an always-on service.

## **IPTV Testing Requirements**

Service providers must systematically test and verify network devices in each of the video transport architectures, including video content servers, core and edge routers, access devices, and customer premises equipment. Such testing provides an understanding of individual device performance and may determine how much impact each has on the overall system.

System-level tests that incorporate more than one demarcation point in the transport architecture are required. In this way, a clear understanding of how well the individual systems play with each other is determined.

Finally, the network must be tested end-to-end. Most standard routing and forwarding performance tests should be performed, looking at packet loss or latency under different load conditions.

The modern IPTV delivery system is a complex beast. The sub-systems that we will look at are as follows:

- Super Video Head-End
- Video Transport Network
- Access/Broadband Network
- Infrastructure Components

## Ixia Test Solutions

Ixia's IxLoad is a highly scalable, integrated test solution that assesses the performance of triple-play networks and devices. IxLoad emulates IPTV and triple-play subscribers and associated protocols to ensure subscriber Quality of Experience.

IxLoad supports the full range of protocols used in IPTV and triple-play testing:

- Video: MPEG, IGMP, RTSP
- Voice: SIP, MGCP, RTP
- Data: HTTP, FTP, SMTP, DNS, DHCP, and others

IxLoad has a number of powerful features designed specifically for IPTV testing:

### Video Subscriber Emulation

- IGMPv1, v2, and v3 with source specific joins
- Multicast channel list, channel selection profile, and viewing behavior
- Video on Demand with RTSP (RTP/UDP, UDP)
- Support for RTCP RR, Redirection, and Proxy
- VoD commands like PLAY, SEEK, PAUSE, FF, RW
- Access protocols, such as DHCP, PPP/L2TP, QnQ, and VLAN tags

### Video Server Emulation

- High performance streaming with multicast and VoD at the same time
- Support for MPEG2, H.264, AVC in SPTS and MPTS
- Support for synthetic and real video files
- Stream CBR and VBR content

### Video Quality Measurement and Analysis

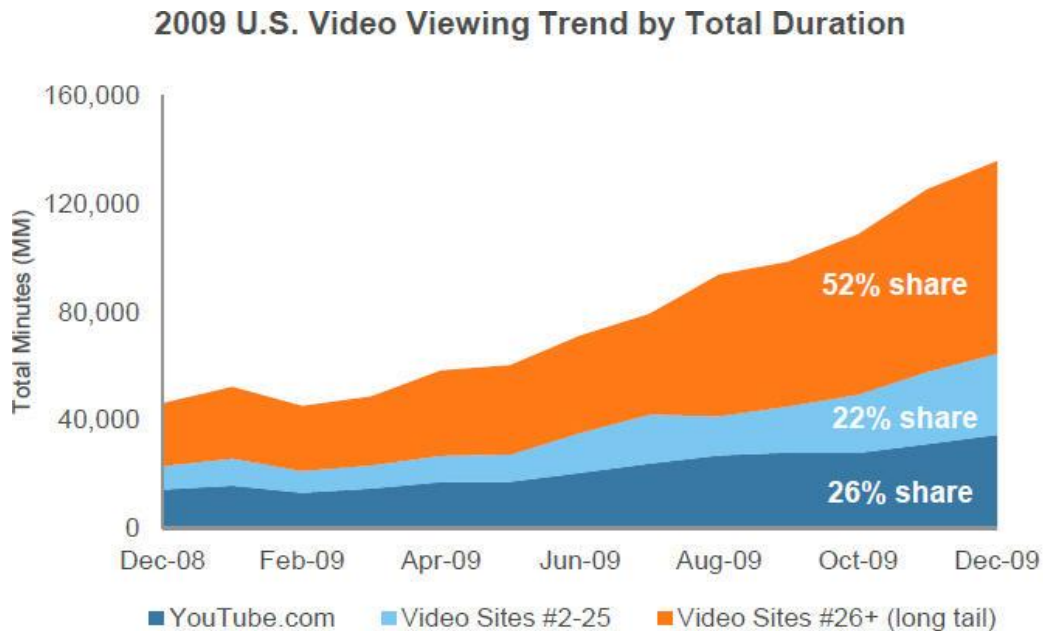
- Extensive statistics for loss, jitter, IAT, PPDV
- Real-time MDI analysis
- True perceptual video quality analysis (MOS)
- MPEG2 video capture and playback
- Full packet capture and analysis

## Over the Top Video Delivery

### Overview

OTT video, or streaming media, is an evolving set of technologies that deliver multimedia content over the Internet and private networks. A number of online media platforms are dedicated to streaming media delivery, including YouTube, Brightcove, Vimeo, Metacafe, BBC, and Hulu. Streaming video delivery is growing dramatically: according to the comScore Video Metrix<sup>1</sup>, Americans viewed a significantly higher number of videos in 2009 than in 2008 (up 19 percent) because of both increased content consumption and the growing number of video ads delivered. In January 2010, more than 170 million viewers watched videos online. The average online viewer consumed 187 videos in December 2009, up 95 percent over the previous year, and the average video duration grew from 3.2 to 4.1 minutes. Hulu, for example, in that same month delivered more than 1 billion streams for a total of 97 million hours.

According to comScore, the character of video viewing is changing as well, with more people watching longer content.



Source: comScore Video Metrix (U.S.)

Figure 3. Changing Video Usage

There is a growing effort by broadcasters to make regular TV content available online. The BBC has developed the BBC iPlayer™ and the bbc.co.uk Web site to support replication of most BBC broadcast material. The service has been outstandingly successful: 79.3 million requests were serviced in October 2009.<sup>2</sup> Most recently, NBC coverage of the 2010 Winter Olympics included live and recently recorded content, complete with commercials.

<sup>1</sup>The comScore 2009 U.S. Digital Year in Review.

<sup>2</sup>BBC iPlayer online monthly press pack. October 2009.

Whenever there is the possibility of a large or dynamic viewer audience, a reliable distribution content delivery network (CDN) is required. CDNs once only used to replicate Web site content around the world. Now, they have expanded dramatically to handle streaming media. Research and markets estimated the value of CDN services for 2008 at USD1.25 billion, up 32 percent from 2007. Top CDNs include Akamai, Mirror Image Internet, Limelight Networks, CDNetworks, and Level 3. Streaming media services must deal with content collected from disparate sources and distributed to a growing number of devices. A generic aggregation/distribution network is shown in the following figure.

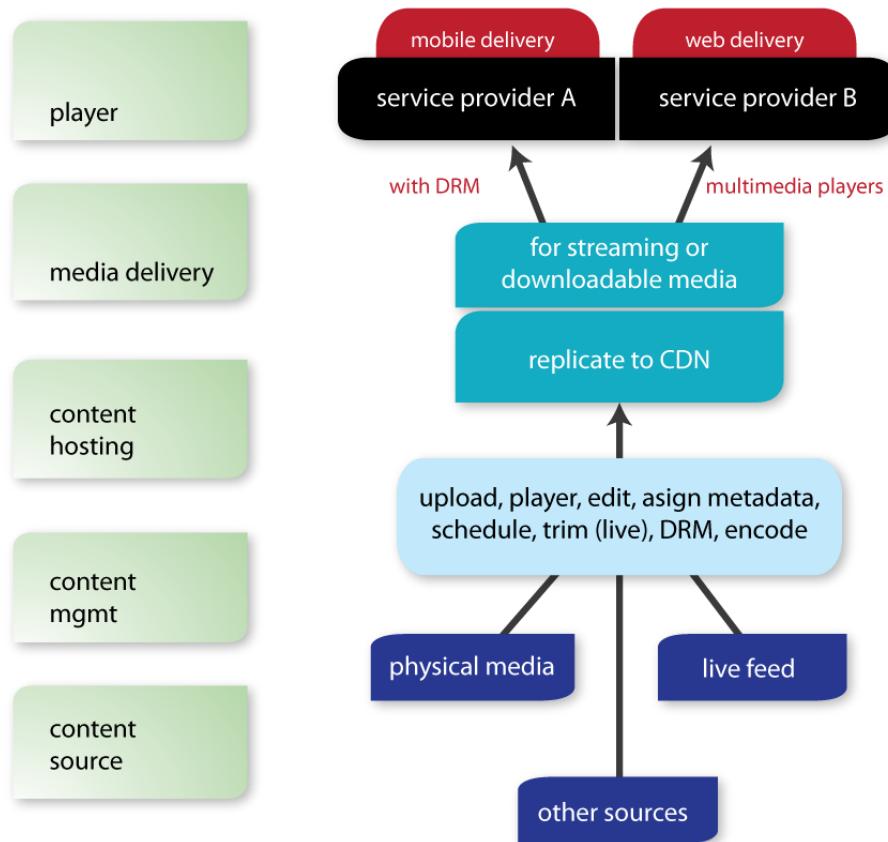


Figure 4. Generic Streaming Media Service

The five stages shown of an aggregation/distribution network are as follows:

- **Content sourcing:** Aggregation of content from physical media, live feeds, and other sources.
- **Content management:** Editing and management of the content, including uploading with digital rights management, with content encoded for multiple forms of delivery.



- **Content hosting:** Centralized library of video content, plus replication to other levels of a content delivery network (CDN).
- **Media delivery:** The process of providing content in response to user requests, either through bulk download or through streaming, with optional digital rights management (DRM).
- **Player:** The software on the end-user device used to view and interact with content.

The number and types of destination devices and platforms are continually growing, and include hybrid set-top boxes (STBs), personal computers, Apple iPhone, and various other Internet connected smart phones, tablets and Netbooks.

### Business Trends

But who is making money here? Media server hardware and software vendors and CDNs can charge for their premium services. Content owners, however, are still trying to find a business model that will allow them to get a return on their investment in their content.

There are three emerging business models being pursued to generate revenue:

- **Based purely on advertisements.** An online property is used as an aggregation site for a variety of content, with advertisements as the primary revenue source. Hulu's effort in particular is on a very large scale, and depends solely on advertising income at the moment. As a venture of NBC Universal, News Corp, the Walt Disney Company, and Providence Equity Partners, Hulu offers more than 1,700 current primetime TV hits using Adobe Flash™ technology as its delivery mechanism. Large advertisement networks from Tremor Media and Brightroll provide intelligent advertising platforms.
- **Based on consumer subscription.** Customers pay for access to premium and syndicated content. Networks such as the XBOX Marketplace provide consumers with access not only to gaming, but also to significant content that can be purchased. Netflix was the first to offer a subscription video rental service, and it is highly regarded by its customer base for quality and content.
- **Based on enterprise subscription, pay per use.** Enterprise customers leverage expert platforms to create a significant online brand presence. Brightcove is a leading platform used by enterprises with a complete video management platform that performs functions from uploading videos, to syndication, to seamless multiplatform distribution.

Operators in three categories are also searching for income-generating business models:

- **Traditional MSOs and service providers.** Carriers in particular are promoting 'TV everywhere,' a feature that makes original broadcast TV programming available as 'catch up TV' online for its subscriber base, free of charge. An ad supported model is frequently used to recoup the cost of this distribution. Service providers are also working towards a harmonious delivery of Internet content for in-home and on-TV entertainment as a way to create differentiation and embrace the new medium. Others are looking to increase their average revenue per user (ARPU) by offering alternate means of viewing

## Video over IP

broadcast and other content by making it available for purchase using video-on-demand services.

- **Stand-alone players and independents.** These are the new players in the video delivery market, and include online platforms or connected devices placed in homes that provide consumers access to movies, TV shows, and Web content for a nominal, recurring fee. Independent operators often do not need to share revenue with service providers that offer Internet access, but instead compete with TV operators in some cases. Companies such as Apple TV, Netflix, Boxee, Hulu, XBOX360 Marketplace, and TiVO are good examples. Walmart recently announced their intention to buy Vudu to distribute films and movies over the Internet in 2010.
- **Content owners.** National television channels with years of valuable content and highly acclaimed new content are also generating revenue through delivery to online and mobile audiences. Content owners often leverage global CDNs to syndicate and distribute their content.

### Technology Trends

The most common network protocol used to transport video over IP networks is real-time streaming protocol (RTSP). RTSP is a stateful protocol used to establish and control media sessions between a media server and client viewer. RTSP clients issue VCR-like commands to control media playback. The transmission of the audio/video stream itself is most often handled by the real-time transport protocol (RTP), although some vendors have implemented their own transport protocol. RTSP and RTP are almost universally used to implement video on demand (VoD) features.

Most video players, such as the Adobe Flash Player, use proprietary protocols that provide additional functionality and flexibility. Flash Player has an almost total presence on PCs and MACs, and is used to deliver over 80 percent of online videos. The Adobe Flash Player is a lightweight client embedded in Web browsers. Adobe uses the real-time messaging protocol (RTMP) to deliver streaming content, providing multiple independent channels, which are used to control and deliver content. RTMPT is an RTMP variant that encapsulates RTMP packets in HTTP. Adobe offers a more modern playback technology called HTTP Dynamic Streaming (HDS) which uses HTTP.

First released in 2007, Microsoft's Silverlight™ player is growing in popularity within the player market. The Silverlight player uses HTTP as its top-level transport mechanism and for media streaming. Using HTTP as a single transport mechanism can result in significant Internal cost reduction for end-to-end delivery. Silverlight includes digital rights management (DRM) features similar to those available in Adobe Flash.

HTTP Live Streaming (HLS) is a media streaming specification that is developed by Apple® Inc that uses HTTP as the transport. Devices such as iPhone, iPad, and Apple compatible platforms support this streaming technology. The 'Live' is misleading in the name, as this technology works for on-demand and live streaming. HLS supports streaming media that is segmented into smaller chunks of data, to improve delivery and user experience. An Extended M3U Playlist format file is used that contains the media segments to download. The HLS specification is documented in the following draft RFC: (draft-pantos-http-live-streaming-04, <http://tools.ietf.org/html/draft-pantos-http-live-streaming-04>).

Modern streaming media technologies adapt to changing network conditions, especially those related to mobile devices. As conditions degrade or improve, the player requests an alternate lower or higher bitrate media stream. Multiple flows are pre-built or constructed at multiple bit rates and divided into chunks so that a player can seamlessly switch different flows. The ability for a video player to adapt to varying network conditions is termed differently across players: In the Silverlight player, it is called smooth streaming; Adobe Flash 10.1 terms it dynamic streaming, and Apple iPhone's HLS refers to it as adaptive streaming.

### Building in Reliability

The ultimate measure of reliable delivery is user satisfaction, often referred to as their quality of experience (QoE). For streaming media delivery, the key QoE factors are as follows:

## Video over IP

- Continuous play: Without start/stop pauses
- Absence of video or audio skips
- Quick response to user actions that select and start the video, as well as for pause, rewind, and fast-forward operations
- Availability of low and high resolution versions

Jitter, loss, and latency are inherent in every IP network. These factors are most often compensated for by buffering at multiple network levels. Players commonly buffer data before beginning a presentation and read ahead to guarantee error-free delivery. CDN nodes provide buffering as well, although usually with larger blocks of data.

Rigorous testing of all streaming media delivery chain components is required to ensure user QoE. Components and networks must be tested under load to determine their limits. Streaming media audiences can be extremely dynamic, responding to special events or viral popularity. It is especially important to test the devices that perform special handling on media flows:

- **Media servers:** Establish client connections, and convert and deliver content.
- **Content delivery networks:** With sophisticated, multi-level architectures that distribute content from a central site to caching nodes, and then finally to streaming servers located regionally and globally. Each level, and combination of levels, must be tested, especially for delay. For example, the first viewer who requests a video that is only present at the central library site must not experience undue delay as the content is distributed to caching and streaming servers.

## Ixia's Test Solutions

IxLoad provides large-scale emulation of streaming media clients to load test media servers, and cache and content delivery networks.

IxLoad has a number of powerful features designed specifically for Over the Top video testing:

### **HTTP Progressive Download client and server**

- HTTP Client to request variety of media content
- Measure time to connect to server and time to download content

### **RTSP Streaming client and server**

- Interact with media streamers and content systems
- Support for WMV, Quicktime, Real media streams
- RTSP with RTP/UDP or RTP/TCP media streams
- Support for redirect and proxy

### **Flash™ Player Emulation**

- Interact with Flash Media Server (FMS) and compatible media systems
- RTMP, RTMPT, RTMPE and RTMPTE protocol support
- On-demand and live stream playback
- Handshake, Netconnection reporting

### **HTTP Dynamic Streaming (HDS) Player Emulation**

- Interact with Adobe FMS and HDS streamers
- On-demand and live stream playback with HTTP proxy support
- Bit-rate shifting to emulate users changing to streams of varying quality

### **Silverlight® Player Emulation**

- Interact with Silverlight compatible media systems
- On-demand and live stream playback with HTTP proxy support
- Bit-rate shifting to emulate users changing to streams of varying quality

### **HTTP Live Streaming (HLS) Player Emulation**

- Interact with Apple® HLS compatible media systems
- On-demand and live stream playback with HTTP proxy support
- Bit-rate shifting to emulate users changing to streams of varying quality

## Getting Started Guide

For application performance tests, Ixia's Layer 7 test solution, IxLoad™, is used in this methodology. The following section is designed for new users who wish to become familiar with the user interface and workflow to create and run tests in IxLoad.

The user interface is presented in Figure 5. The Test Configuration pane is used to navigate the various configuration windows used to create the test elements, including network, traffic, user behavior, and other advanced capabilities.

The Statistics pane is a real-time statistics viewer during an active test. All of the key performance indicators are present in this view.

The Analyzer pane is a real-time packet capture tool and a decode viewer. It offers a simple, powerful way to capture traffic on the test ports for debugging purposes. Packet analysis is done to show conversations between peers and client/server flows.

### User Interface Settings (IxLoad 5.40 shown here)

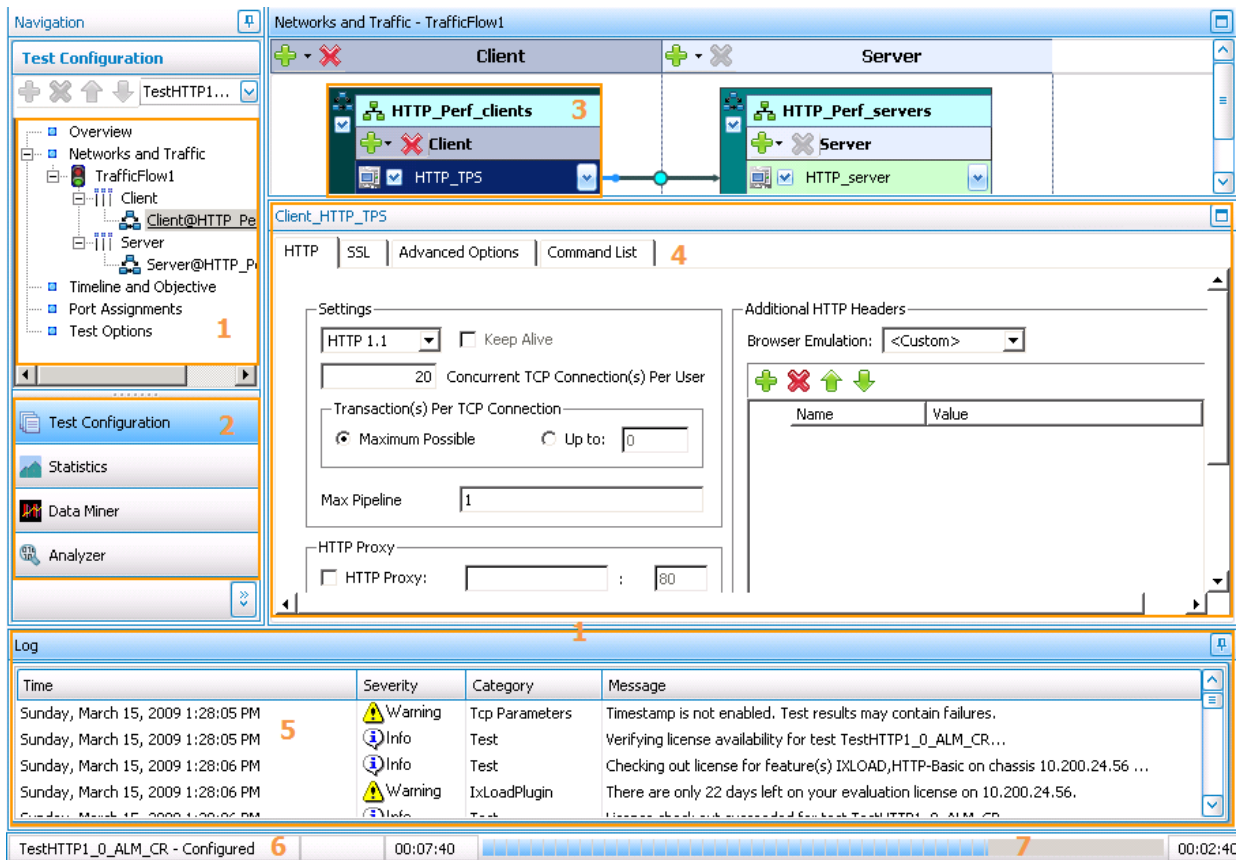


Figure 5. Main IxLoad GUI

The numbers in the following discussion correspond to the numbers in Figure 5.

## Getting Started Guide

1. The left navigation panel is used to select all configuration windows. It contains the network and traffic configurations, setting test duration and objective, and assigning test ports.
2. This panel switches views between test configuration, looking at real-time statistics, or accessing the Analyzer view for analyzing captured packets.
3. The network and protocol configuration object is called a NetTraffic. The network IP addresses, protocol configuration, and request pages are configured by selecting the network or the activity object and configuring the details in the bottom window.
4. Detailed configuration for network and protocol configuration is done here. Network settings, protocol configuration, page sizes, and user behavior (that is, what pages to request) are configured here.
5. The log window provides real-time test progress and indicates warnings and test errors. Keep this window active to become familiar with IxLoad's workflow and test progress.
6. The test status is indicated here, such as Unconfigured or Configured. Configured refers to an active test configuration present on the test ports.
7. Test progress is indicated here for a running test, with total test time and remaining duration. The test objective and duration is configurable from the Timeline and Objective view that is accessed from the tree panel from (1).

Before getting started, refer to the following figure to understand IxLoad's workflow.

**User Workflow for Configuring and Running a Series of Tests**

These are the steps to create and run a series of tests in IxLoad:

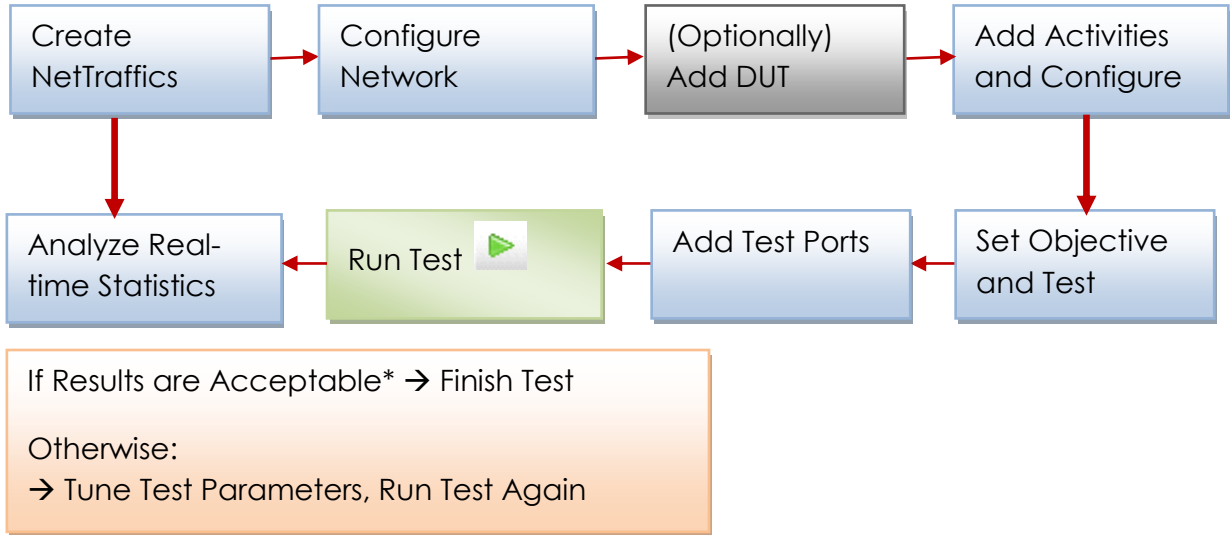


Figure 7. Workflow for Configuring and Running a Series of Tests in IxLoad

\* Acceptable test results are based on the target desired versus what was actually attained. Additionally, for each type of test, key performance metrics should be examined to determine if the results obtained can be considered acceptable.

It is important to establish a baseline performance. A baseline test is one that only uses the test ports; the test profile is configured to be very similar to the actual desired profile to determine the test tool limit. The baseline performance can be used to scale up and build the test profile to appropriately measure the DUT performance.

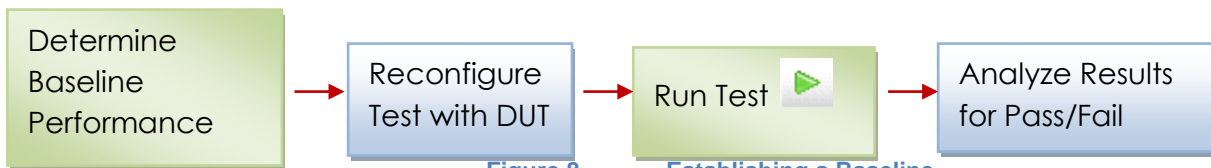


Figure 8. Establishing a Baseline





## Test Case: RTSP Media Server Testing

### Overview

Real-Time Streaming Protocol is used by a number of streaming platforms to deliver instant, on-demand video, and audio to Web browsers on PC, MAC, and mobile handsets. RTSP offers a rich set of capabilities that allow client and server to negotiate capabilities, manage streams in real-time, and give user instant control of playback such as pause, seek, fast forward, and rewind functionalities.

RTSP is used by a leading number of video server vendors and open source platforms, including Real Server, Darwin, and Windows Media Server. Some of the video content that is delivered over RTP includes WMV, ASF, MOV, MPG, MP3, RT, and WAV.

### Objective

Set up a test profile in IxLoad to act as a Windows Media Player to stream content from a Windows Media Services running on the Windows 2003 or 2008 server.

The control plane transport is RTSP and media is delivered over RTP/UDP, using Microsoft compliant RTP stream multiplexing.

### Setup

The following scenario is used: Windows 2003 Server running Windows Media Services (WMS) to host a variety of content, IxLoad 5.10 RTSP Activity with RTP/UDP transport.

One Ixia test port is connected directly to the WMS. The proxy is optional and not used in this test case. RTSP proxy is supported in IxLoad.

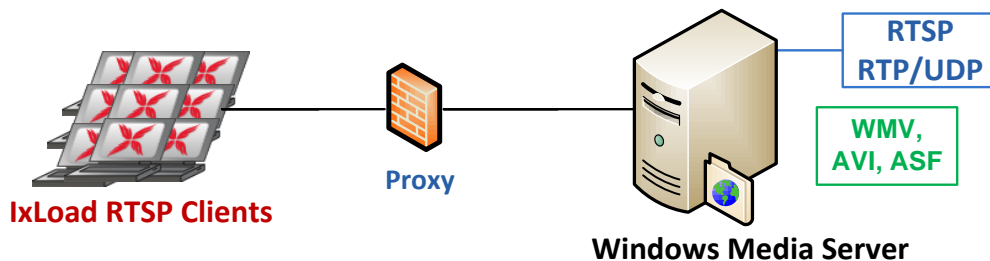


Figure 8. Typical setup with RTSP emulation connecting to Windows Media Server

### Test Variables

#### Test Tool Variables

The RTSP activity supports the following capabilities:

Test Case: RTSP Media Server Testing

Parameters	Description
RTSP Clients	100 IP addresses or more, use sequential or 'use all' IP addresses.
RTSP Client parameters	For Windows Media Server, select RTP/UDP and the <b>Setup all streams on same RTP port</b> option in the <b>Advanced</b> tab.  RTP/TCP is also supported. The <b>Setup all streams on same RTP port</b> option is not required.
TCP parameters	TCP RX and TX buffer at 32768 bytes.
RTSP client command list	<b>{PlayMedia}</b> command Destination: Use server IP ( <b>192.168.1.100</b> ) or Destination: Use FQDN and enable DNS in Network settings Media: Use content name ( <b>/videoasf/video[1-5].asf</b> ).
RTSP Proxy support	(Optional) If clients connect through a content switch or proxy, IxLoad can send all RTSP packets to this configured <b>IP:port</b> .
RTSP Redirect	(Optional) If clients must follow redirects to reach video server, select the <b>Follow RTSP Directs</b> option. The client will follow the URI in the Location header in the 3XX response.
Use of sequence generator	(Optional) Sequence generator is used above to expand <b>video[1-5].asf</b> to video1.asf through video5.asf. Each user in the test will start with video1.asf and cycle through the sequence.

Figure 9. Test Variables

## DUT Test Variables

Device(s)	Variation	Description
Windows Media Server	RTP Transport setting	Enable RTP/UDP for media transport.

Table 1. – DUT Variables

## Step-by-Step Instructions

### Scenario to Emulate

START

PLAYMEDIA (TILL END)

1. Start IxLoad. In the main window, the **Scenario Editor** window appears. All test configurations will be done here.
2. To get familiar with the IxLoad GUI, see the Getting Started Guide section.
3. Add the client NetTraffic object. Configure the Client network with total IP count, gateway, and VLAN, if used. Use a count of 100 to start.
4. For a step-by-step workflow, see Annex A.
5. Configure the RTSP client. Add the RTSP client activity to the client NetTraffic. Refer to the Test Variables section to configure the RTSP parameters. Take note of optional configuration parameters.
6. Having set up the client networks and the traffic profile, the test objective can now be configured. Configure the simulated user as the test objective and set objective value to the desired number of users.
7. To ensure that no interoperability or configuration issues exist (such as wrong content name), you can run a one simulated user test and enable capture on the port to see the packet exchange for possible issues. Use filter 'Protocol Filter' to only capture control packets.
8. Add test port resources and run the test. Refer to the Results Analysis and the Troubleshooting and Diagnostics section for further information.

## Results Analysis

When testing interoperability and performance of a real server, we recommend you to have access to the server administration console and log data to spot performance or configuration or interoperability issues.

Metric	Key Performance Indicators	Statistics View
Performance Metrics	Total Connections, Number of Simulated Users, Throughput	RTSP Client – Objectives RTSP Client – Data Rates
Application Level Transactions  Application Level Failure Monitoring	Presentations Active, Presentations Playing, Paused, Requested, Successful, Failed, and Playback Successful	RTSP Client – Presentations RTSP Client – Latency Distribution RTSP Client - RTP Jitter RTSP Client - Packet Loss
TCP Connection Information  TCP Failure Monitoring	SYNs Sent, SYN/SYN-ACKs Received  RESET Sent, RESET Received, Retries, Timeouts	RTSP Client – TCP Connections RTSP Client – TCP Failures

**Table 2. Key Performance Indicators that Require Monitoring**

## Real-time Statistics

The following graph provides a view of the real-time statistics for the test. Real-time statistics provide instant access to key statistics that should be examined for failures at the TCP and RTSP protocol level.

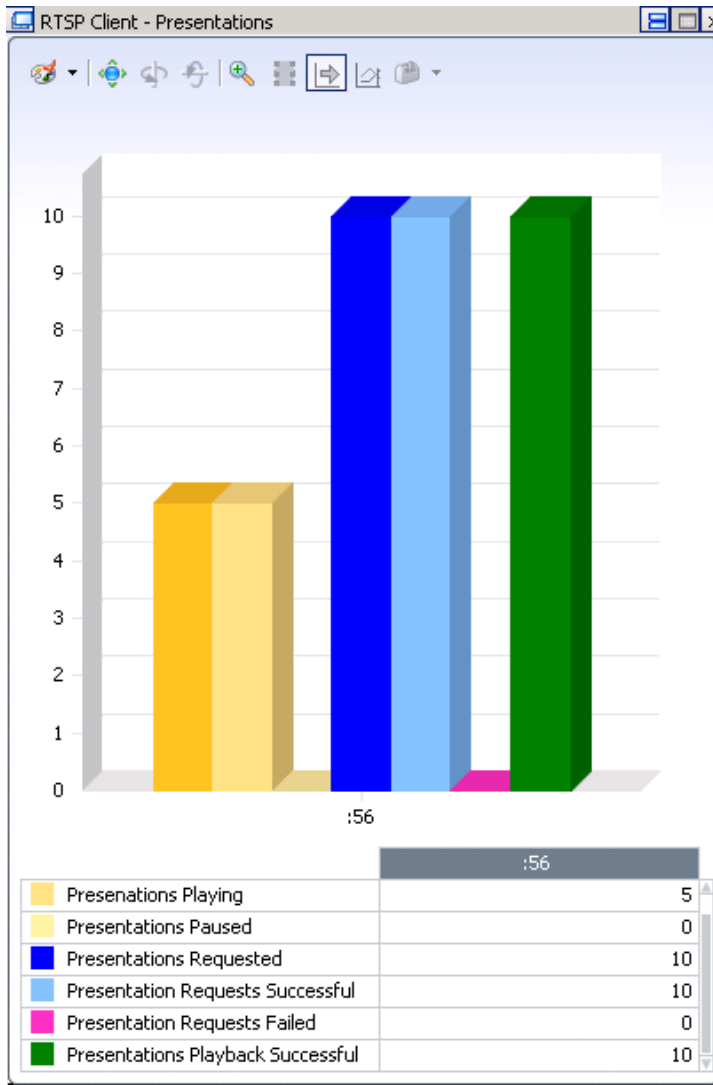


Figure 10. RTSP Client – Presentations view

The **RTSP Client - Presentations** view shows critical statistics for RTSP control plane. The 'Presentations Playback Successful' is an important statistic, which indicates that for each media that was setup, at least one packet was received on the expected RTP port. The 'Presentation Requests Successful' statistic indicates that a 200 OK was received for a PLAY command. For example, if a firewall blocks RTP/UDP packets, the 'Presentations Playback Successful' counter will not match 'Presentation Requests Successful,' identifying that not all streams were actually received, either as a result of firewall blocking RTP/UDP traffic or the server not actually sending the media stream on the expected UDP port.

# Test Case: RTSP Media Server Testing

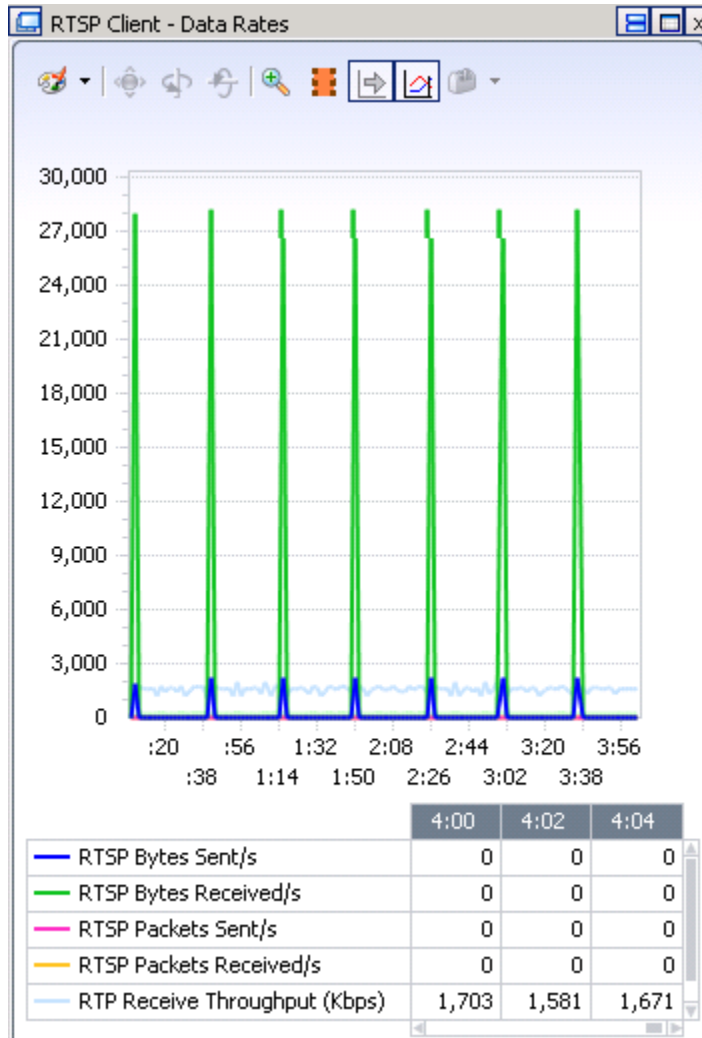


Figure 11. RTSP Client – Data Rates View

## Test Case: RTSP Media Server Testing

The **RTSP Client - Data Rates** view provides control plane and media throughput statistics.

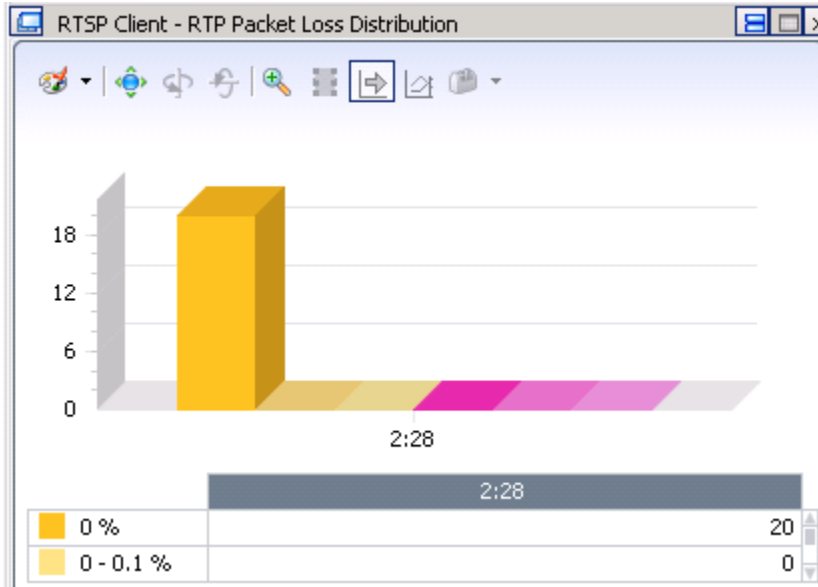


Figure 12. – RTSP Client – RTP Packet Loss Distribution view

The **RTSP Client – RTP Packet Loss Distribution** view considers only RTP media packets. This view provides a test level view on any flows that are being impacted by perturbed network or loaded server conditions causing packet loss.



Figure 13. RTSP Client – Play Latency Distribution view

The **RTSP Client - Play Latency Distribution** view indicates the time it took for the server to respond to the PLAY request. As the server gets loaded with many active sessions, it is possible (and common) to see the PLAY latency increase.



## Test Case: RTSP Media Server Testing

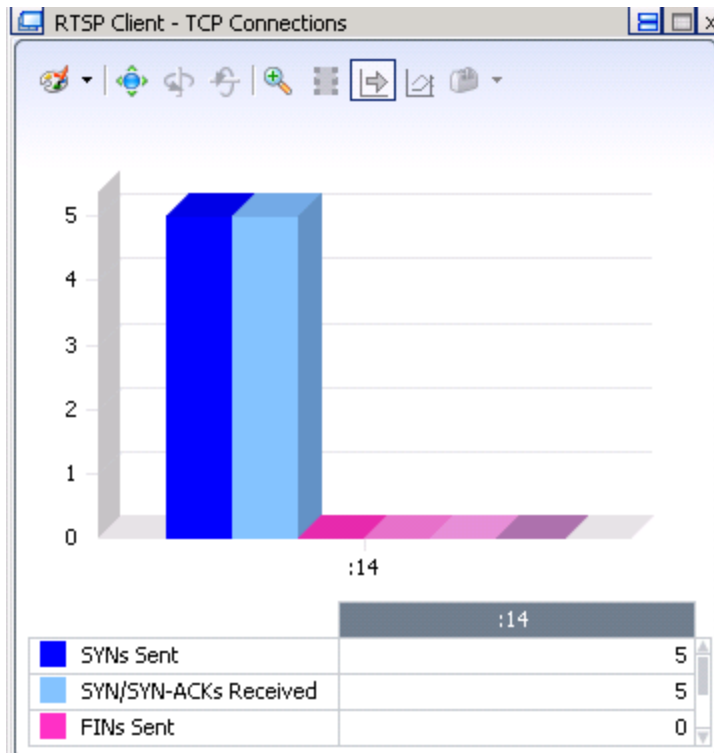


Figure 14. RTSP Client – TCP Connections view

The **RTSP Client - TCP Connections** view is important to identify reachability issues.

For example, if **SYNs Sent = 0**, it indicates that the client test port cannot resolve the gateway MAC or Server MAC; hence, it cannot send for a TCP connection to the unresolved host.

## Test Case: RTSP Media Server Testing

If SYNs Sent are seen, but no SYN-ACKs Received, it may indicate a server error or routing error to reach the server.

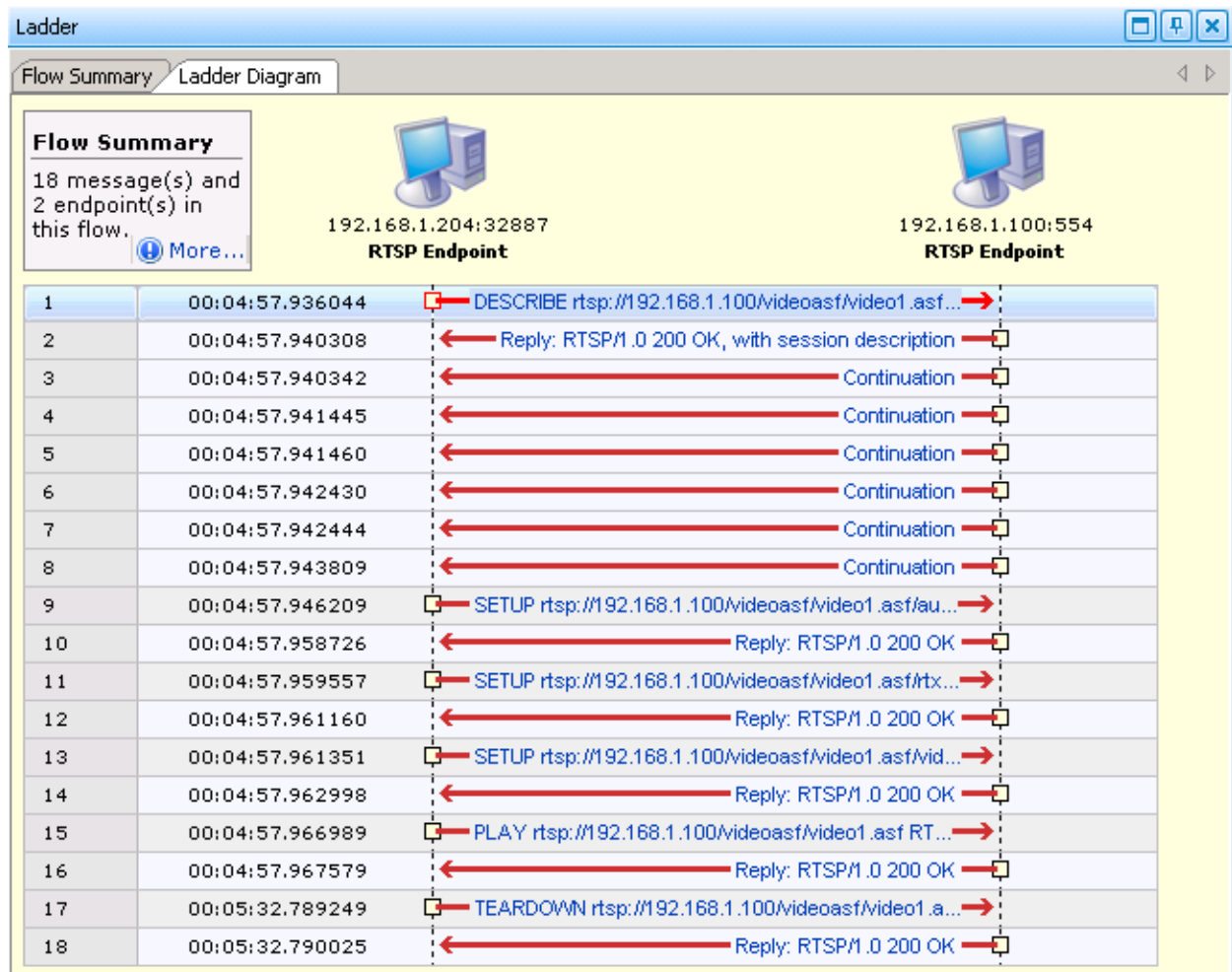


Figure 15. Analyzer view with automatic ladder diagram with RTSP protocol decode

## Test Case: RTSP Media Server Testing

Analyzer is an important debug tool to identify interoperability issues. In the sequence shown in the preceding figure, a complete DESCRIBE, SETUP (rtx), SETUP (audio), SETUP (video), PLAY, and TEARDOWN are present, indicating complete and successful transaction.



Figure 16. RTSP Describe Response with playback duration information

The packet snippet shown in the preceding figure is the Response from Server to the DESCRIBE request.

Note the 'npt=0.000-54.814,' which indicates the total length of the stream playback. By setting the 'Arguments' to 'PLAY\_TILL\_END' in the {PlayMedia} command, IxLoad automatically parses the SDP to learn the total stream length and sends TEARDOWN after this duration.

Command Properties for 'Play Media 1'	
Property Name	Property Value
Destination(IP or IP:Port)	192.168.1.100
Media	/videoasf/video1.asf
Arguments	PLAY_TILL_END

Figure 17. Play Media command property with "PLAY\_TILL\_END" used for playback duration

## Troubleshooting and Diagnostics

Issue	Diagnosis, Suggestions
No SYNs sent.	Test port cannot resolve gateway or Server IP to MAC. Resolve routing or switch issue. Are VLANs required?
TCP connections are succeeding and Presentation Requests Sent are also seen, but none are successful.	This behavior indicates an interoperability issue. Run a single user test and capture packets to see the issue. Is the media name correct? If not, 404 would be seen. Is the Transport correct? Is there any response to DESCRIBE?
When running a single user test to debug, DESCRIBE is sent, response is received, but no SETUP is sent.	This behavior indicates that the response contains unsupported response in the SDP that IxLoad RTSP does not understand.
RTP Received Throughput is 0, but 'Presentation Requests Successful' shows successful connections.	See 'Presentations Playback Successful.' If it is 0, either the server is not sending the actual media stream, or something in the path (firewall, proxy) is blocking it.

Table 3. – Troubleshooting checklist for RTSP Client Emulation



## Test Case: Basic VoD Server Interoperability

### Overview

Video on Demand (VoD) in pure IP-based IPTV deployment delivers high quality real-time experience for movies, events, and other interactive content on the big screen. VoD uses Real-Time Streaming Protocol (RTSP) as the control protocol that translates user's actions on a remote control into network protocol messages that allow instant control of playback, such as Pause, Seek anywhere, Fast Forward, or Rewind functionalities.

Leading VoD vendors such as Technicolor, Kasenna, Bitband, Motorola, Harmonic, Seachange, and Edgeware deliver content primarily over MPEG2 Transport streams, or native RTP transport. The most widely used video codec include MPEG2 and H.264/AVC for standard and high definition content.

### Objective

Set up a test profile in IxLoad to act as a generic VoD client to interact with a Darwin Media Server to stream standard definition content encoded in MPEG4, stored as MP4 container.

Interaction with a VoD server requires knowing the RTSP client specification that complies with the said VoD server. This test case is intended to highlight the steps required to learn the RTSP packet exchanges required, and how to set up an equivalent test profile in IxLoad.

Additionally, a number of advanced capabilities such as RTCP, Keep-alive, RTSP Redirects, and basic video quality analysis will be highlighted.

## Setup

The following scenario is used: Windows 2003 Server running Darwin Media Server hosting MP4 content, IxLoad 5.10 IPTV Client Activity with RTP/UDP transport.

One Ixia test port is connected directly to Darwin Media Server. The topology shown in the following image is representative of a more complex scenario with optional Proxy and Distribution/Origin servers.

- (1) All RTSP messages are sent to a configured proxy.
- (2) The proxy forwards the RTSP messages to the Hostname/IP that is destined in the RTSP messages, that is, the Distribution VoD server.
- (3) If and when content is not present at the Distribution VoD source, a RTSP redirect message is sent back to the client with a Location header that indicates a new URL; the VoD then closes previous TCP connection and initiates a new RTSP connection to the Origin VoD server to stream location through the Proxy.

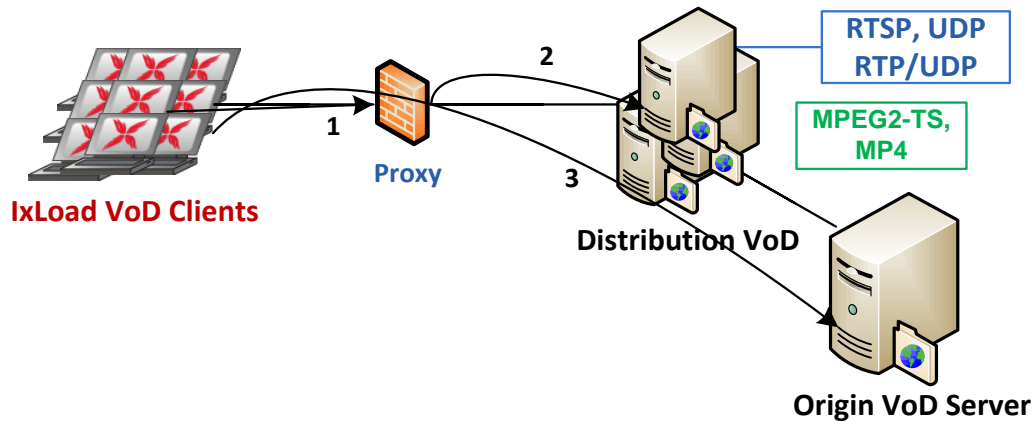


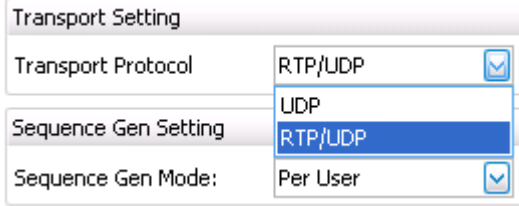
Figure 18. VoD Client Setup to connect to VoD infrastructure

IxLoad's VoD implementation is highly configurable on a per RTSP method basis.

It supports a number of major VoD vendor platforms by customizing the RTSP methods by replicating the details from examining VoD vendors' RTSP client/STB specification document. It is also possible to examine a 'real STB' network traffic exchange to understand and recreate a similar test profile in IxLoad.

## Test Variables

### Test Tool Variables

Parameters	Description
IPTV Clients	100 IP addresses or more, use sequential or 'use all' IP addresses.
IPTV Client parameters	<p>Select RTP/UDP as the 'Transport.' This is the most important setting that you must configure correctly.</p>  <p style="text-align: center;">Figure 19. VoD Transport selection</p>
TCP parameters	TCP RX and TX buffer at 32768 bytes.
IPTV client command list	<p><b>{PlayMedia}</b> command</p> <p>Destination: Use server IP (<b>192.168.1.100</b>) or                      Destination: Use FQDN and enable DNS in Network settings                      Media: Use content name (<b>sample[1-5].mp4</b>).</p>
Video Quality Analysis	(Optional) In the <b>Statistics Options</b> tab, select the <b>Quality Metrics</b> option. Leave the default settings for the buffer configuration.
RTSP Proxy support	(Optional) If clients connect through a content switch or proxy, IxLoad can send all RTSP packets to the proxy <b>IP:port</b> .
RTSP Redirect	(Optional) If clients must follow redirects to reach video server, select the <b>Follow RTSP Directs</b> option. The client will follow the URL in the Location header in the 3XX response.
Use of sequence generator	<p>Sequence generator is used above to expand <b>sample[1-5].mp4</b> to sample1.mp4 through sample5.mp4</p> <p>The way in which each user goes through the sequence is determined using the <b>Sequence Gen Mode</b> option.</p>
Sequence Gen Setting	<p>If <b>Per-User</b> is selected, all users start from the same index (for example, sample1.mp4 used by all users at start) and users move through the index at the end of each stream duration.</p> <p>If <b>Per-Command</b> is selected, all users stagger to start off uniquely (for example, user1 picks sample1.mp4, user2 picks sample2.mp4)</p>



Test Case: Basic VoD Server Interoperability

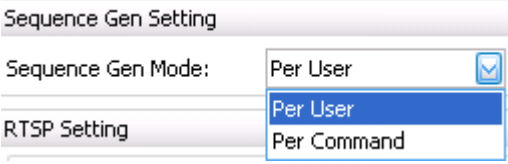
Parameters	Description
	<p>and each user moves through this staggered (unique) list based on stream length or configured time.</p>  <p style="text-align: center;"><b>Figure 20.</b> – Choosing a sequence generator mode of operation</p>
RTCP Receiver Reports (RR)	(Optional) IPTV client supports sending periodic RTCP RR to the server once every 1-3 seconds. The Receiver Reports include the current calculated <i>Inter-arrival Jitter</i> .
Keep-alive capability	(Optional) IPTV client supports using GET_PARAMETER using the {Keep-alive} command. This command will work if placed following the PLAY command. It will not work with other commands.

Table 4. – Test Tool Variables

DUT Test Variables

Device(s)	Variation	Description
Darwin Media Server	RTP Transport setting	Enable RTP/UDP without authentication.

Table 5. DUT Variables

## Step-by-Step Instructions

### Scenario to Emulate



1. Start IxLoad. In the main window, the **Scenario Editor** window appears. All test configurations will be done here.
2. To get familiar with the IxLoad GUI, see the Getting Started Guide section.
3. Add the client NetTraffic object. Configure the Client network with total IP count, gateway, and VLAN, if used.
4. For a step-by-step workflow, see Annex A.
5. Configure the IPTV client. Add the **IPTV Client** activity to the client NetTraffic. Refer to the Test Variables section to configure the IPTV client command list and parameters.
6. To interact with Darwin Media Server, only the {PlayMedia} command is required. The **{PlayMedia}** command is a composite command with DESCRIBE, SETUP, PLAY, and TEARDOWN commands. The response to the DESCRIBE contains all the information required for IxLoad IPTV client to parse and construct appropriate SETUP messages for video and audio streams.

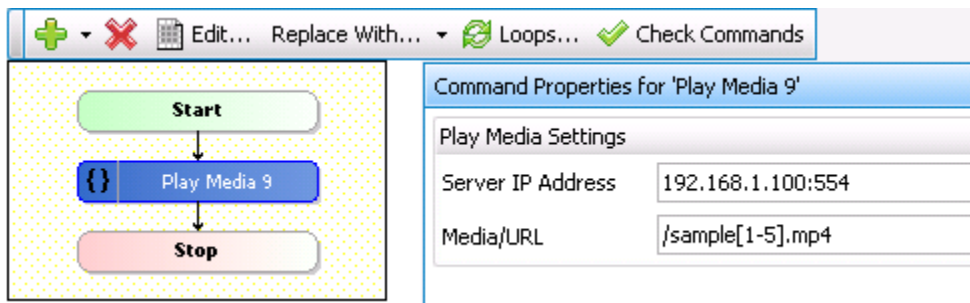


Figure 21. Play Media command setup for playback until end of stream

7. Having set up client networks and the traffic profile, the test objective can now be configured. Configure a simulated user as the test objective and set the objective value to the desired number of users.
8. To ensure that no interoperability or configuration issues exist (such as wrong content name), run a one simulated user test and enable capture on the port to see the packet exchange for possible issues. Use filter 'Protocol Filter' to only capture control packets.
9. Add test port resources and run the test. Refer to the Results Analysis and the Troubleshooting and Diagnostics section for further information.

## Results Analysis

When testing interoperability and performance of a real server, we recommend you to have access to the server administration console and log data to spot performance or configuration or interoperability issues.

Metric	Key Performance Indicators	Statistics View
Performance Metrics	Total Connections, Number of Simulated Users, Throughput	IPTV Client – Objectives IPTV Client – Data Rates IPTV Client - VoD Channel Requests
Application Level Transactions  Application Level Failure Monitoring	Presentations Active, Presentations Playing, Paused, Requested, Successful, Failed, and Playback Successful	IPTV Client – Presentations IPTV Client – Latency Distribution IPTV Client - RTP Jitter IPTV Client - RTP Loss
Video Quality Analysis  Network Metrics	Jitter, Loss, PPDV, IAT, MDI-DF, MDI-MLR, Out of Order, Duplicate	IPTV Client - Per Stream IPTV Client - Data Errors IPTV Client - Jitter IPTV Client - Pkt Latency
Video Quality Analysis  Perceptual Metrics	Avg Absolute MOSV, Ave Relative MOSV, Avg MOSA, Avg Interval Abs MOSV, Avg Interval Rel MOSV	IPTV Client - Per stream IPTV Client - MOS Scores

**Table 6. Results analysis for VoD testing**

## Real-time Statistics

The following graph provides a view of the real-time statistics for the test. Real-time statistics provide instant access to key statistics that should be examined for failures at the TCP and VOD (RTSP) protocol level.

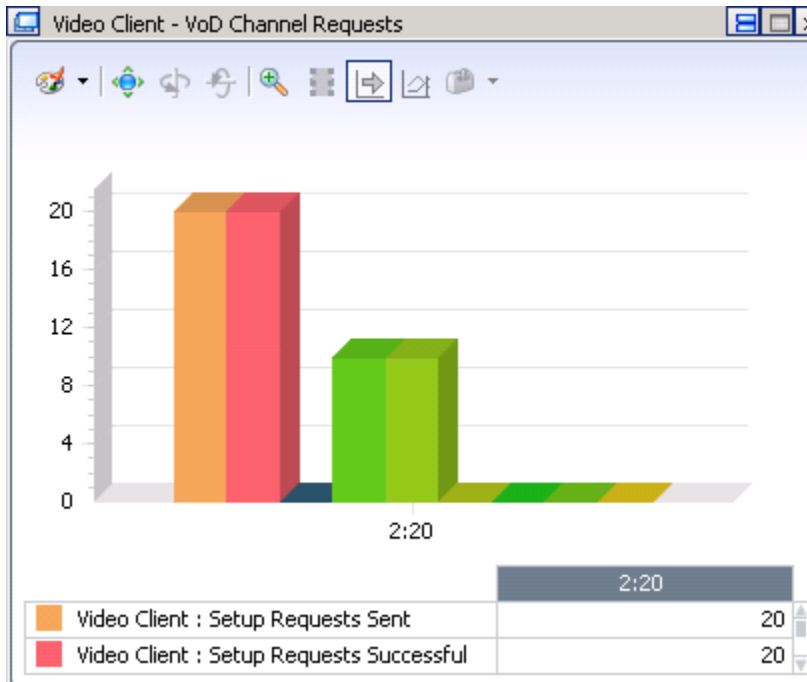


Figure 22. Video Client – VoD Channel Requests view

The **Video Client - VoD Channel Requests** view lists the RTSP control plane messages that are Sent/Successful/Failed. This view can be used to confirm that RTSP sessions are actively being established in real-time.

## Test Case: Basic VoD Server Interoperability

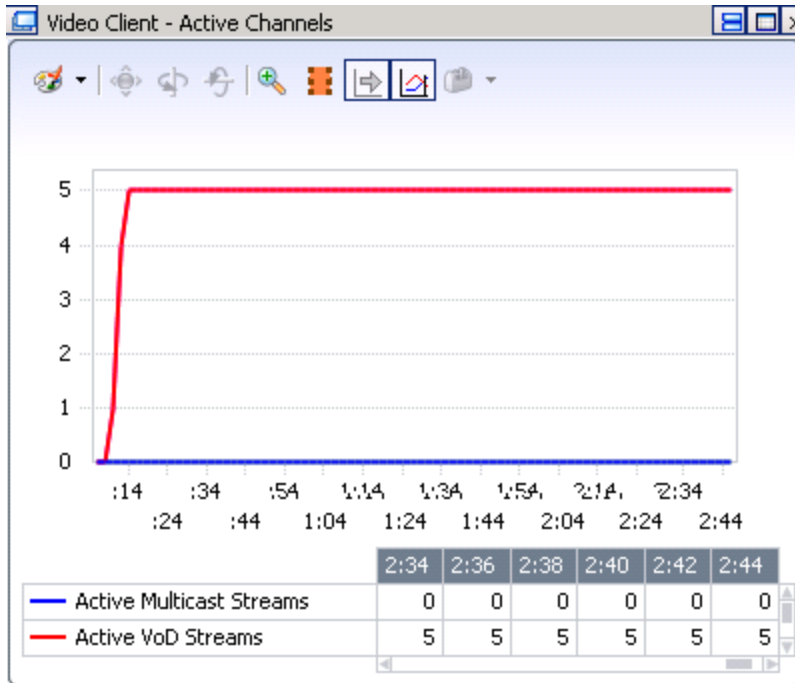


Figure 23. Video Client – Active Channels view

The **Video Client - Active Channels** view is useful to view how many channels are active or torn down in real-time.

## Test Case: Basic VoD Server Interoperability

IxLoad provides detailed per-stream information in the **Video Client - Per Stream** view. Information such as stream details (codec, transport, bitrate), network metrics (loss, jitter, PPDV, IAT, OOO, and duplicate packets) and perceptual video quality (MOSV, MOSA) information is available for 100s of streams.

Stat Name	Active	Stream Name	Flow ID	Transport	Video Codec	Stream Bit Rate (Kbps)
/10.200.134.132/Car...	YES	192.168.1.100-sample...	192.168.1...	RTP	SPTS/H264	8,313.207
/10.200.134.132/Car...	YES	192.168.1.100-sample...	192.168.1...	RTP	SPTS/H264	8,313.205
/10.200.134.132/Car...	YES	192.168.1.100-sample...	192.168.1...	RTP	SPTS/H264	8,313.208
/10.200.134.132/Car...	YES	192.168.1.100-sample...	192.168.1...	RTP	SPTS/H264	8,313.207
/10.200.134.132/Car...	YES	192.168.1.100-sample...	192.168.1...	RTP	SPTS/H264	8,313.206

Figure 24. Video Client – Per Stream view with Network and Stream information

Stat Name	Avg Absolute MOSV	Avg Relative MOSV	Avg MOSA	Avg MOSAV	Avg Interval Abs MOSV	Avg Interval Rel MOSV
/10.200.134.132/Car...	4.508	4.758	4.727	4.426	4.664	4.848
/10.200.134.132/Car...	4.508	4.758	4.727	4.426	4.664	4.848
/10.200.134.132/Car...	4.508	4.758	4.727	4.426	4.664	4.848
/10.200.134.132/Car...	4.508	4.758	4.727	4.426	4.645	4.848
/10.200.134.132/Car...	4.508	4.758	4.727	4.426	4.645	4.848

Figure 25. Video Client – Per Stream view with Perceptual Video Quality Analysis

Stat Name	Jitter (ns)	Inter Pkt Arrival Time (ns)	Min Inter Pkt Arrival Time (ns)	Max Inter Pkt Arrival Time (ns)
/10.200.134.132/Car...	937	1,263,253	1,226,320	1,299,240
/10.200.134.132/Car...	918	1,266,201	1,228,160	1,294,920
/10.200.134.132/Car...	1,133	1,266,121	1,232,960	1,292,340
/10.200.134.132/Car...	1,032	1,266,300	1,229,340	1,295,180
/10.200.134.132/Car...	699	1,265,719	1,232,980	1,293,720

Figure 26. Video Client – Per Stream view with Network level metrics

**Note:** Latency, jitter, and MDI are supported for CBR synthetic streams that IxLoad server sends. With real video streams, MOS video quality is supported for specific audio and video codec. You must select the **Quality Metrics** option for real-video streams to compute MOS. MOS is not supported for synthetic streams. MDI is supported on both synthetic and real-video streams. MDI is not calculated for variable bit-rate (VBR) streams. Global views provide insight into the health of a test. The statistics are cumulative averages across all active users and streams. Some key statistics are outlined in the following image:

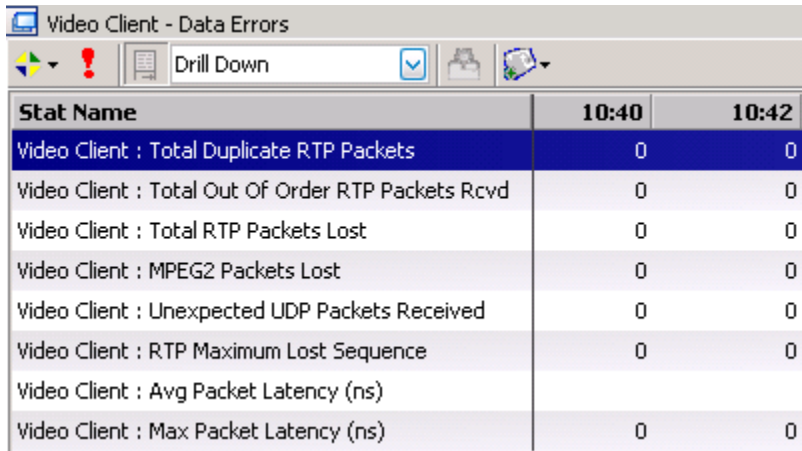
## Test Case: Basic VoD Server Interoperability

Stat Name	10:40 $\Delta$	10:42	10:44
Video Client : MOS AV (Current Streams)	4.426	4.426	0.027
Video Client : MOS AV (Completed Streams)	4.426	4.426	4.426
Video Client : Absolute MOS V (Current Streams)	4.508	4.508	0.027
Video Client : Absolute MOS V (Completed Streams)	4.508	4.508	4.508
Video Client : Interval Absolute MOS V (Current Streams)	4.652	4.648	0.027
Video Client : MOS A (Current Streams)	4.727	4.727	0.027
Video Client : MOS A (Completed Streams)	4.727	4.727	4.727
Video Client : Relative MOS V (Current Streams)	4.758	4.758	0.027
Video Client : Relative MOS V (Completed Streams)	4.758	4.758	4.758
Video Client : Interval Relative MOS V (Current Streams)	4.848	4.848	0.027

**Figure 27.** Video Client – MOS Scores view shows real-time and completed streams quality

## Test Case: Basic VoD Server Interoperability

The global **Video Client - MOS Scores** view provides complete video quality analysis for all streams active during a test. The 'current streams' relate to active streams not yet finished, and 'complete streams' are computed at the end of the test or end of the stream.



Stat Name	10:40	10:42
Video Client : Total Duplicate RTP Packets	0	0
Video Client : Total Out Of Order RTP Packets Rcvd	0	0
Video Client : Total RTP Packets Lost	0	0
Video Client : MPEG2 Packets Lost	0	0
Video Client : Unexpected UDP Packets Received	0	0
Video Client : RTP Maximum Lost Sequence	0	0
Video Client : Avg Packet Latency (ns)		
Video Client : Max Packet Latency (ns)	0	0

**Figure 28.** Video Client – Data Errors view shows overall performance of streams in the test

**Video Client - Data Errors** is a global view that indicates network health and monitors any specific issues such as RTP Packets Lost, Duplicates, and Out of Order. It also indicates 'RTP Maximum Lost Sequence,' which indicates the longest successive sequence of packets lost on any one stream.



## Troubleshooting and Diagnostics


Issue	Diagnosis, Suggestions
No SYNs sent.	Test port cannot resolve gateway or server IP to MAC. Resolve routing or switch issue. Are VLANs required?
TCP connections are succeeding and Presentation Requests Sent are also seen, but none are successful.	This behavior indicates an interoperability issue. Run a single user test and capture packets to see the issue. Is the media name correct? If not, 404 would be seen. Is the transport correct? Is there any response to DESCRIBE?
When running a single user test to debug, DESCRIBE is sent, response is received, but no SETUP is sent.	<p>This behavior indicates that the response contains unsupported response in the SDP that IxLoad RTSP does not understand.</p> <p>See the following example:</p>  <p>The image shows a network traffic capture with two packets. The first packet is a DESCRIBE request to rtsp://192.168.1.100/beach1.ts RTSP/1.0. The second packet is a reply from the server with status code 415 and the message 'Unsupported Media Type'.</p> <p><b>Figure 27 – Unsupported Media Error Decoding</b></p>
RTP Received Throughput is 0, but 'Presentation Requests Successful' shows successful connections.	Look at 'Presentations Playback Successful.' If it is 0, either the server is not sending the actual media stream, or something in the path (firewall, proxy) is blocking it.
There is no jitter or latency in the 'Per-Stream' view.	Delay and jitter is computed for synthetic payloads that are streamed by the Ixia IPTV server.
There are no MDI-DF values shown.	MDI-DF is calculated for constant bitrate streams only (synthetic and real video are supported).
All MOS related metrics show 0.	<p>MOS is computed for real files only, not synthetic.</p> <p>In addition, check that 'Video Quality' is enabled on the IPTV client 'Statistics' tab.</p>

Table 7. Troubleshooting checklist for VoD testing

## Test Case: Large Scale RTP Video - 600,000 Streams

### Overview

There is a rapid growth of multimedia capable smart phones and Internet enabled devices. RTSP with RTP/UDP is a very common way to deliver streaming media to millions of such devices.

Firewalls and edge routing devices that handle mobile IP networks must handle and maintain session state for millions of simultaneous RTP flows, at the rate of millions of small packets per second.

IxLoad's brilliant Acceleron-NP load module uses custom Network Processors (NP) to generate very large scale of RTP sessions. The NP cores can sustain 10 Gbps of line rate performance with small packets to create a realistic scenario with millions of packets per second to push firewalls, edge routers, and security platforms to their limits.

Test Case: Large Scale RTP Video – 600,000 Streams

The following table highlights the performance levels achieved by a pair of Acceleron-NP load modules.

Metric	Performance (Pair of Acceleron-NP)
CPS Without Media	24,000
CPS With Media	10,000
Concurrent sessions with 1 packet/sec	600,000
Concurrent sessions with larger packets/sec/stream	96,000 for 50 packets/sec per stream
Configurable Audio + Video RTP Stream per RTSP Session	Independent configuration for audio and video streams
128 kps, 1400 bytes/packet = 12 packets per second	75,000 streams 10 Gbps
128 kps, 600 bytes/packet = 27 packets per second	75,000 streams 10 Gbps

**Table 8. RTSP and Hardware RTP performance on Acceleron-NP pair**

## Objective

This test case highlights how to set up a test profile that uses a pair of Accelaron-NP to create 600,000 simultaneous RTSP and RTP/UDP media sessions.

In this scenario, the primary goal is to load the device's RTSP/RTP session table and processing multi-gigabit of large packets to maintain RTP sessions for a sustained duration.

## Setup

The following scenario uses two Accelaron-NP load modules running in 10G aggregated mode, connected directly to two ports on the firewall/RTP gateway.

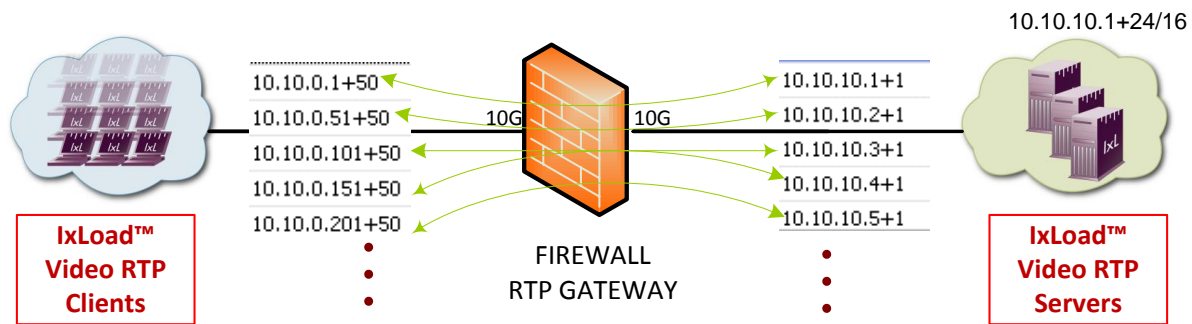
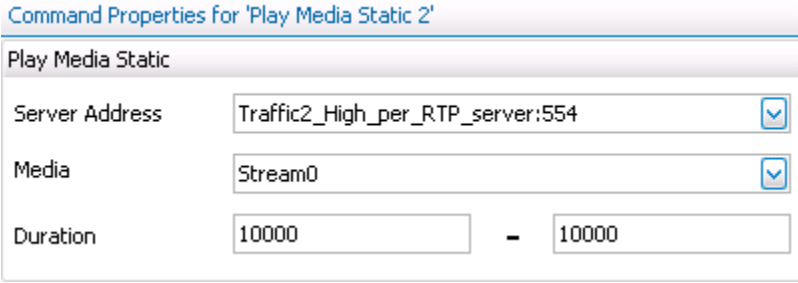
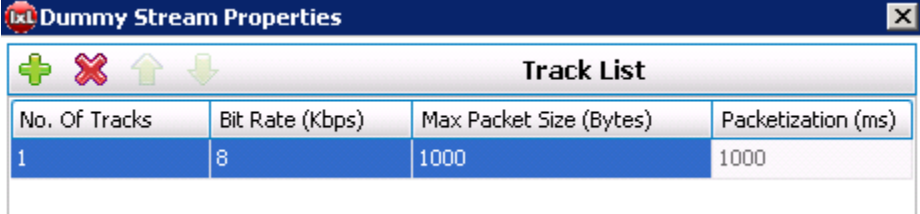


Figure 29. Setup scenario for testing an RTP Gateway

To achieve 600,000 concurrent RTP sessions, each RTP media session is configured at 1 packet per second. If the per-RTP session packet per second is increased, the total concurrent RTP stream count will reduce. This can be determined very accurately. See the **Results Analysis** section for examples.

## Test Variables

### Test Tool Variables

Parameters	Description
IPTV Client Network	12,000 IP addresses; 100 per port, 12 port test.
TCP parameters	TCP RX and TX buffer at 4096 bytes.
IPTV Client Activity parameters	<p>On the <b>Unicast Signaling</b> tab, choose Transport as RTP/UDP. Do not enable RTP HW Acceleration. Use {Play Media Static} command and configure as follows:</p>  <p style="text-align: center;"><b>Figure 30. Configuration for playback or Hardware RTP streams</b> Set Duration to 10000 or more seconds (to sustain RTP session).</p>
IPTV Server Network	24 IP addresses; 2 port test port (required).
IPTV Server Activity parameters	<p>Add VoD Channel Type. Choose 'Dummy Payload' and configure as follows:</p>  <p style="text-align: center;"><b>Figure 31. Configuration of tracks for hardware RTP streams</b></p> <p>Set Duration to 10000 or more. Set Stream Count to 50000. Set Transport to RTP/UDP. (1 RTP flow at 8 Kbps with 1000 byte packet equals 1 PPS).</p>
Use of Traffic Map	<p>A custom traffic map configuration is required. For example: 1200 Client IPs and 24 Server IPs Split 1200 IPs into 24 x 50 IPs for client.</p>

## Test Case: Large Scale RTP Video – 600,000 Streams

Parameters	Description
	Split 24 IPs into 24 x 1 IPs for server. Use 'IP Range Pairs' and apply this configuration. See the Step-by-Step Instructions section for details.

Table 9. Test Variables

### DUT Test Variables

Device(s)	Variation	Description
Firewall	Enable RTSP ALG	Enable RTSP application level gateway (ALG) to ensure that RTP ports are dynamically opened.

Table 10. DUT Variables

## Step-by-Step Instructions

1. Start IxLoad. In the main window, the **Scenario Editor** window appears. All test configurations will be done here.
2. To get familiar with the IxLoad GUI, see the Getting Started Guide section. For a step-by-step workflow, see Annex A.
3. Add a server NetTraffic. Add 24 IP addresses.
4. Add IPTV server activity to the server NetTraffic. Refer to the Test Tool Variables section to configure a new VoD stream that creates RTP streams of 1 packet per second, using large packet sizes.

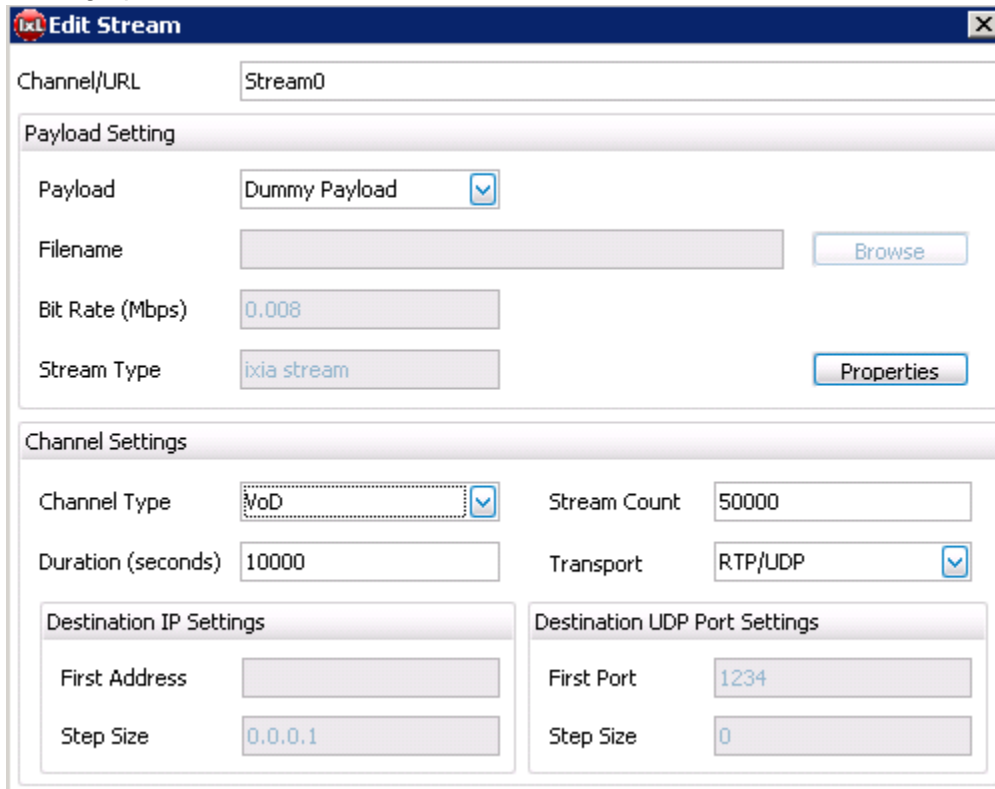


Figure 32. Configuration for creating a new hardware RTP stream

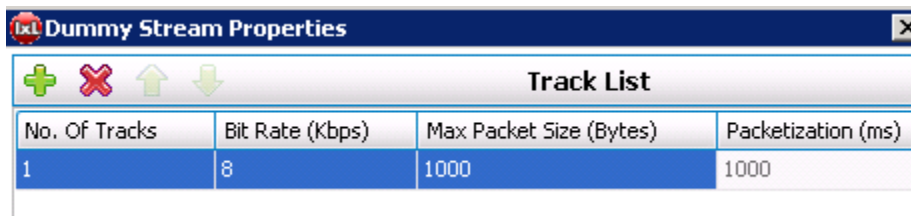


Figure 33. Configuration of tracks for hardware RTP streams

5. Add a client NetTraffic. Add 1200 IP addresses.

## Test Case: Large Scale RTP Video – 600,000 Streams

6. Add IPTV client activity to the client NetTraffic. Refer to the Test Tool Variables section to configure the client activity. Set the transport to RTP/UDP in the **Unicast Signaling** tab.

The {Play Media Static} command configuration is shown here.

Command Properties for 'Play Media Static 2'

Play Media Static

Server Address: Traffic2\_High\_per\_RTP\_server:554

Media: Stream0

Duration: 10000 - 10000 sec

Figure 34. Configuration for playback or Hardware RTP streams

From the **Unicast Signaling** tab, select the 'Enable Graceful Rampdown' check box.

Graceful Rampdown

Enable Graceful Rampdown

Figure 35. Graceful rampdown configuration for RTP streams

7. After both the NetTraffics are configured, the default traffic map configuration must be changed to allow the 1200 client IP addresses to map specifically to the 24 server IP addresses. Select the DOT from the client NetTraffic to server Nettraffic to start and from the **Traffic Map** list, select **Custom**.

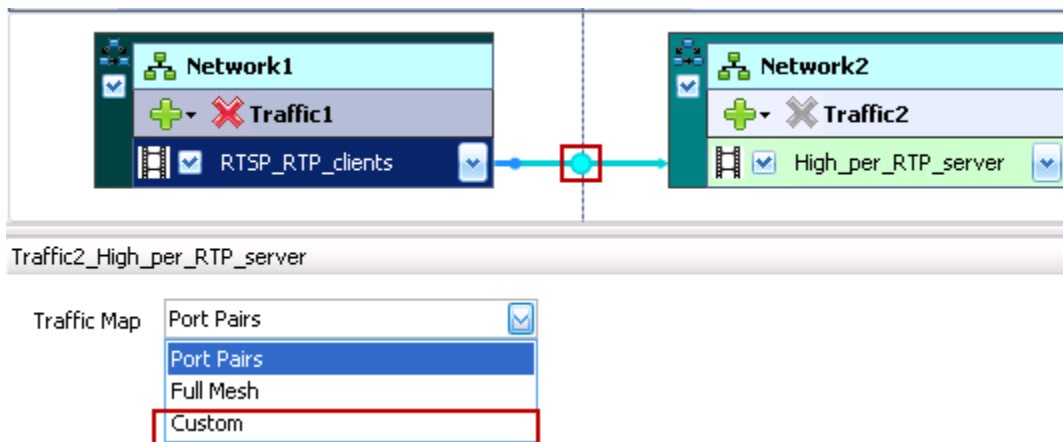


Figure 36. Choosing the Traffic Map as Custom



Test Case: Large Scale RTP Video – 600,000 Streams

- Split the client IPs from 1200 to 24 x 50. Select **Range** and right-click. Next, click **Split** from the shortcut menu.

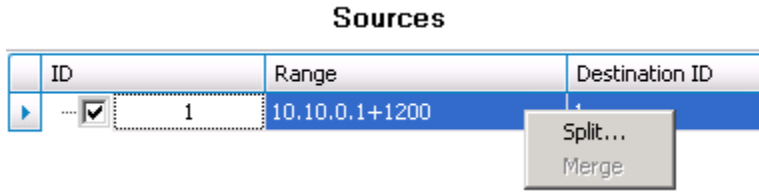


Figure 37. Using Traffic Map feature to split client IP ranges for use on Acceleron-NP for hardware RTP

Choose 24 x 50 IPs to split.

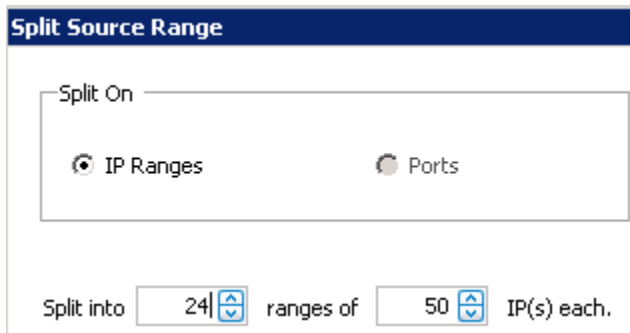


Figure 38. Using Traffic Map feature to split server IP ranges for use on Acceleron-NP for hardware RTP

- Split the server IPs from 24 to 1 x 24 in a similar way as for client.
- To reapply the 'IP Range Pairs' and take effect, choose **IP Range Mesh** from the **Mapping Type** list, and then switch back to **IP Range Pairs**. Click **Apply** to confirm all custom mapping changes. See the final mapping in the following image.

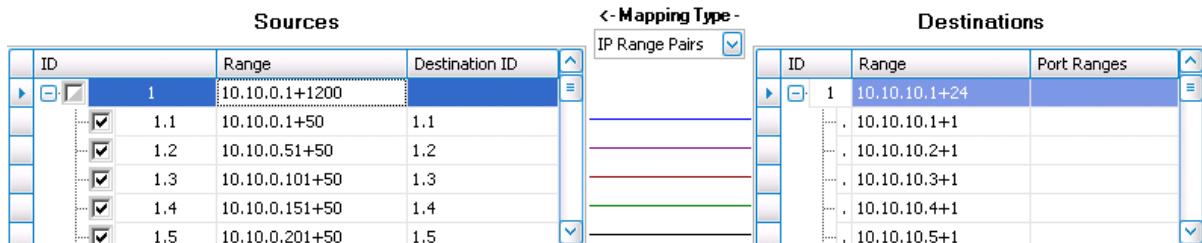


Figure 39. Mapping client and server IP ranges using custom Traffic Map feature

## Test Case: Large Scale RTP Video – 600,000 Streams

- Set the test objective. Configure simulated user as the test objective and set the objective value to 600,000.

Timeline and Objective

Network Traffic Mapping	Objective Type	Objective Value
New Traffic Flow		
<ul style="list-style-type: none"> <li>Traffic1@Network1</li> </ul>	Simulated Users	600,000
<ul style="list-style-type: none"> <li>RTSP_RTP_clients</li> </ul>	Simulated Users	600,000
<ul style="list-style-type: none"> <li>Traffic2@Network2</li> </ul>	N/A	N/A
<ul style="list-style-type: none"> <li>High_per_RTP_server</li> </ul>	N/A	N/A

Figure 40. Configuration for 600,000 Simulated User Test Objective

- Create a timeline that uses 'Smooth Users/Interval' as the ramp-up type. Set the value to 15000 (1250/port x 12 ports/CPU on Accelaron-NP).

Timeline

---

Timeline

Ramp Up Type: Smooth Users/Interval

Ramp Up Value: 15,000

Figure 41. Ramp Up configuration for smooth users per interval

- Set the test duration to 10000 seconds.
- In Port Assignments, add test ports as follows: 1 Accelaron-NP card to the client NetTraffic and 1 Accelaron-NP card to the server Nettraffic.
- Set both the cards to the 10G- AGG mode. Right click the card on the chassis chain and set the Aggregation to '10G Aggregated.'

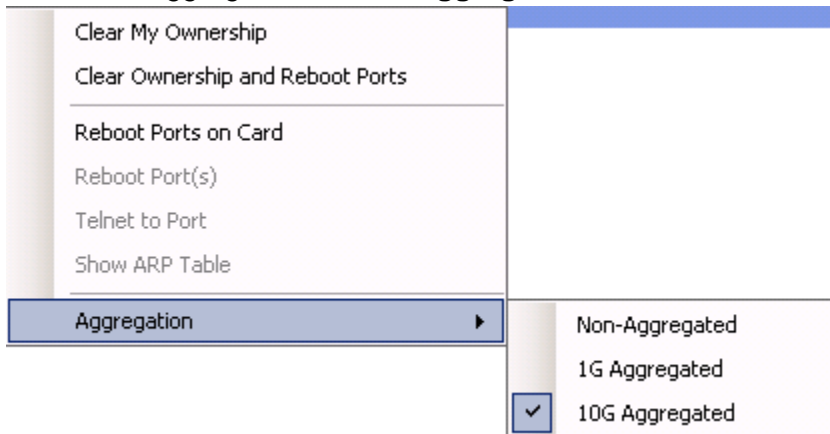


Figure 42. Setting Accelaron-NP to use 10G-Aggregation

- Run the test.

## Results Analysis

Metric	Key Performance Indicators	Statistics View
Performance Metrics	Total Connections, Number of Simulated Users, Throughput	IPTV Client – Objectives IPTV Client – Data Rates IPTV Client - VoD Channel Requests
Application Level Transactions  Application Level Failure Monitoring	Presentations Active, Presentations Playing, Paused, Requested, Successful, Failed, and Playback Successful	IPTV Client – Presentations IPTV Client – Latency Distribution IPTV Client - RTP Jitter IPTV Client - RTP Loss

**Table 11. Results Analysis for 600,000 RTP streams**

The following table provides complete theoretical analysis of various scenarios that can be configured to run large scale RTP tests.

$$\text{Bit Rate (Kbps)} = \text{packet size(Bytes)} * 8 / \text{packetization (ms)}$$

$$\text{PPS (packets / sec)} = 1000 / \text{packetization (ms)}$$

$$\text{Stream count (per port)} = 750 * 1000 / \text{Bit Rate (kbps)}$$

$$\text{Max RTP streams} = \text{Stream Count} * 12$$

Packet Size	RTP Inter-packet gap Packetization (ms)	Bit Rate (Kbps)	PPS / stream	Stream count (per port)	Max RTP streams	Max Bit Rate on 10G	Max Packet Rate on 10G
1000	1000	8	1	50000	600,000	5.1G	600k

**Table 12. Theoretical samples for setting up large scale RTP streams**

The performance listed is for a pair of Acceleron-NP load modules.

## Real-time Statistics

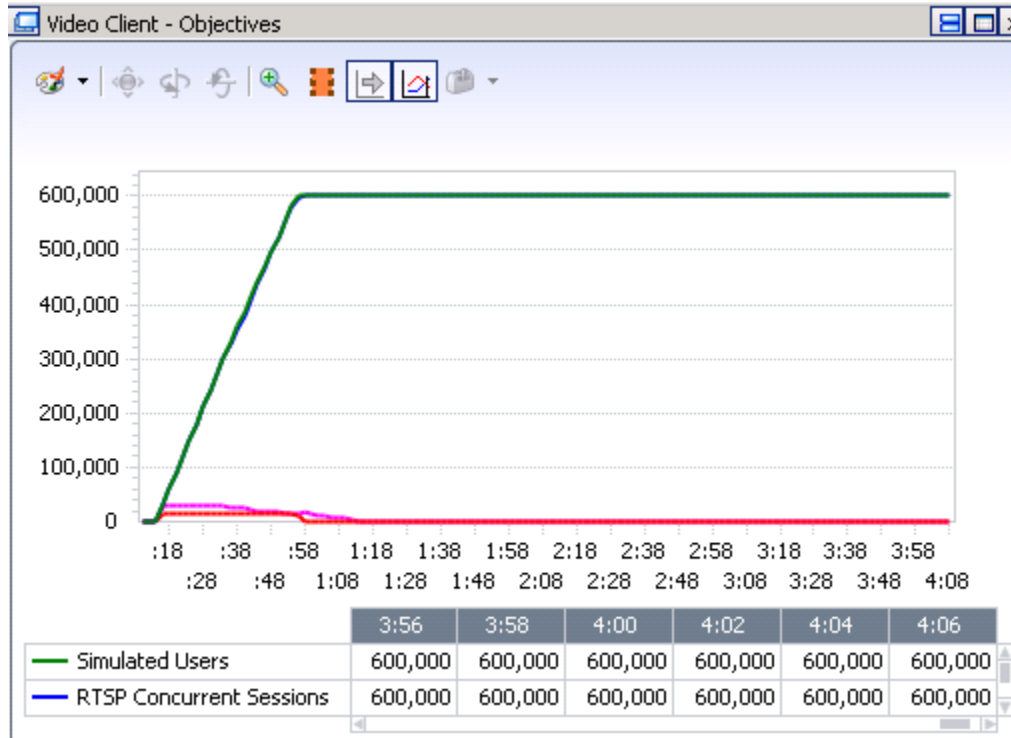


Figure 43. Video Client – Objectives view

The **Video Client - Objectives** view provides real-time user and RTSP session information.

Test Case: Large Scale RTP Video – 600,000 Streams

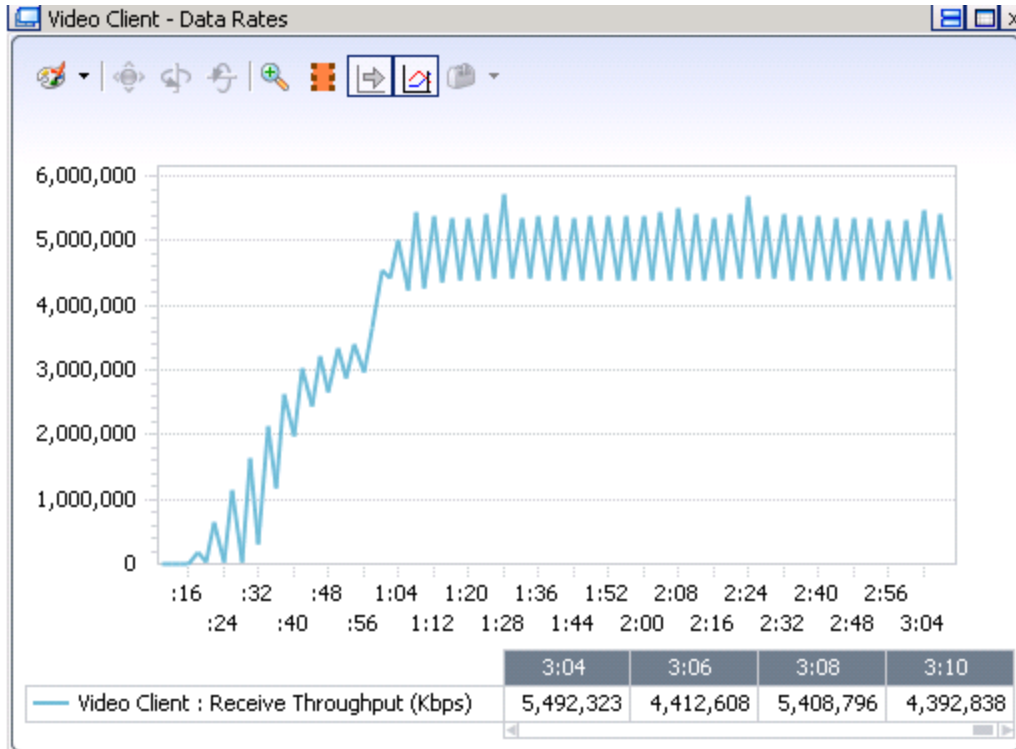


Figure 44. Video Client – Data Rates view

The **Video Client - Data Rates** view provides L7 video received throughput.

Stat Name	Link State	Line Speed	Valid Frames Received Rate	Bits Received Rate (Kbps)
10.200.134.136/Car...	Link Up	10GE LAN	600,033	5,136,266.952

Figure 45. L2-3 Stats for Client Ports view

The **L2-3 Stats for Client Ports** view provides the packets per second 'Valid Frames Received Rate' and the actual frame throughput rate (600,000 PPS).

## Test Case: Large Scale RTP Video - 4.8 Million Packets per Second

### Overview

There is a rapid growth of multimedia capable smart phones and Internet enabled devices. RTSP with RTP/UDP is a very common way to deliver streaming media to millions of such devices.

Firewalls and edge routing devices that handle mobile IP networks must be able to process millions of new TCP sessions per second, at rates of millions of small packets per second. As the number of connections per second increase, the load of the CPU and memory utilization increases. In many cases, performance degrades significantly if the firewall does not have optimizations in place for handling small packets specifically.

IxLoad's brilliant Acceleron-NP load module uses custom Network Processors (NP) to generate very large scale of RTP sessions with the highest packets per second. The NP cores can sustain 4.8 million packets per second and maintain multi-gigabit of throughput with small packets to create a realistic scenario.

## Test Case: Large Scale RTP Video - 4.8 Million Packets per Second

The following table highlights the performance levels achieved by a pair of Acceleron-NP.

Metric	Performance (Pair of Acceleron-NP)
CPS Without Media	24,000
CPS With Media	10,000
Concurrent sessions with 1 packet/sec	600,000
Concurrent sessions with larger packets/sec/stream	96,000 for 50 packets/sec per stream
Configurable Audio + Video RTP Stream per RTSP Session	Independent configuration for audio and video streams
128 kps, 1400 bytes/packet = 12 packets per second	75,000 streams 10 Gbps
128 kps, 600 bytes/packet = 27 packets per second	75,000 streams 10 Gbps

**Table 13.** Performance levels achieved on Acceleron-NP pair

### Objective

This test case highlights how to set up a test profile that uses a pair of Acceleron-NP to create 96,000 simultaneous RTSP + RTP/UDP media sessions with very small packet sizes.

Unlike the previous test case of achieving 600,000 RTSP sessions, the primary goal of this is to generate 4.8 million small packets (on one pair of Acceleron-NP) per second and maintain multi-gigabit per second of throughput. This scenario loads the device and tests that its CPU and packet processing duties are able to handle such extreme loads.

## Setup

The following scenario uses two Acceleron-NP load modules running in 10G aggregated mode, connected directly to two ports on the firewall/RTP gateway DUT.

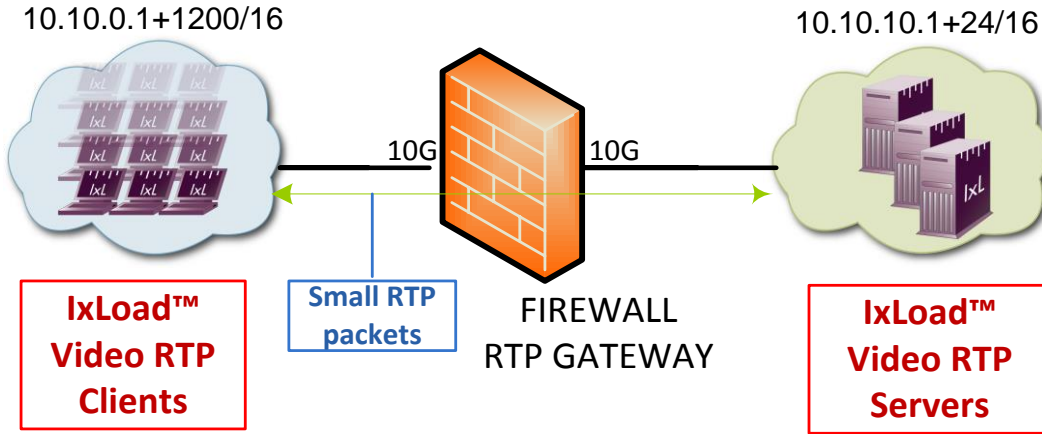


Figure 46. Large-scale RTP setup scenario for 4.8 Million packets per second

To achieve maximum packets per second, the test configuration creates RTP sessions with small packet sizes (frames). If the per-RTP session packet size is increased, the total packets per second rate reduces. This can be determined very accurately. See the **Results Analysis** section for examples.

## Test Variables

### Test Tool Variables

Parameters	Description
IPTV Client Network	12,000 IP addresses; 100 per port, 12 port test.
TCP parameters	TCP RX and TX buffer at 4096 bytes.
IPTV Client Activity parameters	<p>On the <b>Unicast Signaling</b> tab, choose Transport as RTP/UDP. Use {Play Media Static} command and configure as follows:</p> <p><a href="#">Command Properties for 'Play Media Static 2'</a></p> <div style="border: 1px solid #ccc; padding: 5px;"> <p>Play Media Static</p> <p>Server Address <input type="text" value="Traffic2_High_per_RTP_server:554"/></p> <p>Media <input type="text" value="Stream0"/></p> <p>Duration <input type="text" value="10000"/> - <input type="text" value="10000"/></p> </div>

Figure 47. Configuration for playback or Hardware RTP streams



Test Case: Large Scale RTP Video - 4.8 Million Packets per Second

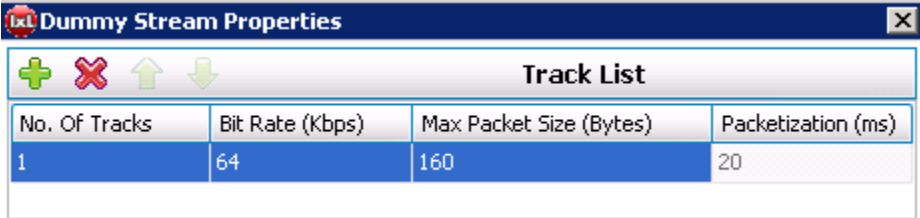
Parameters	Description
	Set Duration to 10000 or more seconds.
IPTV Server Network	24 IP addresses; 2 port test port (required).
IPTV Server Activity parameters	<p>Add VoD Channel Type. Choose 'Dummy Payload' and configure as follows:</p>  <p style="text-align: center;">Figure 48. Configuration of tracks for hardware RTP streams</p> <p>Set Duration to 10000 or more. Set Stream Count to 8000. Set Transport to RTP/UDP. (64 Kbps with 160 byte/packet is 50 packets per second) (8000 streams x 12 ports = 96000 RTP sessions/users) (96000 users x 50 PPS = 4.8 million PPS)</p>
RTP Hardware Acceleration	<p>In Video Client Activity, on the <b>Unicast Signaling</b> tab, select HW Acceleration. In IPTV Video Server Activity, on the <b>Advanced Options</b> tab, enable HW Acceleration.</p>

Table 14. – Test Tool Variables

DUT Test Variables

Device(s)	Variation	Description
Firewall	Enable RTSP ALG	Enable RTSP application level gateway (ALG) to ensure RTP ports are dynamically opened

Table 15. DUT Variables

## Step-by-Step Instructions

1. Start IxLoad. In the main window, the **Scenario Editor** window appears. All test configurations will be done here.
2. To get familiar with the IxLoad GUI, see the Getting Started Guide section. For a step-by-step workflow, see Annex A.
3. Add a server NetTraffic. Add 24 IP addresses.
4. Add IPTV server activity to the server NetTraffic. Refer to the Test Tool Variables section to configure a new VoD stream that creates RTP streams of 1 packet per second, using large packet sizes.

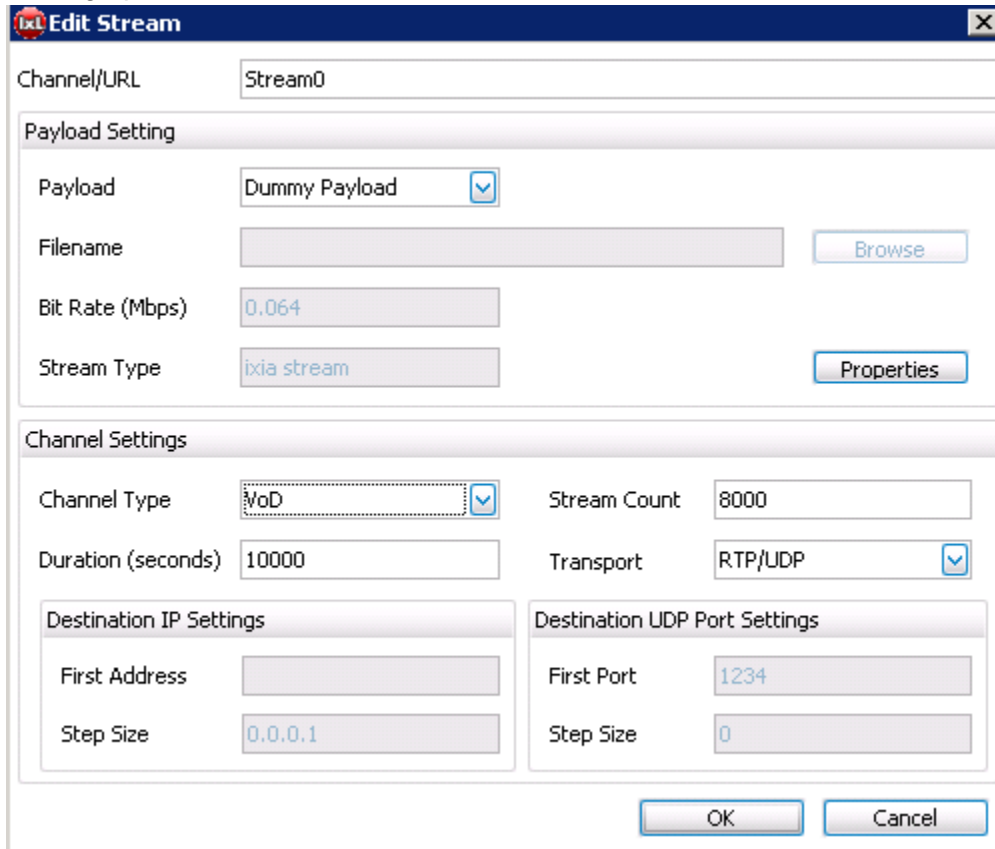


Figure 49. Configuration for creating a new hardware RTP stream

No. Of Tracks	Bit Rate (Kbps)	Max Packet Size (Bytes)	Packetization (ms)
1	64	160	20

Figure 50. Configuration of tracks for hardware RTP streams

5. Add a client NetTraffic. Add 1200 IP addresses.

Test Case: Large Scale RTP Video - 4.8 Million Packets per Second

- Add IPTV client activity to the client NetTraffic. Refer to the Test Tool Variables section to configure the client activity. Set the Transport to RTP/UDP in the **Unicast Signaling** tab.
- The {Play Media Static} command configuration is shown here.

Command Properties for 'Play Media Static 2'

Play Media Static	
Server Address	Traffic2_High_per_RTP_server:554
Media	Stream0 <span>Media Description</span>
Duration	10000 - 10000 sec

Figure 51. Configuration for playback or Hardware RTP streams

- In the **Unicast Signaling** tab, select the **Enable Graceful Rampdown** option.

Graceful Rampdown	
<input checked="" type="checkbox"/>	Enable Graceful Rampdown

Figure 52. Graceful rampdown configuration

- Set the test objective. Configure simulated user as the test objective and set the objective value to 96,000.

Timeline and Objective

Network Traffic Mapping	Objective Type	Objective Value
<ul style="list-style-type: none"> <li>New Traffic Flow                             <ul style="list-style-type: none"> <li>Traffic1@Network1</li> <li>RTSP_RTP_clients</li> <li>Traffic2@Network2</li> <li>High_per_RTP_server</li> </ul> </li> </ul>	<ul style="list-style-type: none"> <li>Simulated Users</li> <li>Simulated Users</li> <li>N/A</li> <li>N/A</li> </ul>	<ul style="list-style-type: none"> <li></li> <li>96,000</li> <li>96,000</li> <li>N/A</li> <li>N/A</li> </ul>

Figure 53. Configuration for 96,000 Simulated User Test Objective

Test Case: Large Scale RTP Video - 4.8 Million Packets per Second

10. Create a timeline that uses 'Smooth Users/Interval' as the ramp-up type. Set the value to 1200 (max per 1 CPU/port on Accelaron-NP).

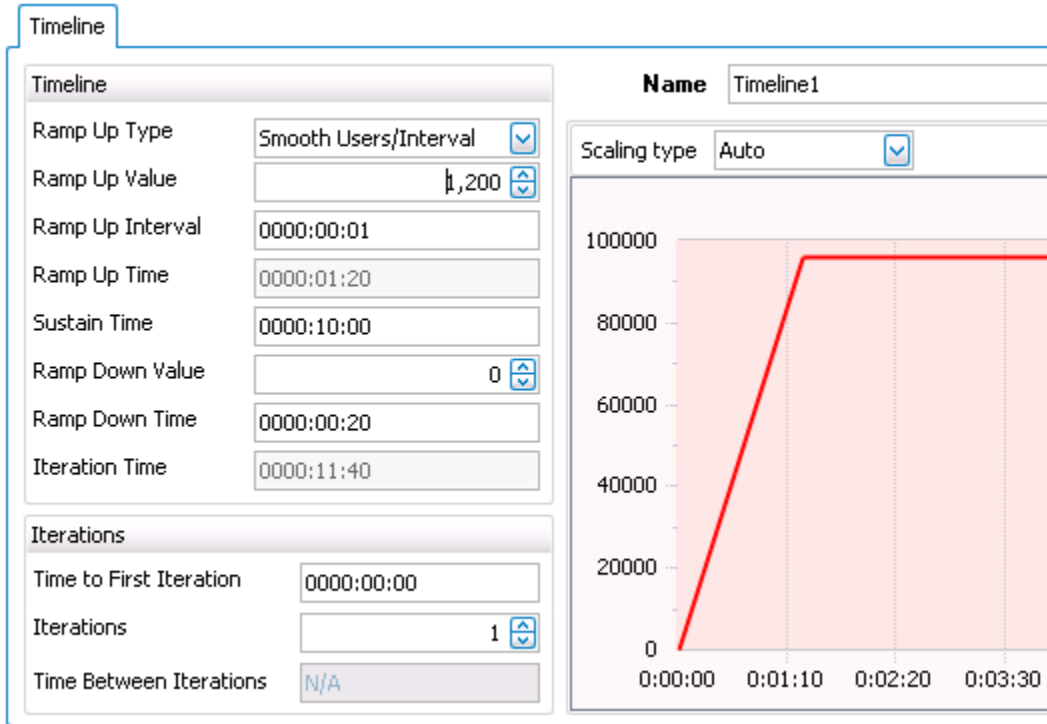


Figure 54. Ramp Up configuration for smooth users per interval

11. Set the test duration to 10000 seconds.
12. Add test ports as follows: one Accelaron-NP card to the client NetTraffic and one Accelaron-NP card to the server Nettraffic.
13. Set both cards to the 10G- AGG mode. Right-click the card on the chassis chain and set the aggregation to **10G Aggregated**.

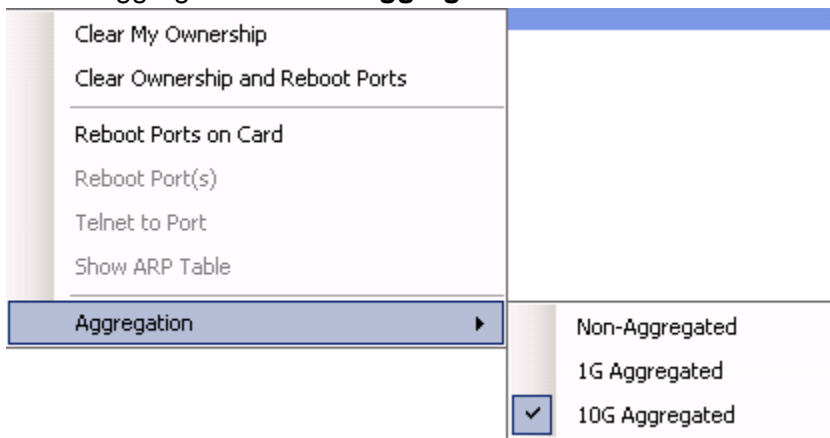


Figure 55. - Setting Accelaron-NP to use 10G-Aggregation

14. Run the test.

## Results Analysis

Metric	Key Performance Indicators	Statistics View
Performance Metrics	Total connections, Number of Simulated Users, Throughput	IPTV Client – Objectives IPTV Client – Data Rates IPTV Client - VoD Channel Requests
Application Level Transactions  Application Level Failure Monitoring	Presentations Active, Presentations Playing, Paused, Requested, Successful, Failed, and Playback Successful	IPTV Client – Presentations IPTV Client – Latency Distribution IPTV Client - RTP Jitter IPTV Client - RTP Loss

**Table 16. Results Analysis for 4.8Million packets per second test**

The following table provides complete theoretical analysis of various scenarios that can be configured to run large scale RTP tests.

$$\text{Bit Rate (Kbps)} = \text{packet size(Bytes)} * 8 / \text{packetization (ms)}$$

$$\text{PPS (packets / sec)} = 1000 / \text{packetization (ms)}$$

$$\text{Stream count (per port)} = 750 * 1000 / \text{Bit Rate (kbps)}$$

$$\text{Max RTP streams} = \text{Stream Count} * 12$$

Packet Size	RTP Inter-packet gap Packetlization (ms)	Bit Rate (Kbps)	PPS/stream	Stream count (Per port)	Max RTP streams	Max Bit Rate on 10G	Max Packet Rate on 10G
160	20	64	50	8000	96000	8.8 G	4.8M
80	20	32	50	8000	96000	5.7 G	4.8M
320	20	128	50	5000	60000	~ 10 G	3 million
320	10	256	100	3000	36000	~ 10 G	3.6 million
320	5	512	200	1500	18000	~ 10 G	3.6 million
400	1	3200	1000	200	2400	~ 10 G	2.4 million

**Table 17. Theoretical samples for setting up large scale RTP streams**

The performance listed in the preceding table is for a pair of Acceleron-NP load modules.

## Real-time Statistics

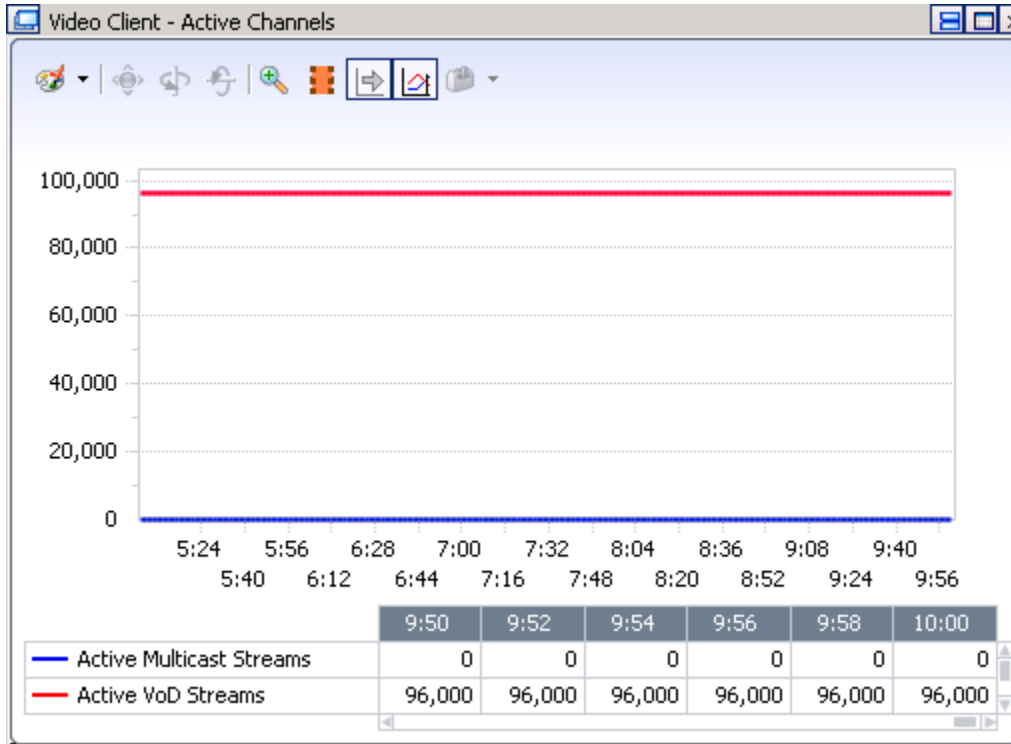


Figure 56. Video Client – Active Channels view

The **Video Client - Active Channels** view provides real-time user and RTSP session information.

Test Case: Large Scale RTP Video - 4.8 Million Packets per Second

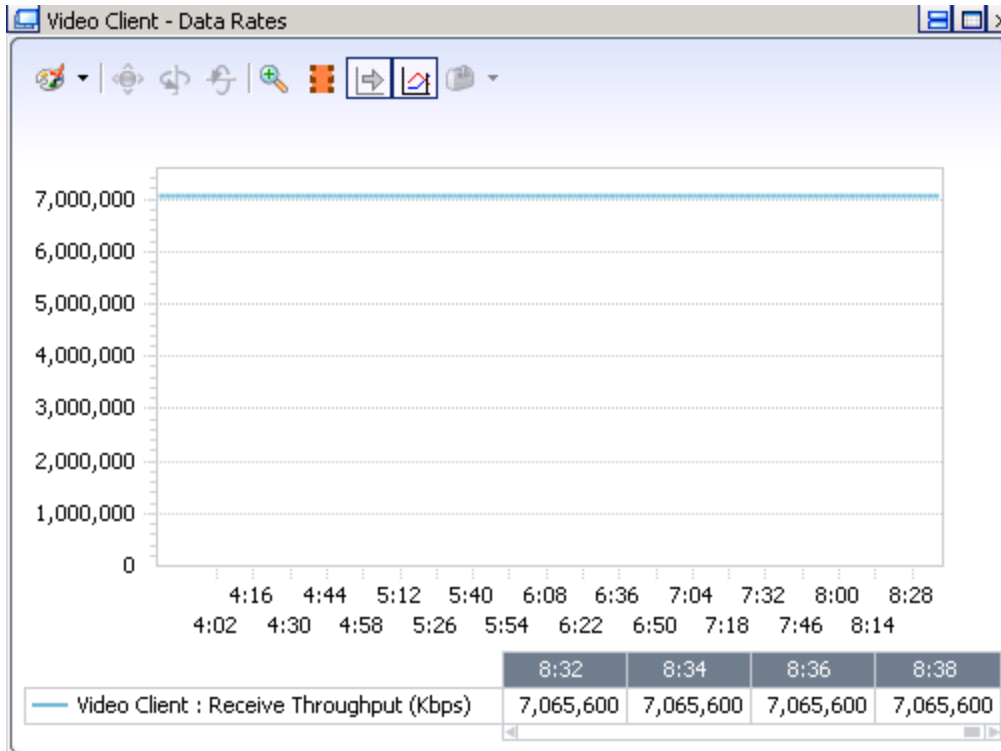


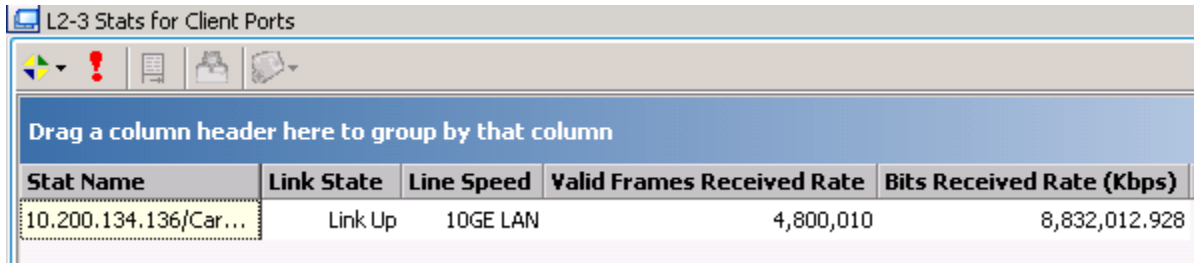
Figure 57. Video Client – Data Rates view

The **Video Client - Data Rates** view shows L7 video received throughput.

## Test Case: Large Scale RTP Video - 4.8 Million Packets per Second

The **L2-3 Stats for Client Ports** view provide the packets per second (Valid Frames Received Rate) and the actual frame throughput rate.

For a **64 kbps RTP stream**, the maximum PPS is 4.8 million. The received bitrate is 8.8 Gbps.

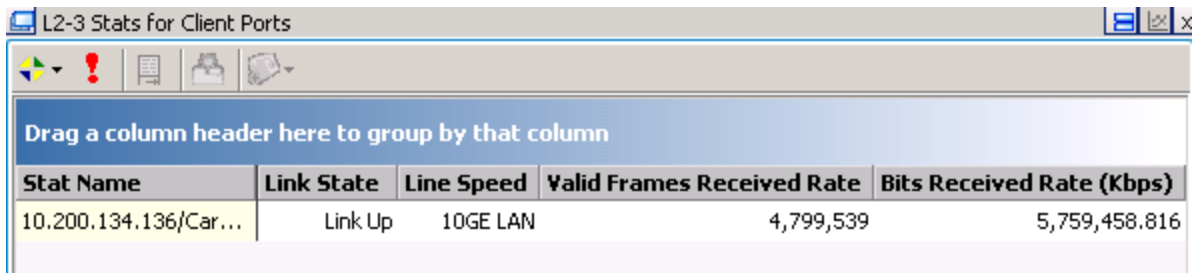


The screenshot shows a window titled "L2-3 Stats for Client Ports". Below the title bar is a toolbar with navigation and action icons. A blue instruction bar reads "Drag a column header here to group by that column". Below this is a table with the following data:

Stat Name	Link State	Line Speed	Valid Frames Received Rate	Bits Received Rate (Kbps)
10.200.134.136/Car...	Link Up	10GE LAN	4,800,010	8,832,012.928

Figure 58. L2-3 Stats for Client Ports view

For a **32 kbps RTP stream**, the maximum PPS is 4.8 million. The received bitrate is 5.8 Gbps.



The screenshot shows a window titled "L2-3 Stats for Client Ports". Below the title bar is a toolbar with navigation and action icons. A blue instruction bar reads "Drag a column header here to group by that column". Below this is a table with the following data:

Stat Name	Link State	Line Speed	Valid Frames Received Rate	Bits Received Rate (Kbps)
10.200.134.136/Car...	Link Up	10GE LAN	4,799,539	5,759,458.816

Figure 59. L2-3 Stats for Client Ports view





## Test Case: Basic Flash™ Player Emulation

### Overview

Adobe's Flash Player is a pioneer in bringing rich and interactive experience to a majority of PC and MAC systems worldwide. It supports an advanced set of capabilities and technologies that enable streaming HD/H.264 and a variety of rich Internet applications (RIA). It is not an RFC based protocol; it is proprietary to Adobe and binary in nature.

Ixia is an exclusive test vendor to license Flash technology from Adobe since 2009.

Some of the protocols used include the following:

- RTMP: Core protocol, unencrypted Real-Time Messaging Protocol. Uses TCP port 1935.
- RTMPT: This protocol is RTMP tunneled over HTTP; the RTMP data is encapsulated as valid HTTP. The default port is 80.
- RTMPS: This protocol is RTMP over SSL. SSL is a protocol for enabling secure communications over TCP/IP. The default port is 443.
- RTMPE: This protocol is an encrypted version of RTMP. RTMPE is faster than SSL, and does not require certificate management.
- RTMPTE: This protocol is RTMPE with an encrypted tunneling connection. The default port is 80.

To learn more about Flash Player technology, go to the Flash Media Server site:

[http://help.adobe.com/en\\_US/FlashMediaServer/3.5\\_TechOverview/WS5b3ccc516d4fbf351e63e3d119ed944a1a-7ffa.html](http://help.adobe.com/en_US/FlashMediaServer/3.5_TechOverview/WS5b3ccc516d4fbf351e63e3d119ed944a1a-7ffa.html)

A simplified protocol exchange may look as follows:

- TCP connection to FMS server, Dest TCP/1935 (default).
- Flash Handshake message exchange.
- During the handshake, an application level NetConnection is created.
- RTMP request and response messages are used for playback control.

## Objective

Create a test profile in IxLoad to act as a Flash Player to stream content from a Flash Media Server (FMS) 3.5 running on Windows 2003 server.

The test case will highlight all the capabilities of stream playback using PLAY command.

## Setup

The following scenario is used: Windows 2003 Server running Flash Media Server 3.5 to host a Flash compatible content, IxLoad 5.10 Flash Player Activity with RTMP and RTMPT.

One Ixia test port is connected directly to Flash Media Server (FMS). The topology shown in the following figure is representative of a more complex scenario with optional proxy that is supported with RTMPT and a CDN infrastructure that hosts the content that is sent to the FMS.

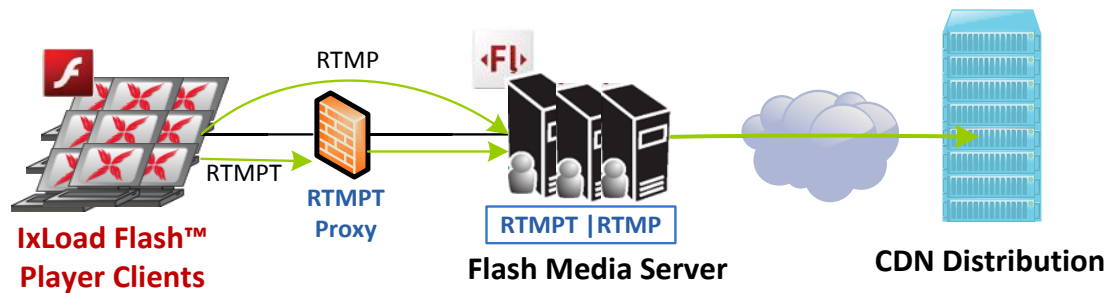


Figure 60. Flash Player setup to test Flash Media Server infrastructure

## Test Variables

### Test Tool Variables

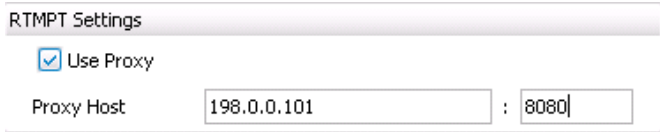
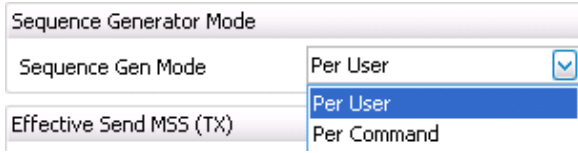
Parameters	Description
Flash Player Clients	100 IP addresses or more, use sequential or 'use all' IP addresses.
Flash Player parameters	Keep default settings.
TCP parameters	TCP RX and TX buffer at 32768 bytes.
Flash Player client command list	The step by step will outline how to configure basic playback and set up user actions such as SEEK, PAUSE, and RESUME to create realistic user scenario.
RTMPT Proxy support	<p>If clients connect through a content switch or proxy, IxLoad can send all RTMPT packets to this configured <b>IP:port</b>.</p>  <p style="text-align: center;"><b>Figure 61. Configuring RTMPT proxy</b></p>
Use of sequence generator	Sequence generator is used above to expand <b>video[1-5].m4v</b> to video1.m4v through video5.m4v. The way in which selection is made is based on the Mode.
Sequence Generator Mode	<p>If <b>Per-User</b> is selected, all users start from the same index (for example, sample1.mp4 used by all users at start) and users moves through the index at the end of each stream duration.</p> <p>If <b>Per-Command</b> is selected, all users stagger to start off uniquely (for example, user1 picks sample1.mp4, user2 picks sample2.mp4) and each user moves through this staggered (unique) list based on stream length or configured time.</p>  <p style="text-align: center;"><b>Figure 62. – Choosing the sequence generator mode</b></p>

Table 18. Test Tool Variables

### DUT Test Variables

Device(s)	Variation	Description
Flash Media	Enable RTMP and	Enable anonymous RTMP and RTMPT controls

## Test Case: Basic Flash Player™ Emulation

Server	RTMPT	and note listening ports.
--------	-------	---------------------------

Table 19. DUT Variables

### Step-by-Step Instructions

1. Start IxLoad. In the main window, the **Scenario Editor** window appears. All test configurations will be done here.
2. To get familiar with the IxLoad GUI, see the Getting Started Guide section.
3. Add the client NetTraffic object. Configure the client network with total IP count, gateway, and VLAN, if used.
4. For a step-by-step workflow, see Annex A.
5. Configure the Flash Player client. Add the **Flash Player** activity to the client NetTraffic.
6. For basic **PLAYBACK**, configure the command list as follows:



7. Add a **PLAY** command and configure the following parameters:
  - Choose RTMP or RTMPT protocol. For RTMP, the Flash Player will connect to TCP port 1935. For RTMPT, the default port is TCP/80.
  - Enter the destination server by IP or hostname (enable DNS to use this). If the server is listening on a non-standard port, it can be specified as IP:port.
  - Enter the Application Name as 'all.' Application name is used by FMS for capabilities. This is unique to an FMS and must be configured correctly.
  - Specify the Media as 'mp4:Extremists1.m4v.' Note the notation. See the notes section later on how to correctly type the media name.
  - The Play Duration can be to play until end when 'Till End' is selected, or the Range option can be used to specify a range of time. Each user will randomly select a duration within this range
  - The Stream Type must also be known. Recorded (that is, VoD) or Live are both supported and must be specified correctly. A fallback mechanism also exists with the 'Try Live then Recorded' option.

## Test Case: Basic Flash Player™ Emulation

Command Properties for 'Play Media 2'

PLAY

Server Address  ://  /

Media

Stream Type

Recorded  Live  Try live then recorded

Play Duration

Till End  Range (secs)  -

Figure 63. Configuration for PLAY command for basic Flash Player playback

8. Add a **STOP** command to ensure graceful teardown of sessions at the end of the test or when the test is stopped manually.
9. Having set up the Flash Player activity, configure simulated user as the test objective and set the objective value to the desired number of users.
10. Add test port resources and run the test. Refer to the Results Analysis and the Troubleshooting and Diagnostics section for further information.

### Note about Media Name and Format to Use

- FLV: Just the name is used without the extension, for example, 'Extremists'
- MP4: Use the format 'mp4:<full-file-name>.'
- MP3: Use the format 'mp3:<full-file-name>.'

## Results Analysis

Metric	Key Performance Indicators	Statistics View
Performance Metrics	User Count, Active Stream, Played, Paused RX/TX Throughput, Audio/Video/Data Throughput	FlashPlayer Client – Objectives FlashPlayer Client - Throughput
Application Level Transactions  Application Level Failure Monitoring	Handshake and NetConnection, Requested, Successful, Failed Packet Counts for Audio, Video, Data Key Error Codes	FlashPlayer Client - Handshake  FlashPlayer Client - AVD Packets FlashPlayer Client - Errors
Streaming Quality	Play Latency, Netconnect Latency	FlashPlayer Client - Latency
TCP Statistics	TCP SYN, SYN-ACK, Connections Established, Failed	FlashPlayer Client - TCP Connections

**Table 20. Results Analysis for Flash Player content playback**

## Real-time Statistics

Flash is a binary protocol; packets can be captured, but the flows are not human readable easily (for example, when using Ethereal to follow a TCP stream). Flash Player activity in IxLoad provides TCP level statistics, all Handshake and Netconnection statistics, and RTMP level commands. It also provides critical error codes sent by the server (for example, to identify if an application name is incorrect, or if a wrong filename is requested), and user experience metrics such as latency.

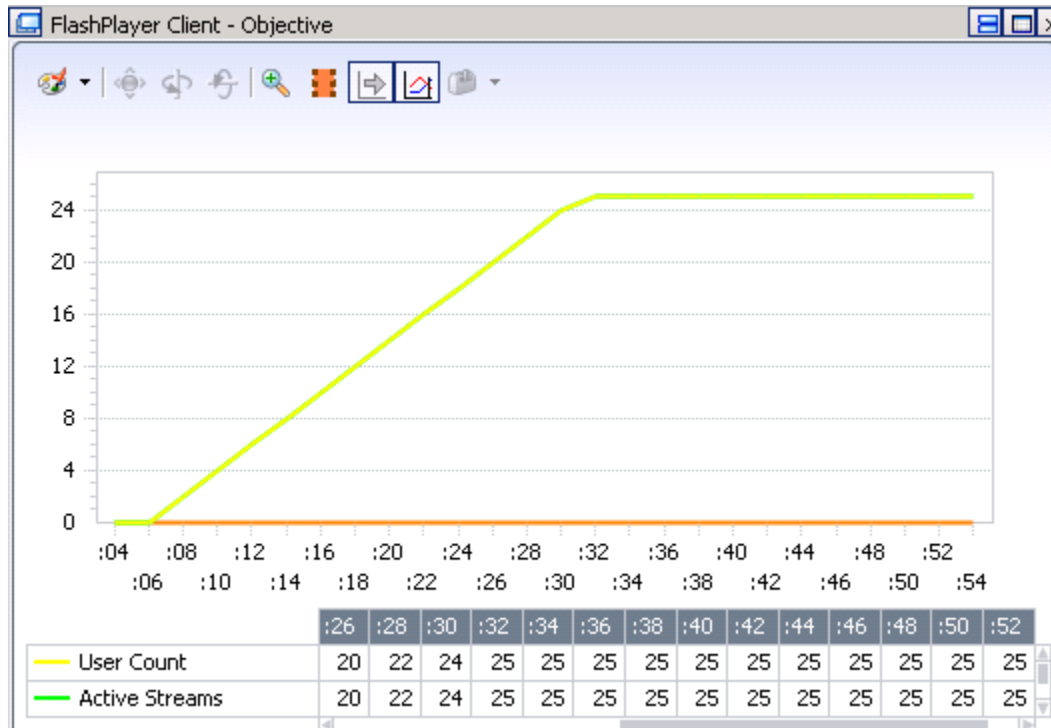


Figure 64. – FlashPlayer Client – Objective view

The **FlashPlayer Client - Objective** view is a real-time view showing active users and streams. The active number of sessions being reported on the DUT should correlate to this view.



## Test Case: Basic Flash Player™ Emulation

The **FlashPlayer Client – Throughput Objectives** view provides visibility into the audio, video, RX, and TX throughput. Use this view to verify that the number of active streams and the total receive bandwidth correlate, if the average bitrate for the streams are known.

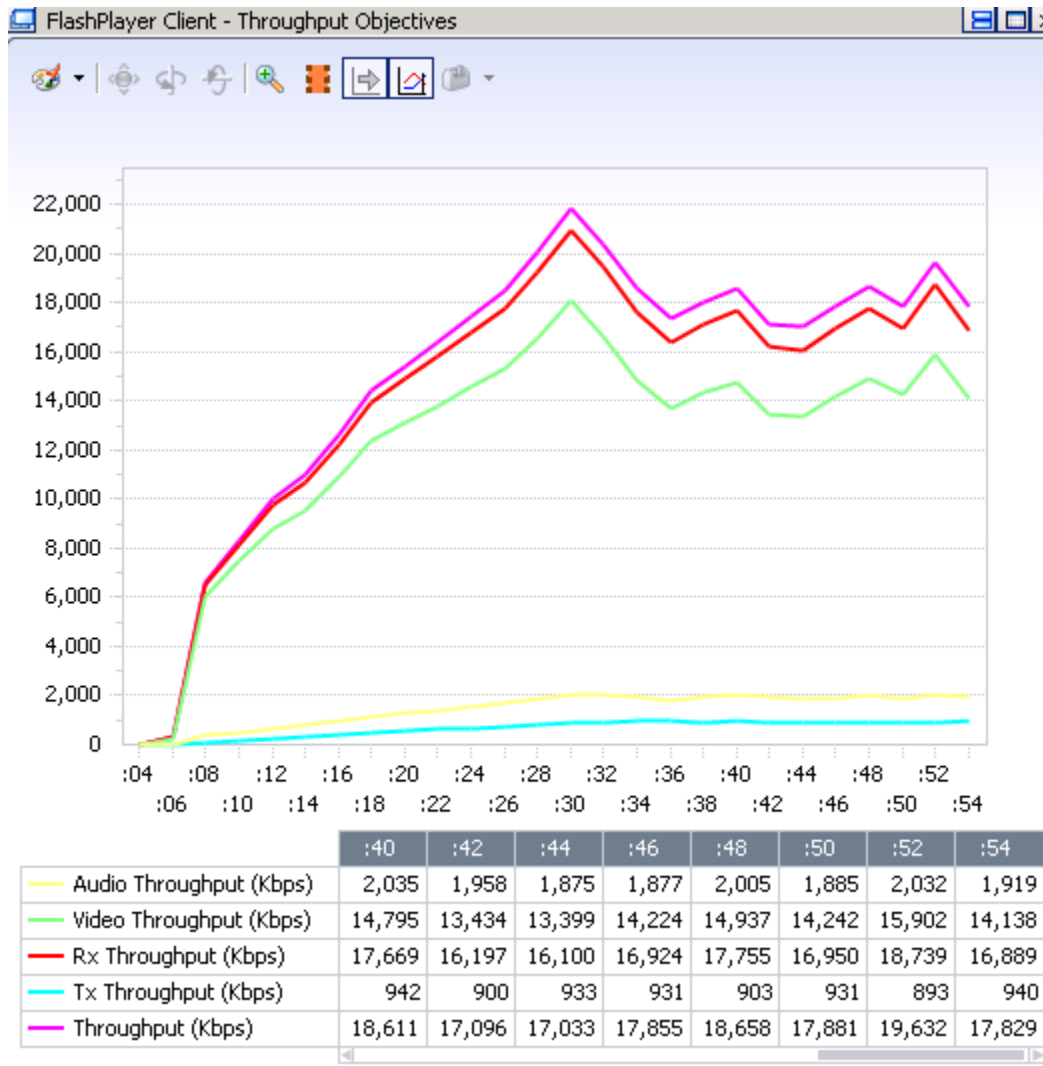


Figure 65. FlashPlayer Client – Throughput Objectives view

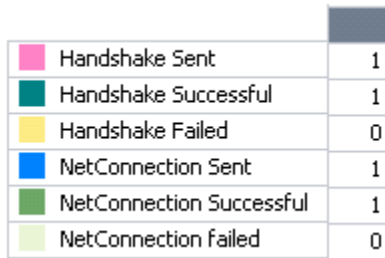
Complete TCP statistics are available in the **FlashPlayer - TCP Connections** view. If there is a connection or reachability issue, this view will show failures.

TCP SYN Sent	1
TCP SYN_SYN-ACK Received	1
TCP Connections Established	1

Figure 66. FlashPlayer Client – TCP Connections view

## Test Case: Basic Flash Player™ Emulation

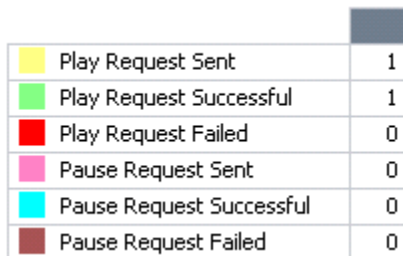
The **FlashPlayer - Handshake** view shows the application level handshake, followed by a NetConnection message. If there is an issue interoperating with the Flash server, this view will indicate if its a handshake or NetConnection failure.



Handshake Sent	1
Handshake Successful	1
Handshake Failed	0
NetConnection Sent	1
NetConnection Successful	1
NetConnection failed	0

Figure 67. FlashPlayer Client – Handshake view

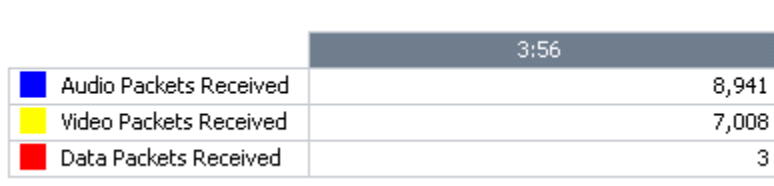
The **FlashPlayer - Command** view shows individual application level events such as Play, Pause, Seek, and Resume actions.



Play Request Sent	1
Play Request Successful	1
Play Request Failed	0
Pause Request Sent	0
Pause Request Successful	0
Pause Request Failed	0

Figure 68. FlashPlayer Client – Command view

The **FlashPlayer - Audio Video Data Packets** view processes header information present in the responses to count the total number of audio, video, or data packets.

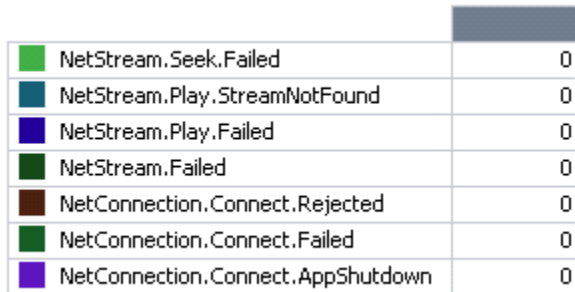


	3:56
Audio Packets Received	8,941
Video Packets Received	7,008
Data Packets Received	3

Figure 69. FlashPlayer Client – Audio Video Data Packets view

## Test Case: Basic Flash Player™ Emulation

The **FlashPlayer - Errors** view provides error codes to identify specific issues in streaming from a Flash Media or proxy server.



NetStream.Seek.Failed	0
NetStream.Play.StreamNotFound	0
NetStream.Play.Failed	0
NetStream.Failed	0
NetConnection.Connect.Rejected	0
NetConnection.Connect.Failed	0
NetConnection.Connect.AppShutdown	0

Figure 70. FlashPlayer Client – Errors view

## Troubleshooting and Diagnostics

Issue	Diagnosis, Suggestions
No SYNs sent.	Test port cannot resolve gateway or server IP to MAC. Resolve routing or switch issue. Are VLANs required?
NetConnection request fails.	Is proxy configured? Is it required? Is the application name correct? Check the Error view.
Play request is sent and it fails.	Is proxy configured? Is it required? Is the media name correct? Check the Error view.
No play requests are sent.	Is proxy configured? Is it required? Is the application name correct? Check the Error view.

Table 21. Troubleshooting checklist

## Test Case: Flash Player User Actions - PAUSE, RESUME

### Overview

Adobe's Flash Player is a pioneer in bringing rich and interactive experience to a majority of PC and MAC systems worldwide. It supports an advanced set of capabilities and technologies that enable streaming HD/H.264 and a variety of rich Internet applications (RIA). It is not an RFC based protocol; it is proprietary to Adobe and binary in nature.

Ixia is an exclusive test vendor to license Flash technology from Adobe since 2009.

### Objective

Create a test profile in IxLoad to act as a Flash Player to stream content from a Flash Media Server (FMS) 3.5 running on Windows 2003 server. This test case will use RTMP level messages to emulate user PAUSE and RESUME actions.

### Setup

The following scenario is used: Windows 2003 Server running Flash Media Server 3.5 to host a Flash compatible content, IxLoad 5.10 Flash Player Activity with RTMP and RTMPT.

One Ixia test port is connected directly to Flash Media Server (FMS). The topology shown in the following figure is representative of a more complex scenario with optional proxy that is supported with RTMPT and a CDN infrastructure that hosts the content that is sent to the FMS.

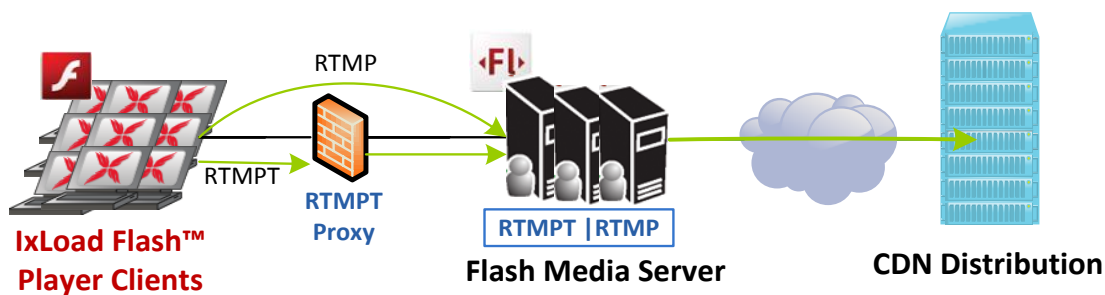
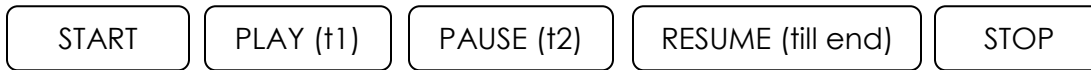


Figure 71. Flash Player setup to test Flash Media Server infrastructure

## Step-by-Step Instructions

1. Start IxLoad. In the main window, the **Scenario Editor** window appears. All test configurations will be done here.
2. To get familiar with the IxLoad GUI, see the Getting Started Guide section.
3. Add the client NetTraffic object. Configure the client network with total IP count, gateway, and VLAN, if used.
4. For a step-by-step workflow, see Annex A.
5. Configure the Flash Player client. Add the **Flash Player** activity to the client NetTraffic.
6. For the Pause and Resume user actions, configure the command list as follows:



7. Add a **PLAY** command and configure the following parameters:
  - Choose RTMP or RTMPT protocol. For RTMP, the Flash Player will connect to TCP port 1935. For RTMPT, the default port is TCP/80.
  - Enter the destination server by IP or hostname (enable DNS to use this). If the server is listening on a non-standard port, it can be specified as IP:port.
  - Enter the application name as 'all.' Application name is used by FMS for capabilities. This is unique to an FMS and must be configured correctly.
  - Specify the media as 'mp4:Extremists1.m4v.' Notice the notation. See the notes section later on how to correctly type in the media name.
  - The Play Duration should use the Range option; enter 10 sec -10 sec. All users will play for 10 seconds before moving to the next command.
  - The Stream Type must also be known. Recorded (that is, VoD) or Live are both supported and must be specified correctly. A fallback mechanism also exists with the 'Try Live then Recorded' option.
8. Add the **PAUSE** command and enter a range of time to pause. If the range is set to the same value, each user will pause for exactly that duration. With a varying start and end range value, each user will randomly pause for duration within this range.

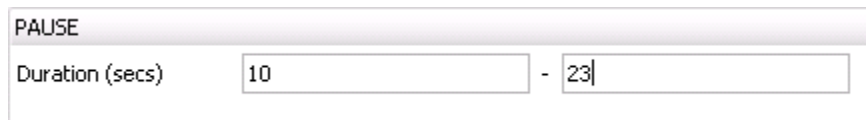
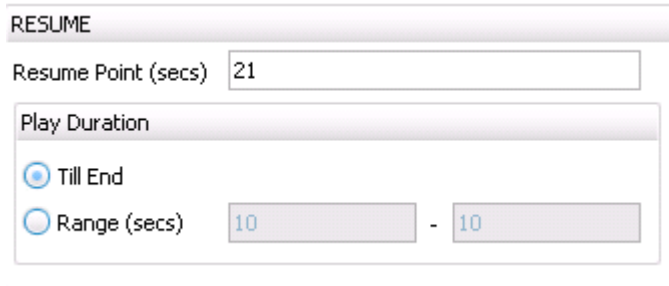


Figure 72. Configuring PAUSE command in Flash Player

9. Add the **RESUME** command and configure the following:

- Set the 'Resume Point' to a specific time.
- Set the 'Play Duration' to 'Till End.' The range option provides flexibility such as a SEEK can last for a duration, followed by a subsequent user action.



RESUME

Resume Point (secs) 21

Play Duration

Till End

Range (secs) 10 - 10

Figure 73. Configuring RESUME command in Flash Player

### Note about Media Support

- FLV: Just the name is used without the extension, for example, 'Extremists.'
- MP4: Use the format 'mp4:<full-file-name>.'
- MP3: Use the format 'mp3:<full-file-name>.'



## Test Case: Flash Player User Actions - PLAY and SEEK

### Overview

Adobe's Flash Player is a pioneer in bringing rich and interactive experience to a majority of PC and MAC systems worldwide. It supports an advanced set of capabilities and technologies that enable streaming HD/H.264 and a variety of rich Internet applications (RIA). It is not an RFC based protocol; it is proprietary to Adobe and binary in nature.

Ixia is an exclusive test vendor to license Flash technology from Adobe since 2009.

### Objective

Create a test profile in IxLoad to act as a Flash Player to stream content from a Flash Media Server (FMS) 3.5 running on Windows 2003 server. This test case will use RTMP level messages to emulate user PLAY and SEEK actions.

### Setup

The following scenario is used: Windows 2003 Server running Flash Media Server 3.5 to host a Flash compatible content, IxLoad 5.10 Flash Player Activity with RTMP and RTMPT.

One Ixia test port is connected directly to Flash Media Server (FMS). The topology shown in the following figure is representative of a more complex scenario with optional proxy that is supported with RTMPT and a CDN infrastructure that hosts the content that is sent to the FMS.

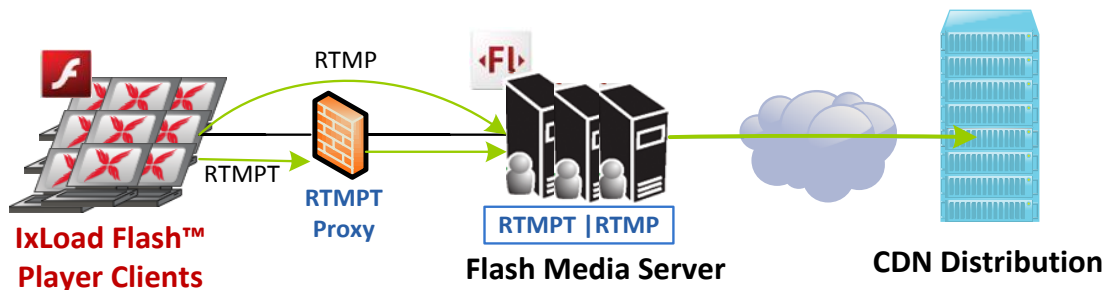


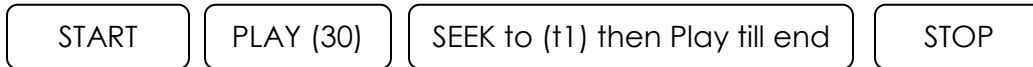
Figure 74. Flash Player setup to test Flash Media Server infrastructure



## Step-by-Step Instructions

1. Start IxLoad. In the main window, the **Scenario Editor** window appears. All test configurations will be done here.
2. To get familiar with the IxLoad GUI, see the Getting Started Guide section.
3. Add the client NetTraffic object. Configure the client network with total IP count, gateway, and VLAN, if used.
4. For a step-by-step workflow, see Annex A.
5. Configure the Flash Player client. Add the **Flash Player** activity to the client NetTraffic.

6. For the Seek user action, configure the command list as follows:



7. Add a **PLAY** command and configure the following parameters:
  - Choose RTMP or RTMPT protocol. For RTMP, the Flash Player will connect to TCP port 1935. For RTMPT, the default port is TCP/80.
  - Enter the destination server by IP or hostname (enable DNS to use this). If the server is listening on a non-standard port, it can be specified as IP:port.
  - Enter the application name as 'all.' Application name is used by FMS for capabilities. This is unique to an FMS and must be configured correctly.
  - Specify the media as 'mp4:Extremists1.m4v.' Notice the notation. See the notes section later on how to correctly type in the media name.
  - The Play Duration should use the Range option; enter 10 sec -10 sec. All users will play for 10 seconds.
  - The Stream Type must also be known. Recorded (that is, VoD) or Live are both supported and must be specified correctly. A fallback mechanism also exists with the 'Try Live then Recorded' option.

8. Add the **SEEK** command and configure as follows:

- Set the 'Seek To' to 40 – 40 seconds. The **Seek To** supports a range; each user will randomly pick a time and seek to that time location.
- Set the 'Play Duration' to 'Till End.' The range option provides flexibility such as a SEEK can last for a duration, followed by a subsequent user action.

The image shows a dialog box titled "Command Properties for 'SEEK 8'". It has two main sections. The first section is labeled "SEEK" and contains a "Seek To (secs)" field with two input boxes containing the values "20" and "40", separated by a hyphen. The second section is labeled "Play Duration" and contains two radio button options. The first option is "Till End" with an unselected radio button. The second option is "Range (secs)" with a selected radio button and two input boxes containing the values "10" and "15", separated by a hyphen.

Figure 75. Configuring SEEK command on Flash Player

#### Note about Media support

- FLV: Just the name is used without the extension, for example, 'Extremists.'
- MP4: Use the format 'mp4:<full-file-name>.'
- MP3: Use the format 'mp3:<full-file-name>.'



## Test Case: Flash Player User Actions - Playback of Multiple Media Files

### Overview

Adobe's Flash Player is a pioneer in bringing rich and interactive experience to a majority of PC and MAC systems worldwide. It supports an advanced set of capabilities and technologies that enable streaming HD/H.264 and a variety of rich Internet applications (RIA). It is not an RFC based protocol; it is 100 percent proprietary to Adobe and binary in nature.

Ixia is an exclusive test vendor to license Flash technology from Adobe since 2009.

### Objective

Create a test profile in IxLoad to act as a Flash Player to stream content from a Flash Media Server (FMS) 3.5 running on Windows 2003 server. This test uses Sequence Generator to allow each user to stream different media and cycle through the media list.

### Setup

The following scenario is used: Windows 2003 Server running Flash Media Server 3.5 to host a Flash compatible content, IxLoad 5.10 Flash Player activity with RTMP and RTMPT.

One Ixia test port is connected directly to Flash Media Server (FMS). The topology shown in the following image is representative of a more complex scenario with optional proxy that is supported with RTMPT and a CDN infrastructure that hosts the content that is sent to the FMS.

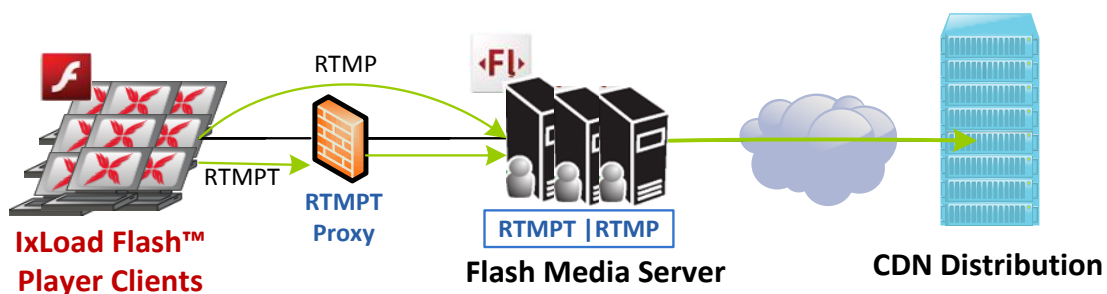
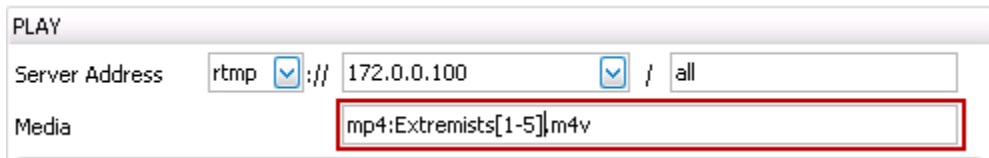


Figure 76. Flash Player setup to test Flash Media Server infrastructure

## Step-by-Step Instructions

1. Start IxLoad. In the main window, the **Scenario Editor** window appears. All test configurations will be done here.
2. To get familiar with the IxLoad GUI, see the Getting Started Guide section.
3. Add the client NetTraffic object. Configure the client network with total IP count, gateway, and VLAN, if used.
4. For a step-by-step workflow, see Annex A.
5. Configure the Flash Player client. Add the **Flash Player** activity to the client NetTraffic.
6. To PLAYBACK multiple content files, configure the PLAY command to use sequence generators in the Application Name and the Media Name fields. See the following image.



**Figure 77.** Configuring PLAY command with sequence generator to play back several files

IxLoad supports the following types of sequence generators:

- Numbers 0-9
- Letters A-Z and a-z
- System variables

Sequence generators can be combined with fixed text to create the user names, passwords, and domain names. For example, enter user[00-] to create a range of unique user names that begin with the characters 'user' (user00, user01, and so on).

### Note about Media Support

- FLV: Just the name is used without the extension, for example, 'Extremists.'
- MP4: Use the format 'mp4:<full-file-name>..'
- MP3: Use the format 'mp3:<full-file-name>.'

## Test Case: Flash™ Player Emulation - Secure streaming using RTMPE

### Overview

Adobe's Flash Player is a pioneer in bringing rich and interactive experience to a majority of PC and MAC systems worldwide. It supports an advanced set of capabilities and technologies that enable streaming HD/H.264 and a variety of rich Internet applications (RIA). It is not an RFC-based protocol, it is proprietary to Adobe and binary in nature.

Ixia is an exclusive test vendor that has been licensing Flash technology from Adobe since 2009.

RTMPE is an encrypted protocol which wraps the traditional RTMP using well-known industry-standard cryptographic primitives in a proprietary security mechanism implementation resulting in a faster and lighter-weight version as compared to RTMPS (RTMP over SSL).

One of the RTMPE drivers was the need to prevent third party stream capture tools from maliciously listening to and storing video streams intended for legitimate destinations.

Another key aspect was the complexity introduced by traditional DRM solutions in many layers of the aggregation/distribution network finally impacting user experience. In such cases, RTMPE proved to be a very good solution for premium content protection, also securing excellent user experience.

RTMPTE is the tunneled version of RTMPE over HTTP. Client RTMPE messages are sent as the payload of HTTP POST requests and, in turn, server RTMPE messages are encapsulated in HTTP 200 OK packets.

### Objective

Create a test profile in IxLoad to act as a Flash Player and stream content using RTMPE from a Flash Media Server (FMS) 4.5 running on Windows 2008 server.

The test case will highlight all the capabilities of secure stream playback using RTMPE.

## Setup

The following scenario is used:

Windows 2008 Server running Flash Media Server 4.5 to host Flash-compatible content, and IxLoad 6.10 Flash Player Activity with RTMP, RTMPT, RTMPE and RTMPTE capabilities.

One Ixia test port is connected directly to the Flash Media Server (FMS). The topology shown in Figure 76 is representative of a more complex scenario which features an optional proxy that is supported with RTMPTE and a CDN infrastructure hosting the content sent to the FMS.

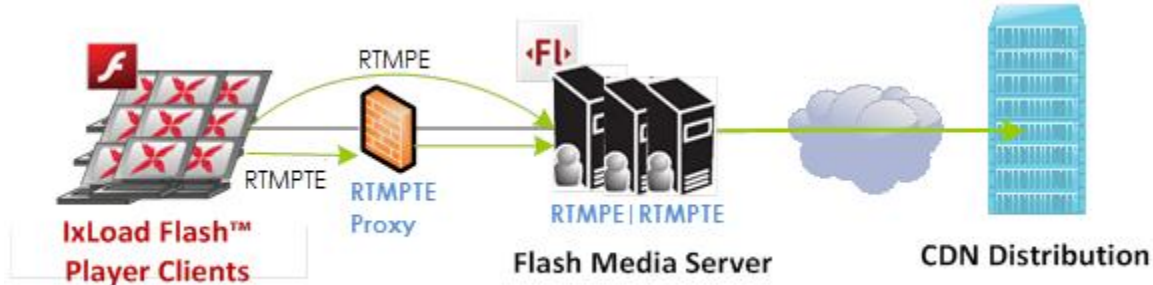


Figure 78. Flash Player setup to test Flash Media Server infrastructure

**Note:** In this particular test case, no RTMPTE Proxy is used.

## Test Variables

### Test Tool Variables

Parameters	Description
Flash Player Clients	According to the <b>Simulated Users</b> objective value (For details, please refer to the Step-by-step Instructions.)
Flash Player parameters	Keep the default settings. (Please refer to the Step-by-step Instructions for the parameters to modify for the specific RTMPE test.)
TCP parameters	TCP RX and TX buffer at 32768 bytes.
Flash Player client command list	Please refer to the Step-by-step Instructions for details on how to configure basic playback command.
RTMPT/RTMPTE Proxy support	If clients connect through a content switch or a proxy, IxLoad can send all RTMPT/RTMPTE packets to this configured <b>IP:port</b> .

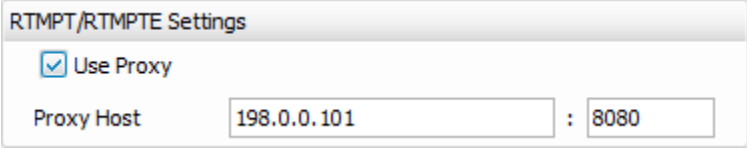
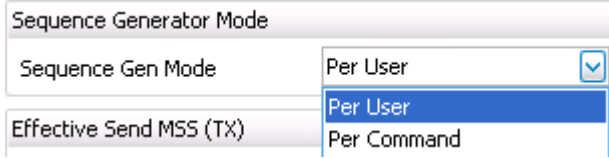
Parameters	Description
	 <p style="text-align: center;">Figure 79. Configuring RTMPT/RTMPTE proxy</p>
Use of sequence generator	<p>Sequence generator can be used to expand requested media files. For example, <b>video[1-5].m4v</b> file is actually expanded to video1.m4v through video5.m4v.</p> <p>The way in which the selection is made is based on the Mode.</p>
Sequence Generator Mode	<p>If <b>Per-User</b> is selected, all users start from the same index (for example, sample1.mp4 used by all users at start) and users move through the index at the end of each stream duration.</p> <p>If <b>Per-Command</b> is selected, all users stagger to start off uniquely (for example, user1 picks sample1.mp4, user2 picks sample2.mp4) and each user moves through this staggered (unique) list based on stream length or configured time.</p>  <p style="text-align: center;">Figure 80. Choosing the sequence generator mode</p>

Table 22. Test Tool Variables

### DUT Test Variables

Device(s)	Variation	Description
Flash Media Server	Enable RTMPE and RTMPTE	Make sure that RTMPE and RTMPTE is enabled and note listening ports.

Table 23. DUT Variables



## Step-by-step Instructions

1. Start IxLoad. In the main window, the **Scenario Editor** window opens. All test configuration is performed here.
2. To get familiar with the IxLoad GUI, please refer to the Getting Started Guide section.
3. Add the client NetTraffic object. Configure the client network with total IP count, gateway, and VLAN, if used.
4. For a step-by-step workflow, see Annex A.
5. Configure the Flash Player client. Add the **Flash Player** activity to the client NetTraffic.
6. **For basic PLAYBACK**, configure the command list as follows:



7. Add a **PLAY** command and configure the following parameters:
  - Enter the destination server by IP or by hostname (enable DNS to use this). If the server is listening on a non-standard port, it can be specified as IP:port.
  - Enter the Application Name as 'all.' Application Name is used by the FMS for capabilities. This is unique to an FMS and must be configured correctly.
  - Specify the Media as 'mp4:Extremists1.m4v.' Note the format in which the media name is typed. Please refer to the NOTES section later on for details on how to correctly type the media name.
  - The Play Duration can be either playing until the end when 'Till End' is selected, or the Range option, which can be used to specify a time values range. Each user randomly selects a duration within this range.
  - The Stream Type must also be known. Both the Recorded (that is, VoD) and Live options are supported and must be specified correctly. A fallback mechanism is also in place with the 'Try Live then Recorded' option.

The screenshot shows a dialog box titled "Command Properties for 'PLAY'". It contains the following fields and options:

- Server Address:** A text field containing "172.0.0.100" and a dropdown menu with a checkmark icon, followed by a slash and a text field containing "all".
- Media:** A text field containing "mp4:Extremists.m4v".
- Play Duration:** A section with two radio buttons: "Till End" (selected) and "Range (secs)". The "Range (secs)" option has two adjacent text input fields, both containing the number "10", separated by a minus sign.
- Stream Type:** A section with three radio buttons: "Recorded" (selected), "Live", and "Try live then recorded".

Figure 81. Configuration for PLAY command for RTMPE Flash Player playback

## Test Case: Flash™ Player Emulation - Secure streaming using RTMPE

8. Add a **STOP** command to ensure graceful teardown of sessions at the end of the test or when the test is stopped manually.

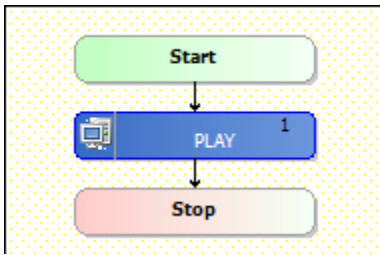


Figure 82. Flash Player command list example

9. From the Flash Player **Global Settings** tab, change the **Protocol** field to *RTMPE*.

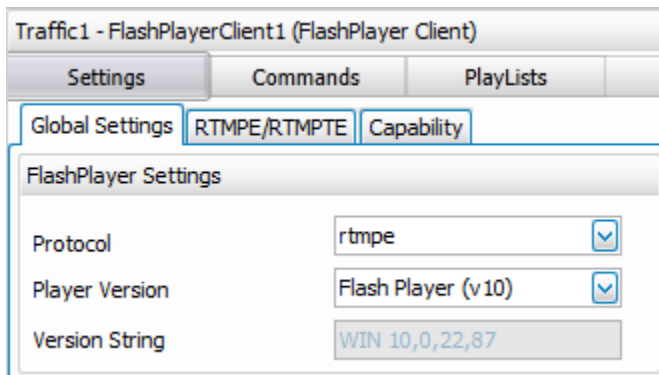


Figure 83. RTMPE Configuration as Flash Player streaming protocol

10. Keep the default values for the parameters in the Flash Player RTMPE/RTMPTE tab:

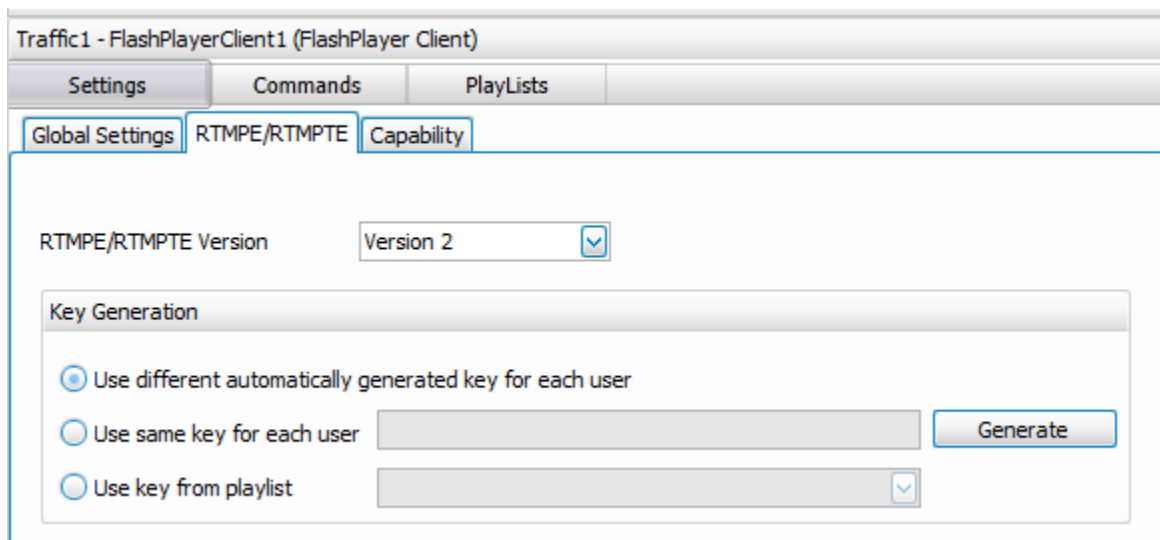


Figure 84. RTMPE/RTMPTE parameters configuration

**NOTES on RTMPE/RTMPTE options:**

- RTMPE/RTMPTE Version: Supported versions are 1 and 2. The difference between the versions resides in how the handshake messages are constructed.
- Key Generation: Specifies the method to use for the private cryptographic key generation:
  - Use a different automatically generated key for each user: a unique key will be generated by IxLoad for each user.
  - Use the same key for each user: all simulated users will be using the same key, which can be either automatically generated (by clicking the **Generate** button) or manually entered (in hex format). The key should be 1024 bits (128 hex characters); if a shorter key is entered, IxLoad automatically pads the remainder of the key with zeros.
  - Use a key from a playlist: a defined playlist can be used from which to import keys. Playlists can be added from the **PlayLists** tab.

11. After you set up the Flash Player activity, configure simulated user as the test objective and set the objective value to the desired number of users.
12. Add test port resources and run the test. Please refer to the Results Analysis and the Troubleshooting and Diagnostics sections for further details.

**NOTES on Media Name and Format to Use**

- FLV: Only the name is used, without the extension, for example, 'Extremists'
- MP4: Use the 'mp4:<full-file-name>' format.
- F4V: Use the 'mp4:<full-file-name>' format.
- MP3: Use the 'mp3:<full-file-name>' format.

**Results Analysis**

Metric	Key Performance Indicators	Statistics View
Performance Metrics	User Count, Active Stream, Played, Paused RX/TX Throughput, Audio/Video/Data Throughput	FlashPlayer Client – Objectives FlashPlayer Client – Throughput Objectives
Application Level Transactions	Handshake and NetConnection, Requested, Successful, Failed	FlashPlayer Client – RTMPE/RTMPTE Handshake
Application Level	Play/Pause/Resume/Seek/Stop Requests	FlashPlayer Client –

Test Case: Flash™ Player Emulation - Secure streaming using RTMPE

Commands	Send/Successful/Failed/Timeout	Command
Application Level Failure Monitoring	Packet Counts for Audio, Video, Data Key Error Codes	FlashPlayer Client - AVD Packets FlashPlayer Client – Errors  FlashPlayer – RTMPE/RTMPTE Errors
Streaming Quality	Play Latency, Netconnect Latency	FlashPlayer Client - Latency
TCP Statistics	TCP SYN, SYN-ACK, Connections Established, Failed	FlashPlayer Client - TCP Connections  FlashPlayer Client - TCP Failures

Table 24. Results Analysis for Flash Player content playback

## Test Case: Flash™ Player Emulation - Secure streaming using RTMPE

The encrypted nature of RTMPE can be easily revealed by trying to analyze a capture which was taken from a point in between client and server.

123	0.000016	SuperMic_6a:fc:3100:ac:10:00:65:0	ARP	42	172.16.0.1	is at 00:25:90:6a:fc:31
124	0.000072	172.16.0.101	172.16.0.1	TCP	70	32790 > macromedia-fcs [SYN] Seq=0 W
125	0.000120	172.16.0.1	172.16.0.101	TCP	70	macromedia-fcs > 32790 [SYN, ACK] Se
126	0.000073	172.16.0.101	172.16.0.1	TCP	66	32790 > macromedia-fcs [ACK] Seq=1 A
127	0.000741	172.16.0.101	172.16.0.1	RTMP	1514	Unknown (0x0)
128	0.000003	172.16.0.101	172.16.0.1	TCP	155	32790 > macromedia-fcs [PSH, ACK] Se
129	0.000023	172.16.0.1	172.16.0.101	TCP	66	macromedia-fcs > 32790 [ACK] Seq=1 A

Frame 127: 1514 bytes on wire (12112 bits), 1514 bytes captured (12112 bits)

- Ethernet II, Src: 00:ac:10:00:65:00 (00:ac:10:00:65:00), Dst: SuperMic\_6a:fc:31 (00:25:90:6a:fc:31)
- Internet Protocol Version 4, Src: 172.16.0.101 (172.16.0.101), Dst: 172.16.0.1 (172.16.0.1)
- Transmission Control Protocol, Src Port: 32790 (32790), Dst Port: macromedia-fcs (1935), Seq: 1514, Win: 0, Len: 0
- Real Time Messaging Protocol (Unknown (0x0))
  - RTMP Header
    - 10.. .... = Format: 2
    - ..00 0011 = Chunk Stream ID: 3
    - Timestamp delta: 8684934
    - Timestamp: 8684934 (calculated)
  - RTMP Body

Figure 85. RTMPE message exchange

As shown in Figure 83, the packet capture tool cannot properly decode the selected RTMPE packet type and headers. Since this is the first exchanged RTMPE message, it is obvious that it is part of the handshake phase; however, an unauthorized sniffing attack will not be able to interpret such RTMPE messages as in the case of the clear text RTMP packets.

## Real-time Statistics

Flash Player activity in IxLoad provides TCP-level statistics, all Handshake and Netconnection statistics, and RTMP level commands. It also provides critical error codes sent by the server (for example, to identify if an application name is incorrect, or if a wrong filename is requested), and user experience metrics such as latency.

# Test Case: Flash™ Player Emulation - Secure streaming using RTMPE

The **FlashPlayer Client - Objective** view is a real-time view showing active users and streams. The active number of sessions being reported on the DUT should correlate to this view.

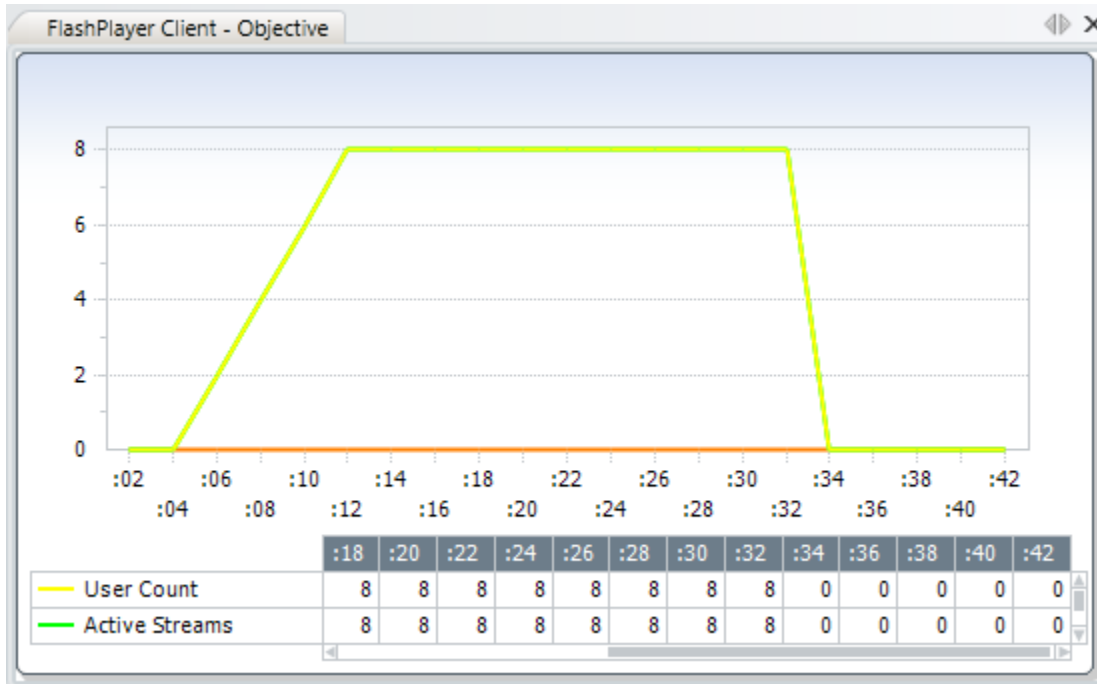


Figure 86. FlashPlayer Client – Objective view

The **FlashPlayer Client – Throughput Objectives** view provides visibility into the audio, video, RX, and TX throughput. Use this view to verify that the number of active streams and the total receive bandwidth correlate, if the average bitrate for the streams is known.

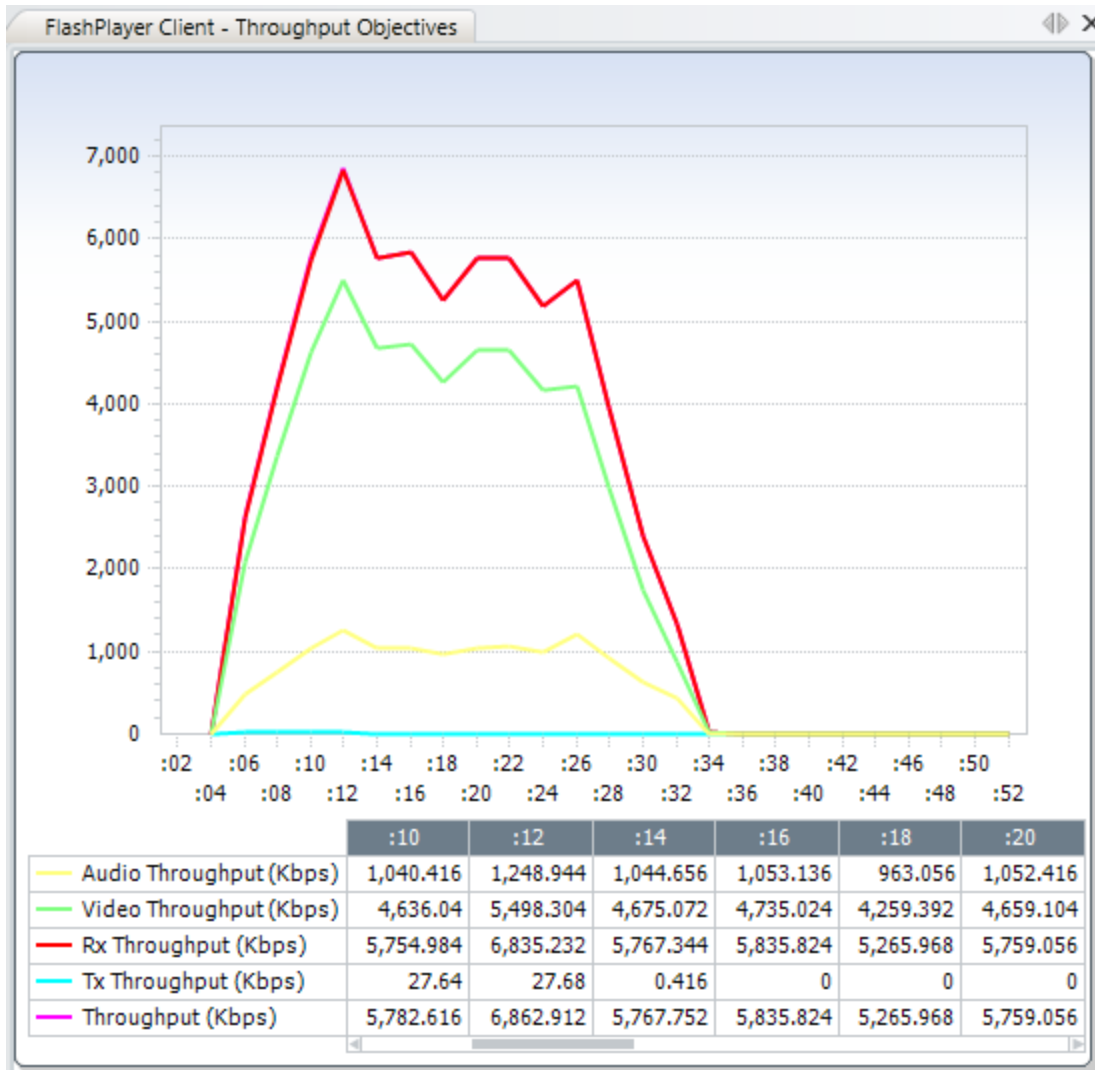


Figure 87. FlashPlayer Client – Throughput Objectives view

Complete TCP statistics are available in the **FlashPlayer - TCP Connections** view. If there is a connection or reachability issue, this view shows failures.

		:52
<span style="color: red;">■</span>	TCP SYN Sent	8
<span style="color: magenta;">■</span>	TCP SYN_SYN-ACK Received	8
<span style="color: pink;">■</span>	TCP Connections Established	8
<span style="color: purple;">■</span>	TCP FIN Sent	8
<span style="color: blue;">■</span>	TCP FIN-ACK Received	8
<span style="color: green;">■</span>	TCP FIN-ACK Sent	5
<span style="color: cyan;">■</span>	TCP FIN Received	5
<span style="color: yellow;">■</span>	TCP Retries	0
<span style="color: grey;">■</span>	TCP Timeouts	0

Figure 88. FlashPlayer Client – TCP Connections view

The **FlashPlayer Client – RTMPE/RTMPTE Handshake** view shows the application level handshake, followed by a NetConnection message. If there is a Flash server interoperability issue, this view indicates if it is a handshake or a NetConnection failure.

		:52
<span style="color: orange;">■</span>	Handshake Sent	8
<span style="color: darkgreen;">■</span>	Handshake Successful	8
<span style="color: blue;">■</span>	Handshake Failed	0
<span style="color: lightgreen;">■</span>	RTMPE/RTMPTE Handshake Timeout	0
<span style="color: olive;">■</span>	NetConnection Sent	8
<span style="color: gold;">■</span>	NetConnection Successful	8
<span style="color: cyan;">■</span>	NetConnection Failed	0
<span style="color: grey;">■</span>	RTMPE/RTMPTE NetConnection Timeout	0

Figure 89. FlashPlayer Client – RTMPE/RTMPTE Handshake view

The **FlashPlayer Client - Command** view shows individual application-level events such as Play, Pause, Seek, and Resume actions.

		:52
<span style="color: yellow;">■</span>	Play Request Sent	8
<span style="color: lightgreen;">■</span>	Play Request Successful	8
<span style="color: red;">■</span>	Play Request Failed	0
<span style="color: blue;">■</span>	Play Request Timeout	0
<span style="color: pink;">■</span>	Pause Request Sent	0
<span style="color: cyan;">■</span>	Pause Request Successful	0
<span style="color: brown;">■</span>	Pause Request Failed	0
<span style="color: grey;">■</span>	Pause Request Timeout	0

Figure 90. FlashPlayer Client – Command view

The **FlashPlayer Client - Audio Video Data Packets** view processes header information present in the responses to count the total number of audio, video, or data packets.



## Test Case: Flash™ Player Emulation - Secure streaming using RTMPE

:52	
Audio Packets Received	7,656
Video Packets Received	4,792
Data Packets Received	28

Figure 91. FlashPlayer Client – Audio Video Data Packets view

The **FlashPlayer Client – Errors** view provides error codes to identify specific issues in streaming from a Flash Media or a proxy server.

:52	
NetStream.Seek.Failed	0
NetStream.Play.StreamNotFound	0
NetStream.Play.Failed	0
NetStream.Failed	0
NetConnection.Connect.Rejected	0
NetConnection.Connect.Failed	0
NetConnection.Connect.AppShutdown	0

Figure 92. FlashPlayer Client – Errors view

The **FlashPlayer Client – Latency** view can be used to evaluate user experience. It contains statistics such as the time required to receive the very first byte of a stream since the client issued the play request command (Play Latency) or the time elapsed to receive a response to a NetConnect command (NetConnect Latency). High but constant latency values can indicate an overloaded media server, while high variation of the latency values can indicate congestion in the delivery network.

	:06	:08	:10	:12
Play Latency (ms)	1	1	1	1
NetConnect Latency (ms)	2	1	1	2

Figure 93. FlashPlayer Client – Latency

## Troubleshooting and Diagnostics

Issue	Diagnosis, Suggestions
No SYNs sent.	Test port cannot resolve gateway or server IP to MAC. Resolve routing or switch issue. Are VLANs required?
NetConnection request fails.	Is proxy configured? Is it required? Is the application name correct? Check the Error view.
Play request Is sent and it fails.	Is proxy configured? Is it required?

Issue	Diagnosis, Suggestions
	Is the media name correct? Check the Error view.
No play requests are sent.	Is proxy configured? Is it required?  Is the application name correct? Check the Error view.

Table 25. Troubleshooting checklist



## Test Case: Flash™ Player Emulation - Secure streaming using RTMPTE

### Overview

Adobe's Flash Player is a pioneer in bringing rich and interactive experience to a majority of PC and MAC systems worldwide. It supports an advanced set of capabilities and technologies that enable streaming HD/H.264 and a variety of rich Internet applications (RIA). It is not an RFC-based protocol, it is proprietary to Adobe and binary in nature.

Ixia is an exclusive test vendor that has been licensing Flash technology from Adobe since 2009.

RTMPE is an encrypted protocol which wraps the traditional RTMP using well-known industry-standard cryptographic primitives in a proprietary security mechanism implementation resulting in a faster and lighter-weight version as compared to RTMPS (RTMP over SSL).

One of the RTMPE drivers was the need to prevent third party stream capture tools from maliciously listening to and storing video streams intended for legitimate destinations. Another key aspect was the complexity introduced by traditional DRM solutions in many layers of the aggregation/distribution network finally impacting user experience. In such cases, RTMPE proved to be a very good solution for premium content protection, also securing excellent user experience.

RTMPTE is the tunneled version of RTMPE over HTTP. Client RTMPE messages are sent as the payload of HTTP POST requests and, in turn, server RTMPE messages are encapsulated in HTTP 200 OK packets.

### Objective

Create a test profile in IxLoad to act as a Flash Player and stream content using RTMPTE from a Flash Media Server (FMS) 4.5 running on Windows 2008 Server.

The test case will highlight all the capabilities of secure stream playback using RTMPTE.

### Setup

The following scenario is used:

Windows 2008 Server running Flash Media Server 4.5 to host Flash compatible content, IxLoad 6.10 Flash Player Activity with RTMP, RTMPT, RTMPE and RTMPTE capabilities.

One Ixia test port is connected directly to the Flash Media Server (FMS). The topology shown in Figure 92 is representative of a more complex scenario featuring an optional proxy that is supported with RTMPTE and a CDN infrastructure hosting the content sent to the FMS.

Test Case: Flash™ Player Emulation - Secure streaming using RTMPTE

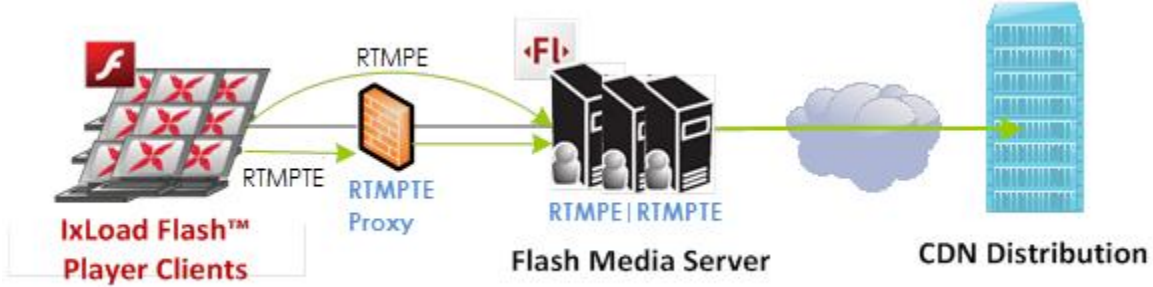


Figure 94. Flash Player setup to test Flash Media Server infrastructure

**Note:** In this particular test case, no RTMPTE Proxy is used.

## Test Variables

### Test Tool Variables

Parameters	Description
Flash Player Clients	According to the <b>Simulated Users</b> objective value (For details, please refer to the Step-by-step Instructions).
Flash Player parameters	Keep default settings. Please refer to the Step-by-step Instructions for the parameters to modify for the RTMPTE test.
TCP parameters	TCP RX and TX buffer at 32768 bytes.
Flash Player client command list	Please refer to the Step-by-step Instruction for details on how to configure basic playback command.
RTMPT/RTMPTE Proxy support	If clients connect through a content switch or a proxy, IxLoad can send all RTMPT/RTMPTE packets to this configured <b>IP:port</b> .  <div data-bbox="527 1438 1269 1585" data-label="Image"> <p>The screenshot shows a configuration window titled 'RTMPT/RTMPTE Settings'. It has a checked checkbox for 'Use Proxy'. Below it, the 'Proxy Host' field contains '198.0.0.101' and the port field contains '8080'.</p> </div>
Use of sequence generator	Sequence generator can be used to expand requested media files. For example, <b>video[1-5].m4v</b> file is actually expanded to video1.m4v through video5.m4v The way in which the selection is made is based on the Mode.
Sequence Generator	If <b>Per-User</b> is selected, all users start from the same index (for

Figure 95. Configuring RTMPT/RTMPTE proxy

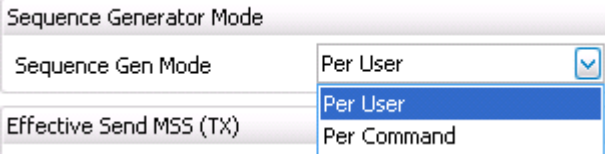
Parameters	Description
Mode	<p>example, sample1.mp4 used by all users at start) and users move through the index at the end of each stream duration.</p> <p>If <b>Per-Command</b> is selected, all users stagger to start off uniquely (for example, user1 picks sample1.mp4, user2 picks sample2.mp4) and each user moves through this staggered (unique) list based on stream length or configured time.</p>  <p style="text-align: center;">Figure 96. Choosing the sequence generator mode</p>

Table 26. Test Tool Variables

### DUT Test Variables

Device(s)	Variation	Description
Flash Media Server with HTTP Server	Enable RTMPE and RTMPTE	Make sure that RTMPE and RTMPTE is enabled and note listening ports.

Table 27. DUT Variables

### Step-by-Step Instructions

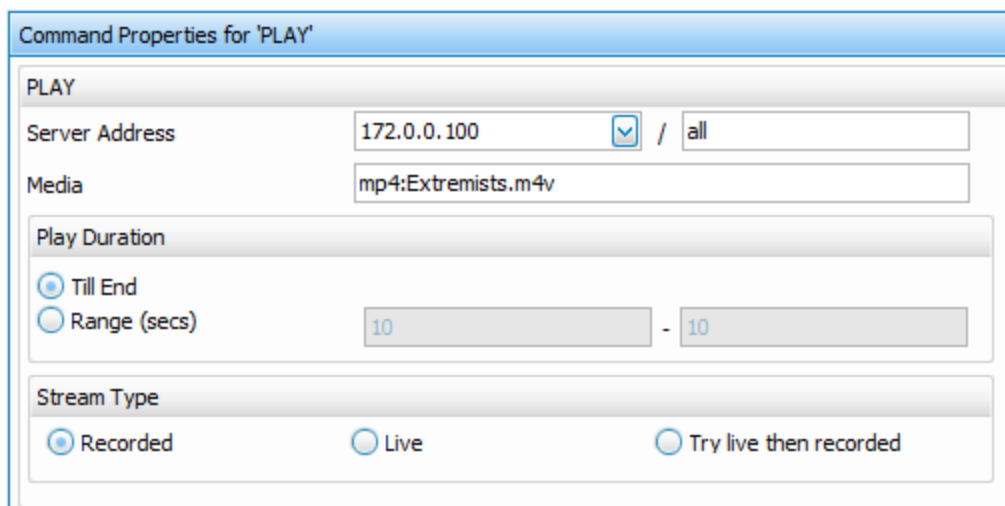
1. Start IxLoad. In the main window, the **Scenario Editor** window opens. All test configuration is performed here.
2. To get familiar with the IxLoad GUI, please refer to the Getting Started Guide section.
3. Add the client NetTraffic object. Configure the client network with total IP count, gateway, and VLAN, if used.
4. For a step-by-step workflow, see Annex A.
5. Configure the Flash Player client. Add the **Flash Player** activity to the client NetTraffic.
6. **For basic PLAYBACK**, configure the command list as follows:



7. Add a **PLAY** command and configure the following parameters:
  - Enter the destination server by IP or hostname (enable DNS to use this). If the server is listening on a non-standard port, it can be specified as IP:port.

## Test Case: Flash™ Player Emulation - Secure streaming using RTMPTE

- Enter the Application Name as 'all.' Application Name is used by the FMS for capabilities. This is unique to an FMS and must be configured correctly.
- Specify the Media as 'mp4:Extremists1.m4v.' Note the format in which the media name is typed. Please refer to the NOTES section later on for details on how to correctly type the media name.
- The Play Duration can be either playing until the end when 'Till End' is selected, or the Range option, which can be used to specify a time values range. Each user randomly selects a value within this range
- The Stream Type must also be known. Both the Recorded (that is, VoD) and Live options are supported and must be specified correctly. A fallback mechanism is also in place with the 'Try Live then Recorded' option.



Command Properties for 'PLAY'

PLAY

Server Address: 172.0.0.100 / all

Media: mp4:Extremists.m4v

Play Duration

Till End

Range (secs): 10 - 10

Stream Type

Recorded  Live  Try live then recorded

Figure 97. PLAY command configuration for RTMPTE Flash Player playback

## Test Case: Flash™ Player Emulation - Secure streaming using RTMPTE

8. Add a **STOP** command to ensure graceful teardown of sessions at the end of the test or when the test is stopped manually.

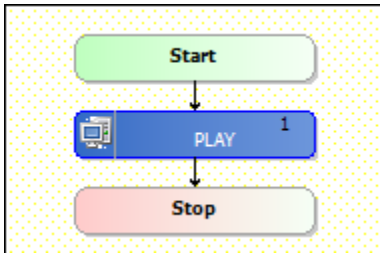


Figure 98. Flash Player command list example

9. From the Flash Player **Global Settings** tab, change the **Protocol** field to *RTMPTE*.

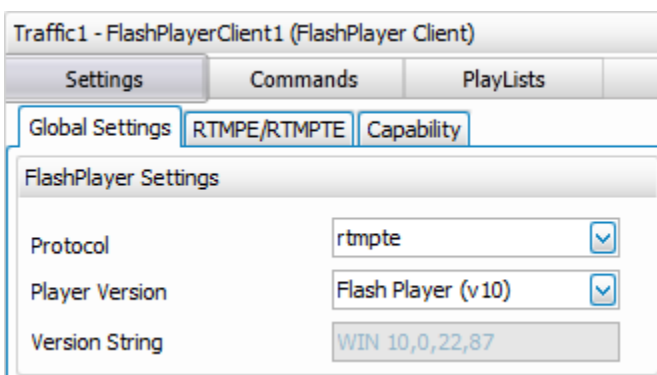


Figure 99. Configuration of RTMPTE as Flash Player streaming protocol

10. Keep the default values for the parameters in the Flash Player RTMPE/RTMPTE tab:

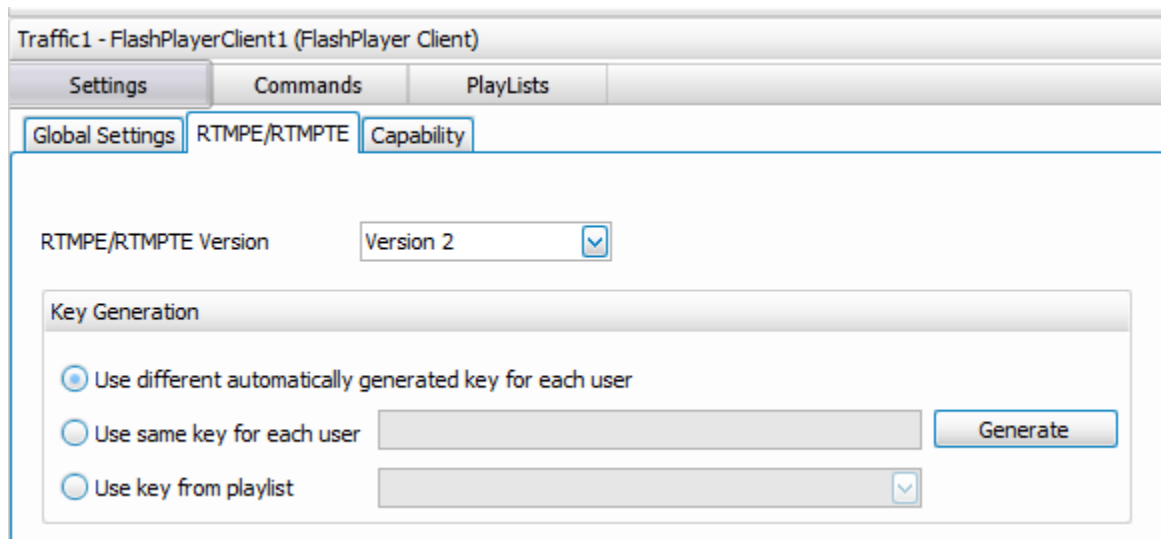


Figure 100. RTMPE/RTMPTE parameters configuration

### NOTES on the RTMPE/RTMPTE options:



## Test Case: Flash™ Player Emulation - Secure streaming using RTMPTE

- RTMPE/RTMPTE Version: Supported versions are 1 and 2. The difference between the versions resides in how the handshake messages are constructed.
  - Key Generation: Specifies the method to use for the private cryptographic key generation:
    - Use a different, automatically generated key for each user: a unique key is generated by IxLoad for each user.
    - Use the same key for each user: all simulated users use the same key, which can be either automatically generated (by clicking the **Generate** button) or manually entered (in hex format). The key should be 1024 bits (128 hex characters). If a shorter key is entered, IxLoad automatically pads the remainder of the key with zeros.
    - Use a key from a playlist: a defined playlist can be used from which to import keys. Playlists can be added from the **PlayLists** tab.
11. After you set up the Flash Player activity, configure simulated user as the test objective and set the objective value to the desired number of users.
12. Add test port resources and run the test. Please refer to the Results Analysis and the Troubleshooting and Diagnostics sections for further information.

### NOTES on Media Name and Format to Use

- FLV: Only the name is used, without the extension, for example, 'Extremists'
- MP4: Use the 'mp4:<full-file-name>' format.
- F4V: Use the 'mp4:<full-file-name>' format.
- MP3: Use the 'mp3:<full-file-name>' format.

## Results Analysis

Metric	Key Performance Indicators	Statistics View
Performance Metrics	User Count, Active Stream, Played, Paused  RX/TX Throughput, Audio/Video/Data Throughput	FlashPlayer Client – Objectives FlashPlayer Client – Throughput Objectives
Application Level Transactions	Handshake and NetConnection, Requested, Successful, Failed	FlashPlayer Client – RTMPE/RTMPTE Handshake
Application Level Commands	Play/Pause/Resume/Seek/Stop Requests Send/Successful/Failed/Timeout	FlashPlayer Client – Command
Application Level	Successful, Failed	FlashPlayer Client - AVD

Test Case: Flash™ Player Emulation - Secure streaming using RTMPTE

Metric	Key Performance Indicators	Statistics View
Failure Monitoring	Packet Counts for Audio, Video, Data Key Error Codes	Packets FlashPlayer Client – Errors  FlashPlayer Client – RTMPE/RTMPTE Errors
Streaming Quality	Play Latency, Netconnect Latency	FlashPlayer Client - Latency
TCP Statistics	TCP SYN, SYN-ACK, Connections Established, Failed	FlashPlayer Client - TCP Connections  FlashPlayer Client – TCP Failures
HTTP Statistics	Open/Send/Idle URL Request Sent/Successful/Failed	FlashPlayer Client – RTMPT/RTMPTE

Table 28. Results Analysis for Flash Player content playback

As opposed to RTMPE (which is running directly over TCP), RTMPTE implies transport over HTTP messages. As such, RTMPTE protocol functionality is achieved by using two HTTP packet types: POST (issued by the client) and 200OK (generated by the server). The payload of these packets consists of encrypted RTMPE messages.

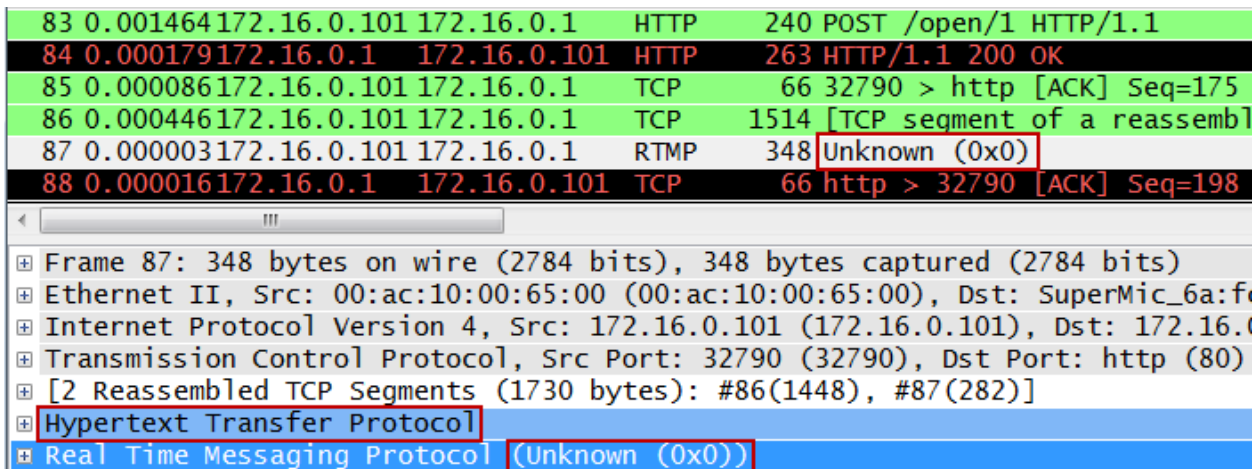


Figure 101. RTMPTE message exchange

Fig. 99 shows a packet capture screenshot of an RTMPTE message exchange taken from a point in between client and server. Although all HTTP packet types and headers are readable, their payload – that is, the RTMPE messages- is not properly decoded by the packet capture tool due to its encrypted nature

## Real-time Statistics

Flash Player activity in IxLoad provides TCP-level statistics, all Handshake and Netconnection statistics, and RTMP-level commands. It also provides critical error codes sent by the server (for example, to identify if an application name is incorrect, or if a wrong filename is requested), and user experience metrics such as latency.

The **FlashPlayer Client - Objective** view is a real-time view showing active users and streams. The active number of sessions being reported on the DUT should correlate to this view.

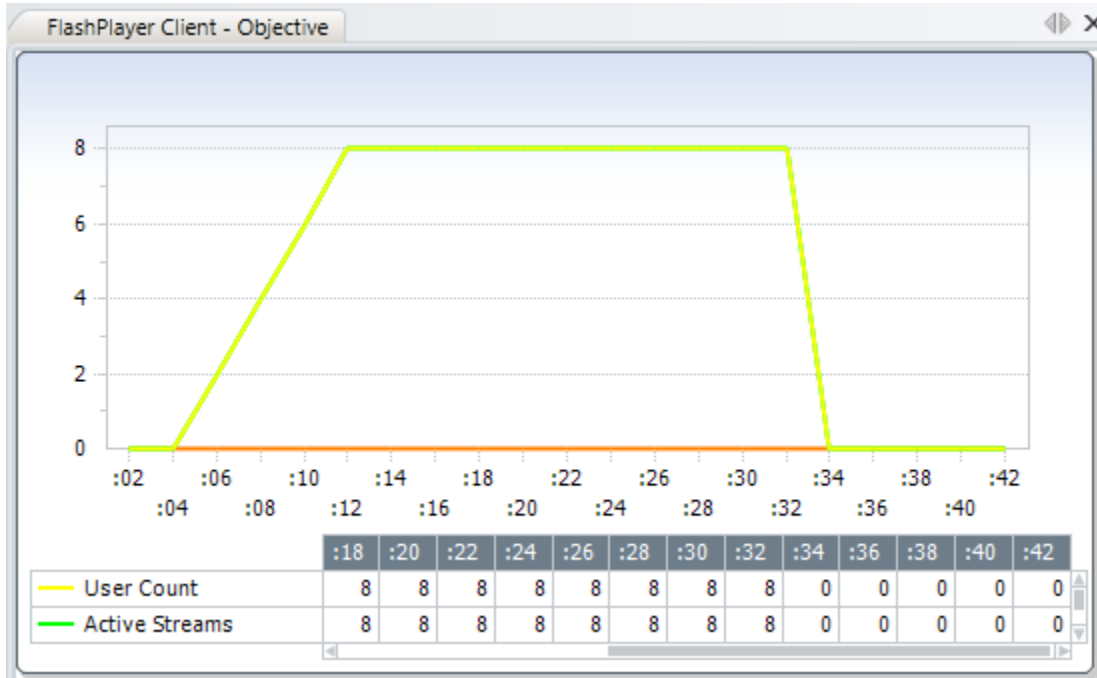


Figure 102. FlashPlayer Client – Objective view

The **FlashPlayer Client – Throughput Objectives** view provides visibility into the audio, video, RX, and TX throughput. Use this view to verify that the number of active streams and the total receive bandwidth correlate, if the average bitrate for the streams is known.

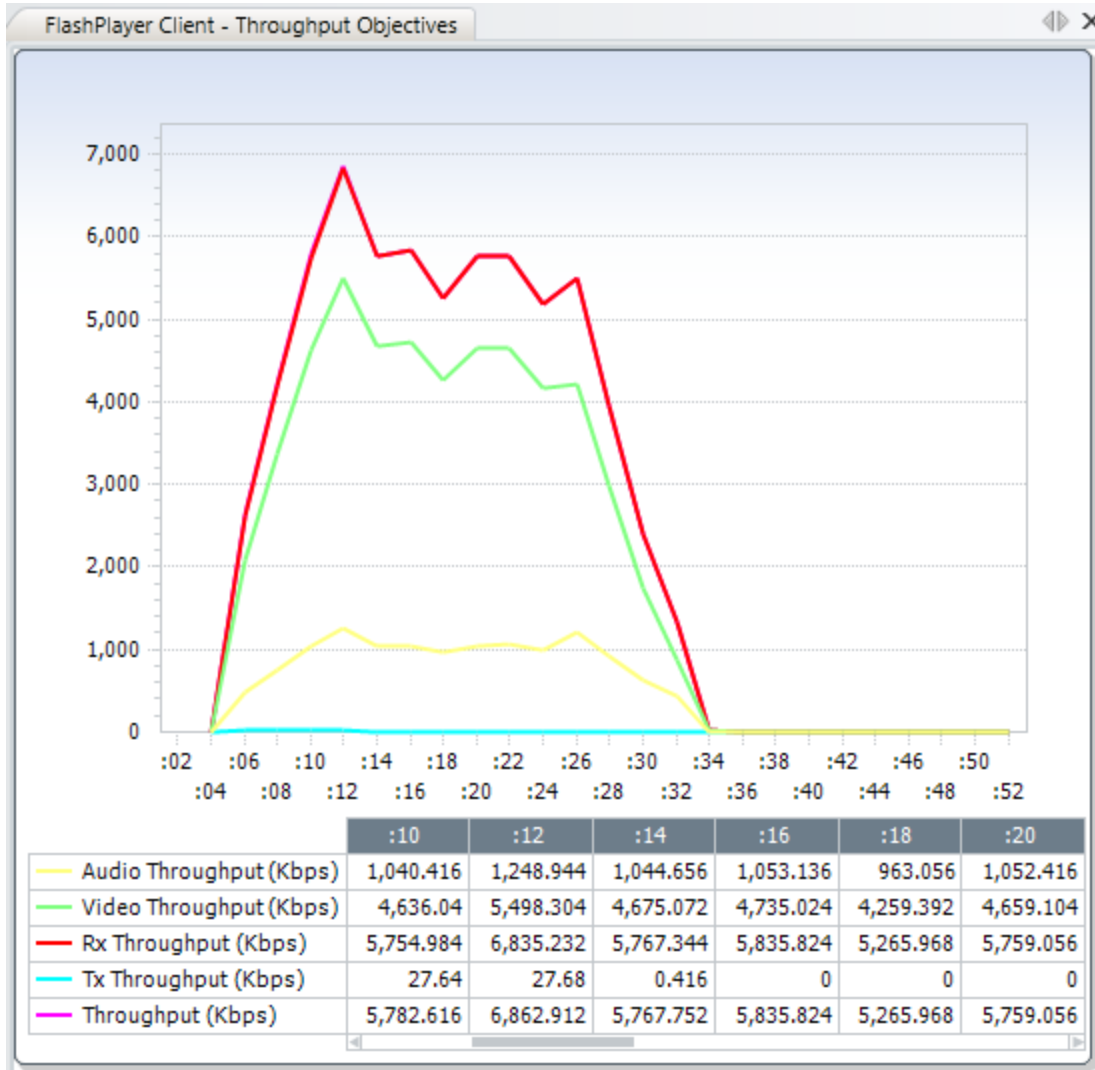


Figure 103. FlashPlayer Client – Throughput Objectives view

Complete TCP statistics are available in the **FlashPlayer - TCP Connections** view. If there is a connection or reachability issue, this view shows failures.

		:52
TCP SYN Sent		8
TCP SYN_SYN-ACK Received		8
TCP Connections Established		8
TCP FIN Sent		8
TCP FIN-ACK Received		8
TCP FIN-ACK Sent		5
TCP FIN Received		5
TCP Retries		0
TCP Timeouts		0

Figure 104. FlashPlayer Client – TCP Connections view

The **FlashPlayer Client – RTMPE/RTMPTE Handshake** view shows the application-level handshake, followed by a NetConnection message. If there is a Flash server interoperation issue, this view indicates whether it is a handshake or a NetConnection failure.

		:52
Handshake Sent		8
Handshake Successful		8
Handshake Failed		0
RTMPE/RTMPTE Handshake Timeout		0
NetConnection Sent		8
NetConnection Successful		8
NetConnection Failed		0
RTMPE/RTMPTE NetConnection Timeout		0

Figure 105. FlashPlayer Client – Handshake view

The **FlashPlayer Client - Command** view shows individual application-level events such as Play, Pause, Seek, and Resume actions.

		:52
Play Request Sent		8
Play Request Successful		8
Play Request Failed		0
Play Request Timeout		0
Pause Request Sent		0
Pause Request Successful		0
Pause Request Failed		0
Pause Request Timeout		0

Figure 106. FlashPlayer Client – Command view

The **FlashPlayer Client - Audio Video Data Packets** view processes header information present in the responses to count the total number of audio, video, or data packets.

Test Case: Flash™ Player Emulation - Secure streaming using RTMPTE

		:52
Audio Packets Received		7,656
Video Packets Received		4,792
Data Packets Received		28

Figure 107. FlashPlayer Client – Audio Video Data Packets view

The **FlashPlayer Client – Errors** view provides error codes to identify specific issues in streaming from a Flash Media or a proxy server.

		:52
NetStream.Seek.Failed		0
NetStream.Play.StreamNotFound		0
NetStream.Play.Failed		0
NetStream.Failed		0
NetConnection.Connect.Rejected		0
NetConnection.Connect.Failed		0
NetConnection.Connect.AppShutdown		0

Figure 108. FlashPlayer Client – Errors view

The **FlashPlayer Client – Latency** view can be used to evaluate user experience. It contains statistics such as the time required to receive the very first byte of a stream since the client issued the play request command (Play Latency) or the time elapsed to receive a response to a NetConnect command (NetConnect Latency). High but constant latency values can indicate an overloaded media server, while high variation of the latency values can indicate congestion in the delivery network.

	:06	:08	:10	:12
Play Latency (ms)	1	1	1	1
NetConnect Latency (ms)	2	1	1	2

Figure 109. FlashPlayer Client – Latency

The **FlashPlayer Client – RTMPT/RTMPTE** view provides HTTP-level statistics.

		:52
Open Url Request Sent		8
Open Url Request Successful		8
Open Url Request Failed		0
Send Url Request Sent		40
Send Url Request Successful		40
Send Url Request Failed		0
Idle Url Request Sent		632
Idle Url Request Successful		626
Idle Url Request Failed		0

Figure 110. FlashPlayer Client – RTMPT/RTMPTE

## Troubleshooting and Diagnostics

Issue	Diagnosis, Suggestions
No SYNs sent.	Test port cannot resolve gateway or server IP to MAC. Resolve routing or switch issue. Are VLANs required?
NetConnection request fails.	Is proxy configured? Is it required? Is the application name correct? Check the Error view.
Play request is sent and it fails.	Is proxy configured? Is it required? Is the media name correct? Check the Error view.
No play requests are sent.	Is proxy configured? Is it required? Is the application name correct? Check the Error view.

Table 29. Troubleshooting checklist

## Test Case: Basic Adobe® HTTP Dynamic Streaming Client

### Overview

Adobe®'s HTTP Dynamic Streaming (HDS) is a high-quality video delivery mechanism which provides an alternative to the RTMP streaming based methods, especially for devices which do not natively use Flash Player. HDS does not rely on expensive and specialized streaming servers or closed and inaccessible proprietary protocols, enabling on-demand and live adaptive bitrate video delivery over regular HTTP connections.

HDS is streaming video content in small chunks called fragments (F4F file format), which are based on standard open-source MP4 fragments (fMP4).

Along with the video fragments file itself, manifest files (F4M files in XML-based format) play an important part in HDS streaming. They contain information about available bitrates, fragment format, metadata information, and other properties available for a video presentation.

Some of the HDS technology benefits are:

- Support for both on-demand and live video streaming:
  - Typically on demand video content is already stored in fragmented, ready to stream files, while live feeds (RTMP stream) are going through a process which creates on-the-fly MP4 fragmented media and virtual manifest files.
- Adaptive bitrate streaming:
  - This is an extremely important functionality which allows the client to perform real-time upshifts or downshifts to different quality levels (specified in manifest files) based on various factors such as bandwidth and computing power.
- Media navigation support:
  - This is the ability to perform SEEK type commands in a video without having to download it first.
- Video delivery throttling:
  - This capability allows bandwidth saving as it does not download the media segments which are not watched by the user.
- Quality of Service (QoS) monitoring.
- DVR-like functionality.
- Content protection:
  - Protected media delivery can be achieved by using Flash Access.

Considering that HDS uses HTTP as transport, existing caching infrastructure can be leveraged to extend capacity and reach. To deliver high-quality video, HDS supports codecs such as VP6/MP3 and H.264/AAC.



## Test Case: Basic Adobe® HTTP Dynamic Streaming Client

For more information related to HDS specifications and functionality, please consult the technical Whitepaper available on the Adobe web site at:

[http://www.adobe.com/products/httpdynamicstreaming/pdfs/httpdynamicstreaming\\_wp\\_ue.pdf](http://www.adobe.com/products/httpdynamicstreaming/pdfs/httpdynamicstreaming_wp_ue.pdf)

### Objective

Create a test profile in IxLoad to act as an Adobe HDS Player and stream content from an Adobe HDS-Compatible Video Server (Flash Media Server 4.5).

The test case will outline how to configure Adobe HDS player in IxLoad and examine real-time statistics for key performance indicators. The PLAY command is used to retrieve the manifest file and automatically parse it to play the media stream at the default bitrate until the next test or end of stream.

### Setup

The following scenario is used:

Windows 2008 Server running Flash Media Server 4.5 to host HDS-compatible content and IxLoad 6.10 HDS Player Activity.

One Ixia test port is connected directly to the Flash Media Server (FMS). The topology shown in Figure 109 is representative of a more complex scenario which features an optional proxy and a CDN infrastructure hosting the content sent to the FMS.

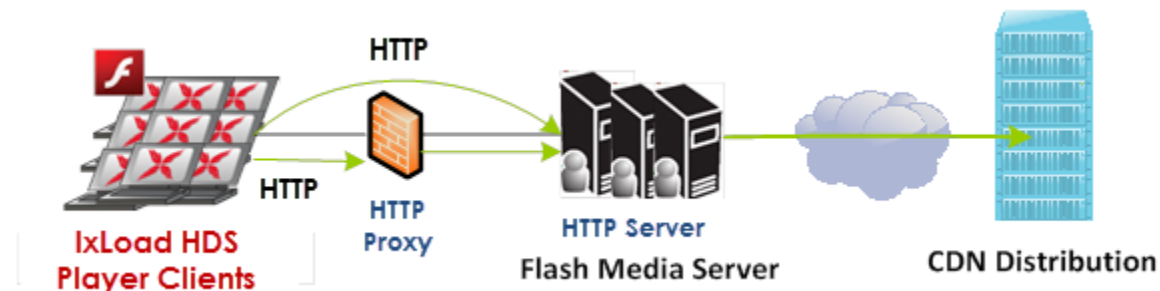
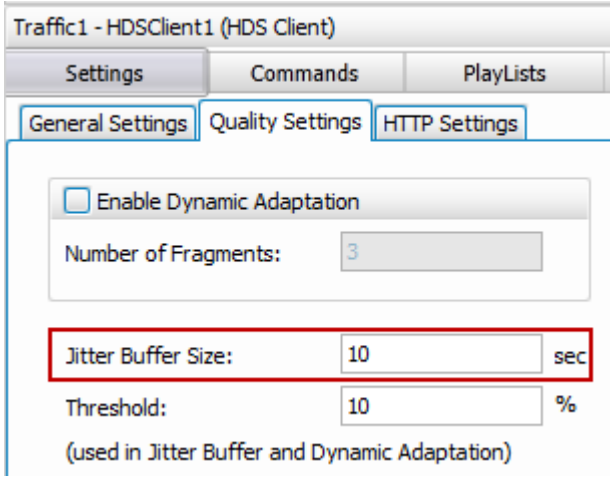


Figure 111. HTTP Dynamic Streaming test setup

## Test Variables

### Test Tool Variables

Parameters	Description
HDS Player Clients	According to the <b>Simulated Users</b> objective value (For details, please refer to the Step-by-step Instructions).
TCP parameters	TCP RX and TX buffer at 32768 bytes.
HDS Player client command list	Use PLAY command. Please refer to the Step-by-step Instructions for details on how to configure basic playback command.
Buffer configuration	<p>By default, a 'finite' value of 10 sec is used.</p> <p>With an "infinite" value (i.e. 0 sec), all the segment 'GET' requests are sent at the beginning of the stream playback.</p> <p>With a 'finite' value, the buffer size (in seconds) can be configured. With a finite buffer size, the segment 'GET' requests are timed to fill the buffer and send continuously.</p>  <p>The screenshot shows the 'Traffic1 - HDSClient1 (HDS Client)' settings window. The 'HTTP Settings' tab is selected. Under 'HTTP Settings', the 'Enable Dynamic Adaptation' checkbox is unchecked. The 'Number of Fragments' is set to 3. The 'Jitter Buffer Size' is set to 10 seconds, which is highlighted with a red box. The 'Threshold' is set to 10%.</p> <p style="text-align: center;"><b>Figure 112. HDS Player buffer size</b></p>
HDS Proxy support	If clients connect through a content switch or a proxy, IxLoad can send all HTTP packets to this configured <b>IP:port</b> .

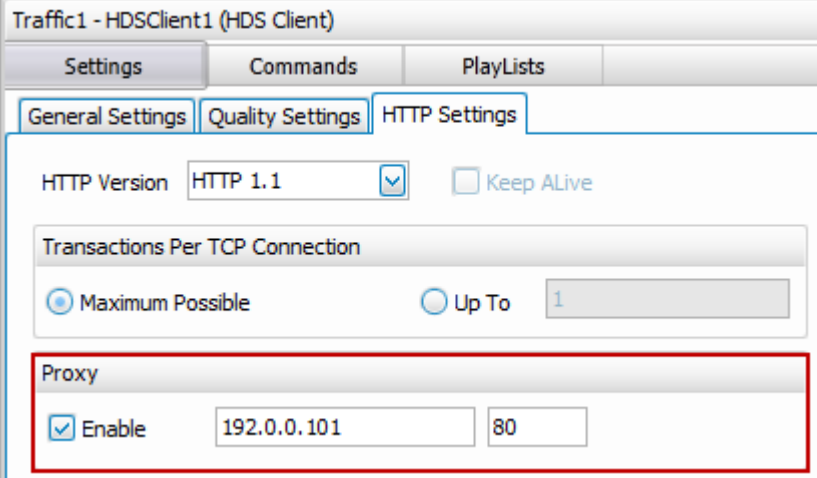
Parameters	Description
	 <p style="text-align: center;">Figure 113. Configuring HTTP proxy for HDS player</p>
<p>Use of sequence generator</p>	<p>Sequence generator can be used to expand the <b>Media URL</b> name requested in the PLAY command. For example, <b>videosample[1-5]_Manifest.f4m</b> can be used to expand to videosample1_Manifest.f4m through videosample5_Manifest.f4m.</p> <p>Each user in the test starts with the next index, that is, user1 starts with videosample1_Manifest.f4m, user2 starts with videosample2_Manifest.f4m, and cycles through the sequence.</p>

Table 30. Test Tool Variables

DUT Test Variables

Device(s)	Variation	Description
Flash Media Server	Start Apache Web server	Make sure that the FMS is installed along with the build in Apache server and that it is properly running.

Table 31. DUT Variables

## Step-by-step Instructions

1. Start IxLoad. In the main window, the **Scenario Editor** window opens. All test configuration is performed here.
2. To get familiar with the IxLoad GUI, please refer to the Getting Started Guide section.
3. Add the client NetTraffic object. Configure the client network with total IP count, gateway, and VLAN, if used.
4. For a step-by-step workflow, see Annex A.
5. Configure the HDS Player client. Add the **HDS** activity to the client NetTraffic.
6. **For basic PLAYBACK**, configure the command list as follows:



7. Add a **PLAY** command and configure the following parameters:
  - Enter the destination server by IP or hostname (enable DNS to use this). If the server is listening on a non-standard port, it can be specified as IP:port.
  - Enter the **Media URL** field: it must contain the F4M manifest file of the media to retrieve. Regardless of the server under test, which could have different formats, when configuring the PLAY command, the **Media URL** field should always point to a manifest file and never to the actual media file.  
A valid example can be: `'vod/video_sample_manifest.f4m'`.
  - Choose 'Till End' as the play duration.
  - The 'Range' option can be used to specify a time values range. Each user randomly chooses playback values within this range.

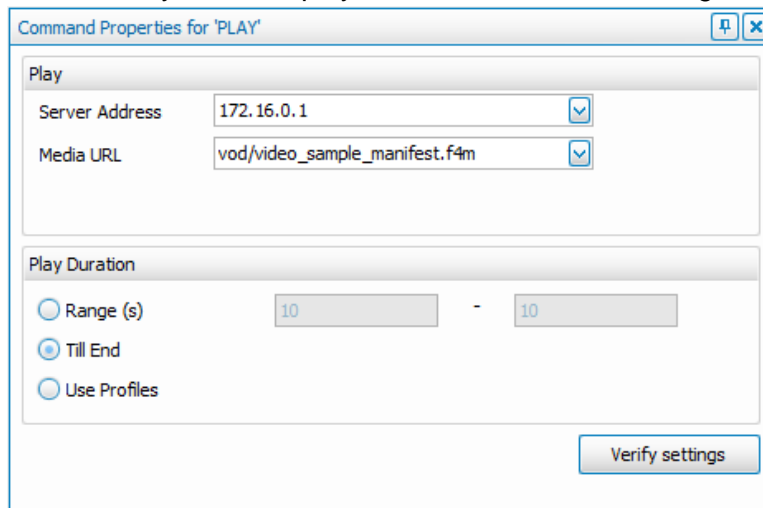


Figure 114. PLAY command configuration for basic HDS Player playback

8. Add a **STOP** command to ensure graceful teardown of sessions at the end of the test or when the test is stopped manually.

Test Case: Basic Adobe® HTTP Dynamic Streaming Client

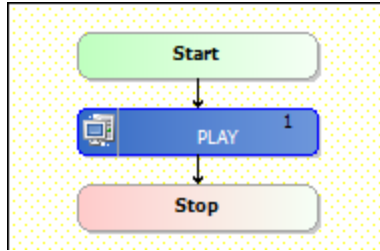


Figure 115. HDS Player command list example

9. After you set up the HDS Player activity, configure simulated user as the test objective and set the objective value to the desired number of users.
10. Add test port resources and run the test. Please refer to the Results Analysis and the Troubleshooting and Diagnostics sections for further details.

## Results Analysis

Metric	Key Performance Indicators	Statistics View
Performance Metrics	Simulated Users, Streams VoD/Live Attempted/Played Successful/Played With Error, Active VoD/Live Streams Total/RX/TX HTTP Throughput	HDS Client - Objective HDS Client – Streams HDS Client - Active Streams HTTP Throughput
Application Level Transactions Application Level Failure Monitoring	Fragments Requested/Request Successful/Request Failed, Manifest Error, Stream Response Error Manifest Requested/Request Successful/Request Failed, Successful	HDS Client - Fragments HDS Client – Errors HDS Client - Manifest
TCP Statistics	TCP SYN, SYN-ACK, FIN, Retries, Timeouts TCP Connections Requests Failed, Resets	TCP Stats, TCP Failures
HTTP Level Failure Monitoring	HTTP Requests Failed, HTTP Session Timeouts/Rejected	HTTP Failures

Table 32. Results Analysis for HDS Player playback

### Real-time Statistics

Adobe HDS activity in IxLoad provides TCP and HTTP level statistics, all Manifests and Fragments statistics, and VoD/Live streams status.

The **HDS Client – Active Streams** view shows the concurrent number of VoD/live streams.

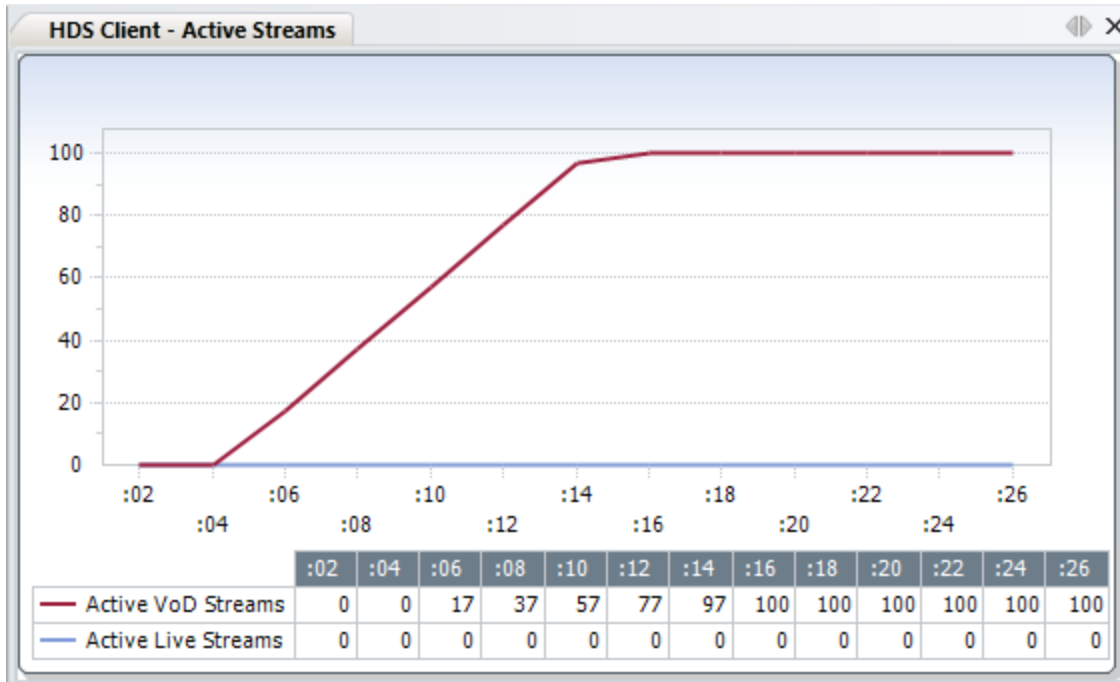


Figure 116. HDS Client – Active Streams view

## Test Case: Basic Adobe® HTTP Dynamic Streaming Client

The **HDS Client - Streams** view updates at the end of a stream playback. For example, if the stream duration is 120 seconds, this view updates after 120 seconds elapse. If one or more fragment requests fail, the test continues to run, but the 'Streams Played with Error' indicate such transient and important error conditions.

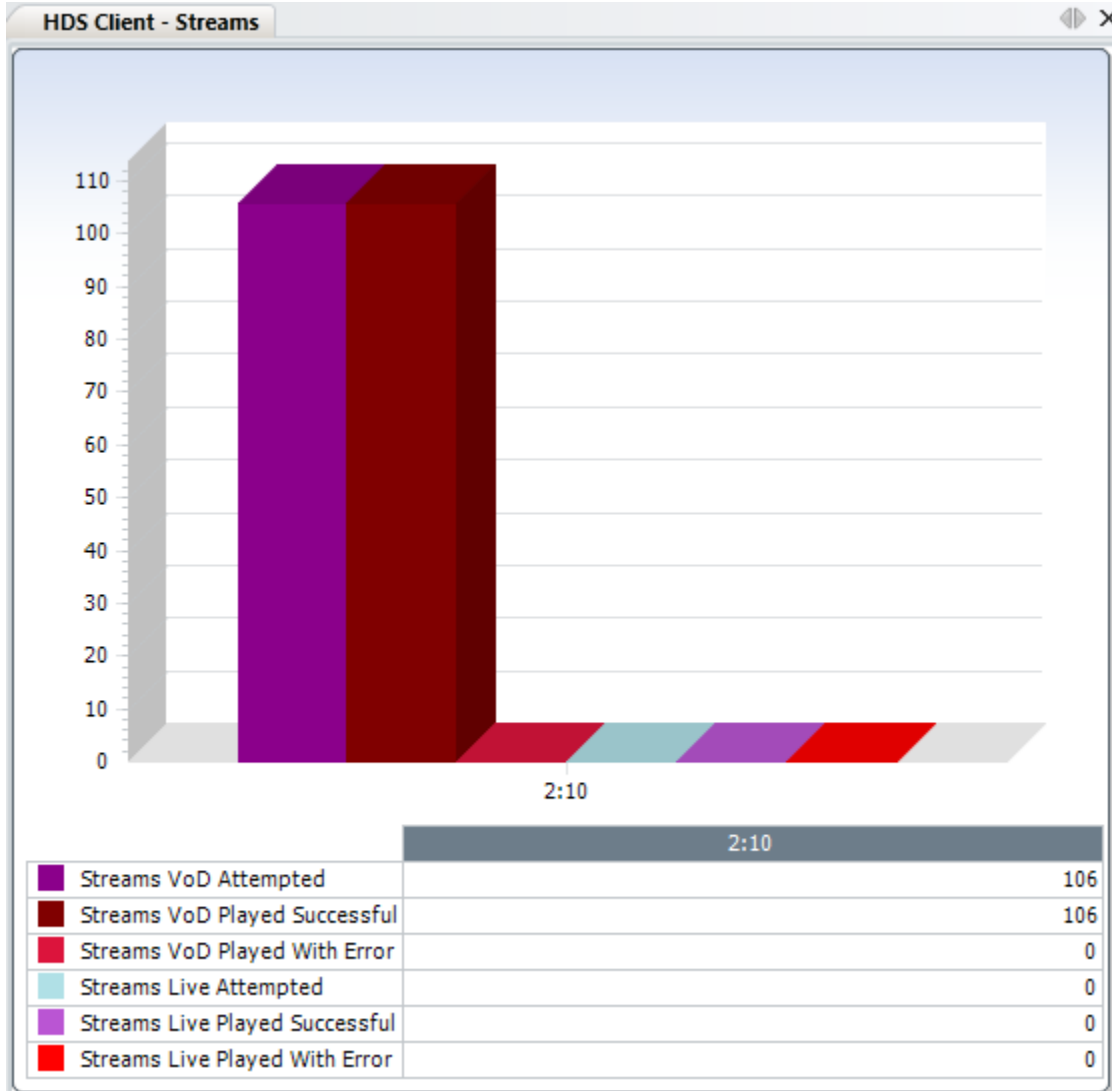


Figure 117. HDS Client – Streams view

The **HDS Client - Fragments** view shows all the media fragments that were requested and how many of them failed or succeeded.

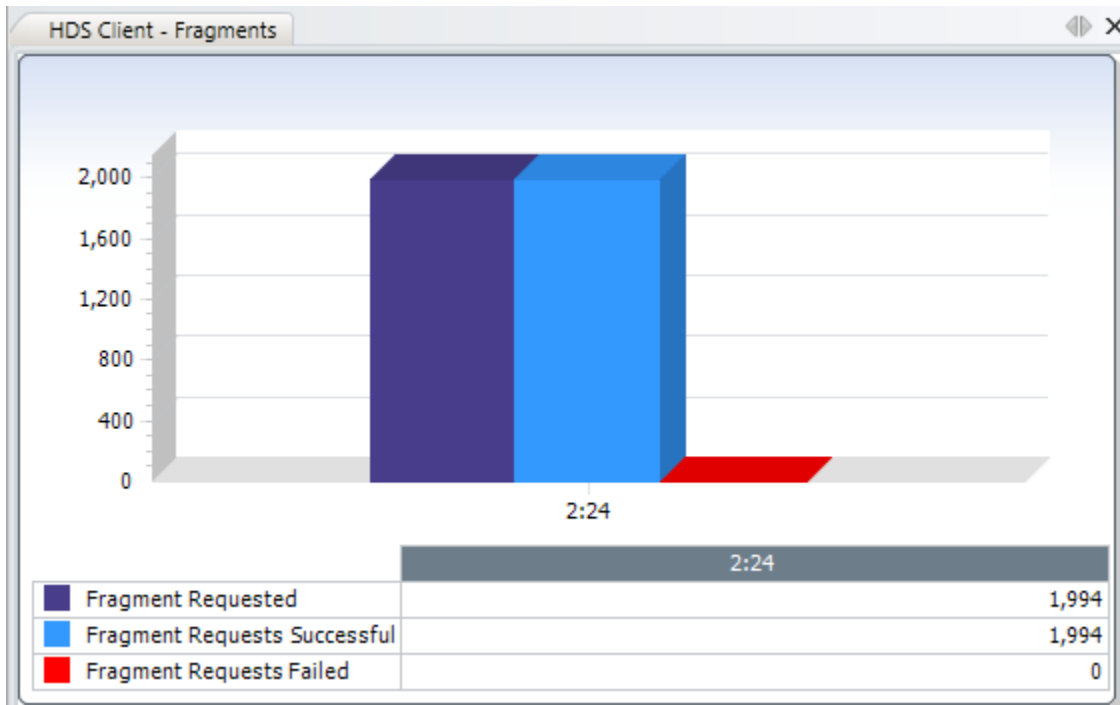


Figure 118. HDS Client – Fragments



The **HDS Client - Manifests** view shows all the manifests that were requested and how many of them failed or succeeded.

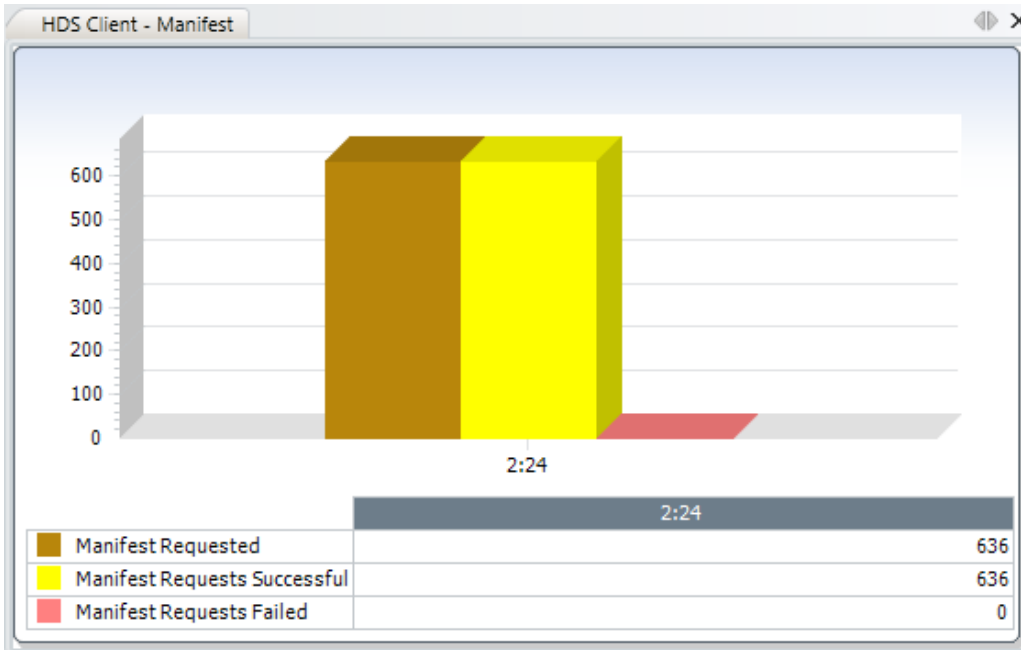


Figure 119. HDS Client – Manifests

The **HDS Client - Errors** view indicates the number of errors encountered by the client while parsing the manifests (that is, Manifest Error) or stream requests for which the client received an error in response (that is, Stream Response Error).

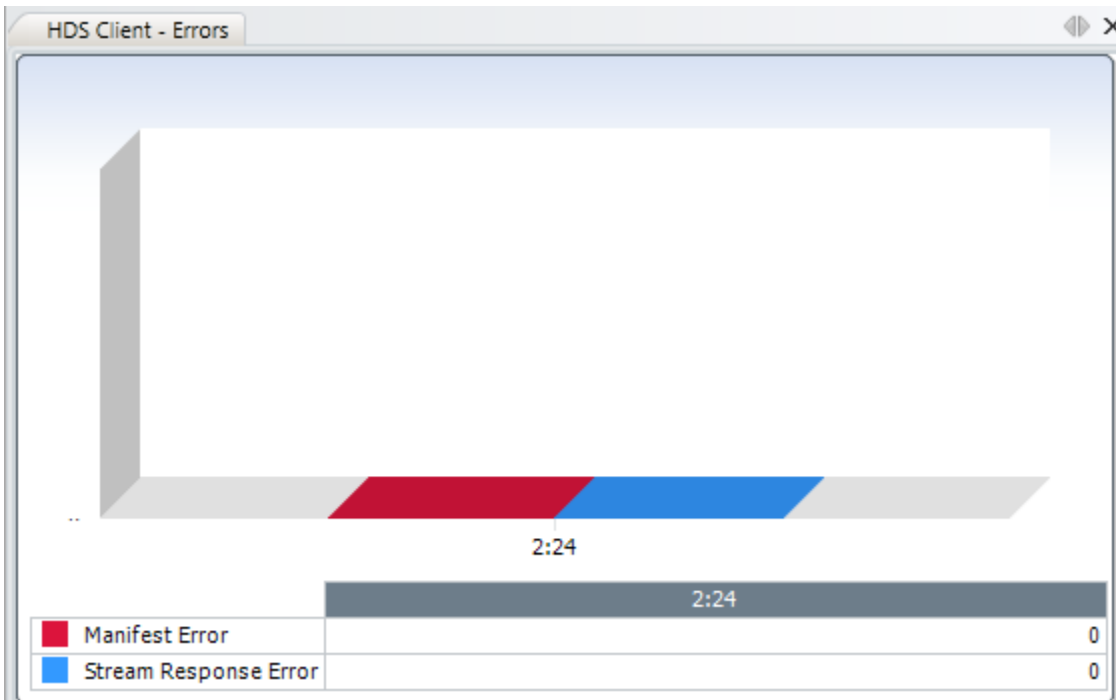


Figure 120. HDS Client – Errors

## Troubleshooting and Diagnostics

Issue	Diagnosis, Suggestions
No SYNs sent.	Test port cannot resolve gateway or server IP to MAC. Resolve routing or switch issue. Are VLANs required?
Manifest request fails.	Is proxy configured? Is it required? Is the media URL correct? Ensure that in the PLAY command the manifest is requested and not the media file.
Partial fragment requests are failing.	The network may be dropping packets, the server may be congested and causing issues, the manifest file is not correctly constructed, or media files are missing from the server.

Table 33. Troubleshooting checklist



## Test Case: Adobe® HTTP Dynamic Streaming with Pre-selected or Automatic Adaptive Bitrate Shifting

### Overview

Adobe®'s HTTP Dynamic Streaming (HDS) is a high-quality video delivery mechanism which provides an alternative to the RTMP streaming-based methods, especially for devices which do not natively use Flash Player. HDS does not rely on expensive and specialized streaming servers or closed and inaccessible proprietary protocols, enabling on-demand and live adaptive bitrate video delivery over regular HTTP connections.

HDS is streaming video content in small chunks called fragments (F4F file format), which are based on standard open-source MP4 fragments (fMP4).

Along with the video fragments file itself, manifest files (F4M files in XML based format) play an important part in HDS streaming. They contain information about available bitrates, fragment format, metadata information, and other properties available for a video presentation.

Some of the HDS technology benefits are:

- Support for both on-demand and live video streaming:
  - Typically on demand video content is already stored in fragmented, ready to stream files, while live feeds (RTMP stream) are going through a process which creates on-the-fly MP4 fragmented media and virtual manifest files.
- Adaptive bitrate streaming:
  - This is an extremely important functionality which allows the client to perform real-time upshifts or downshifts to different quality levels (specified in the manifest file) based on various factors such as bandwidth and computing power.
- Media navigation support:
  - This is the ability to perform SEEK type commands in a video without having to download it first.
- Video delivery throttling:
  - This capability allows bandwidth saving as it does not download the media segments which the user does not watch.
- Quality of Service (QoS) monitoring.
- DVRlike functionality.
- Content protection:
  - Protected media delivery can be achieved by using Flash Access.

Considering that HDS uses HTTP as transport, existing caching infrastructure can be leveraged to extend capacity and reach. To deliver high-quality video, HDS supports codecs such as VP6/MP3 and H.264/AAC.

## Test Case: Adobe® HTTP Dynamic Streaming with Pre-selected or Automatic Adaptive Bitrate Shifting

As previously mentioned, one very important benefit of HDS is adaptive bitrate streaming which empowers the video client to make decisions such as automatically shifting to a higher or lower video quality (upshifting/downshifting) depending on certain factors such as bandwidth and computing power. This type of decision is made by the video client alone, in a dynamic manner.

For more information about HDS specifications and functionality, please consult the technical Whitepaper available on the Adobe web site at:

[http://www.adobe.com/products/httpdynamicstreaming/pdfs/httpdynamicstreaming\\_wp\\_ue.pdf](http://www.adobe.com/products/httpdynamicstreaming/pdfs/httpdynamicstreaming_wp_ue.pdf)

### Objective

Create a test profile in IxLoad to act as an Adobe HDS Player and stream content from an Adobe HDS Compatible Video Server (Flash Media Server 4.5), using adaptive bitrate shifts to emulate different users upshifting or downshifting to different levels of media quality during playback.

The test case will outline how to configure the Adobe HDS player in IxLoad so as to use profiles and create a scenario to upshift and downshift to different video streams bitrate during playback.

Profiles are used to create percentages of users taking different shifting actions. The up or down shift is performed by using a deterministic and repeatable bit-rate shift table. The PLAY command is used to retrieve the manifest file and automatically parse it to play the media stream.

Real-time statistics for key performance indicators are examined during test run.

## Test Case: Adobe® HTTP Dynamic Streaming with Pre-selected or Automatic Adaptive Bitrate Shifting

### Setup

The following scenario is used:

Windows 2008 Server running Flash Media Server 4.5 to host HDS-compatible content and IxLoad 6.10 to emulate HDS Player Activity with adaptive bitrate streaming capabilities.

One Ixia test port is connected directly to the Flash Media Server (FMS). The topology shown in Figure 119 is representative of a more complex scenario which features an optional proxy and a CDN infrastructure hosting the content sent to the FMS.

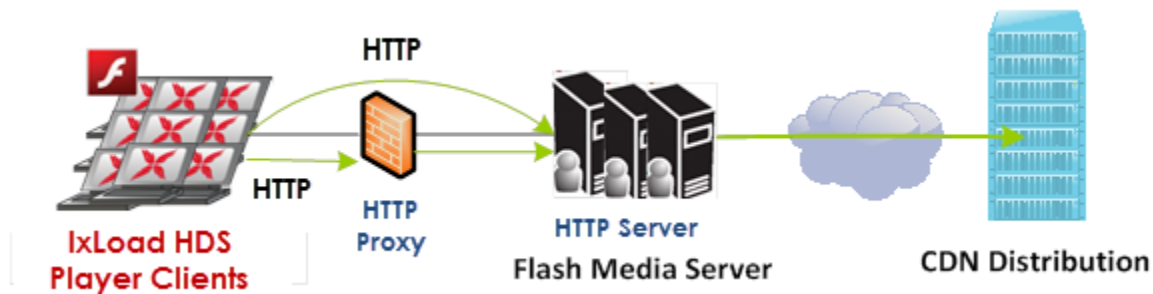


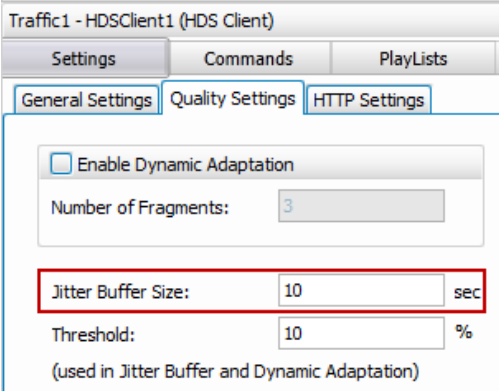
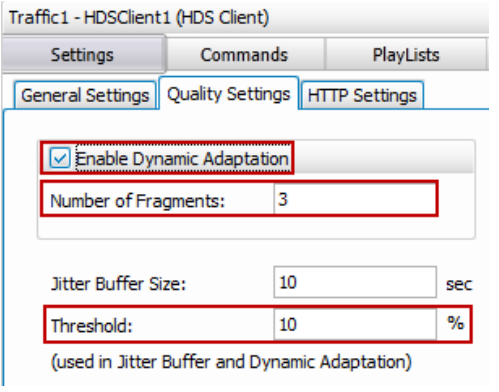
Figure 121. HTTP Dynamic Streaming test setup

### Test Variables

#### Test Tool Variables

Parameters	Description
HDS Player Clients	According to the <b>Simulated Users</b> objective value (For details, please refer to the Step-by-step Instructions).
TCP parameters	TCP RX and TX buffer at 32768 bytes.
HDS Player client command list	Use PLAY command. (Please refer to the Step-by-step Instructions for details on how to configure basic playback command.
Buffer configuration	By default, a 'finite' value of 10 sec is used.  With an "infinite" value (that is, 0 sec), all the segment 'GET' requests are sent at the beginning of the stream playback.  With a 'finite' value, the buffer size (in seconds) can be configured. With a finite buffer size, the segment 'GET' requests are timed to fill the buffer and send continuously.

Test Case: Adobe® HTTP Dynamic Streaming with Pre-selected or Automatic Adaptive Bitrate Shifting

Parameters	Description
	 <p style="text-align: center;"><b>Figure 122. HDS Player buffer size</b></p>
<p>Adaptive bitrate settings</p>	<p>Adaptive bitrate can be enabled by using either of the following two models:</p> <ol style="list-style-type: none"> <li><b>Using Profiles</b> (as described in this Test Case): This option allows static bitrate shifting patterns which can be customized by users by using a bitrate shift table. (Please refer to the Step-by-step Instructions for more details).</li> <li><b>Dynamic Adaptation:</b> This option is useful if automatic bitrate shifting is required. The emulated HDS player makes real-time decisions whether to upshift or downshift to a different quality level (according to the manifest file) based on dynamically-computed available network bandwidth. Users can also configure how aggressive the shifts should be using parameters such as <b>Number of Fragments</b> and <b>Threshold</b>.</li> </ol>  <p style="text-align: center;"><b>Figure 123. Dynamic Adaptation parameters</b></p>
<p>HDS Proxy support</p>	<p>If the clients connect through a content switch or a proxy, IxLoad can send all HTTP packets to this configured <b>IP:port</b>.</p>

Test Case: Adobe® HTTP Dynamic Streaming with Pre-selected or Automatic Adaptive Bitrate Shifting

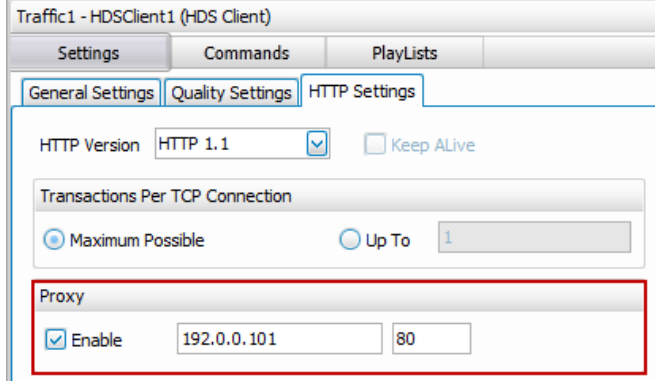
Parameters	Description
	 <p style="text-align: center;">Figure 124. Configuring HTTP proxy for HDS player</p>
Use of sequence generator	<p>Sequence generator can be used to expand the <b>Media URL</b> name requested in the PLAY command. For example, <b>videosample[1-5]_Manifest.f4m</b> can be used to expand to videosample1_Manifest.f4m through videosample5_Manifest.f4m.</p> <p>Each user in the test starts with the next index, that is, user1 starts with videosample1_Manifest.f4m, user2 starts with videosample2_Manifest.f4m, and cycles through the sequence.</p>

Table 34. Test Tool Variables

DUT Test Variables

Device(s)	Variation	Description
Flash Media Server	Start Apache Web server	Make sure that the FMS is installed along with the build in Apache server and that it is properly running.

Table 35. DUT Variables



## Step-by-step Instructions

1. Start IxLoad. In the main window, the **Scenario Editor** window opens. All test configuration is performed here.
2. To get familiar with the IxLoad GUI, please refer to the Getting Started Guide section.
3. Add the client NetTraffic object. Configure the client network with total IP count, gateway, and VLAN, if used.
4. For a step-by-step workflow, see Annex A.
5. Configure the HDS Player client. Add the **HDS** activity to the client NetTraffic.
6. **For basic PLAYBACK**, configure the command list as follows:



7. Add a **PLAY** command and configure the following parameters:
  - Enter the destination server by IP or hostname (enable DNS to use this). If the server is listening on a non-standard port, it can be specified as IP:port.
  - Enter the **Media URL** field: it must contain the F4M manifest file of the media to retrieve. Regardless of the server under test, which could have different formats, when configuring the PLAY command, the **Media URL** field should always point to a manifest file and never to the actual media file.  
A valid example can be: `'vod/video_sample_manifest.f4m'`.
  - Choose 'Use Profiles' as the play duration. This option allows multiple groups of users to upshift or downshift to different media quality.

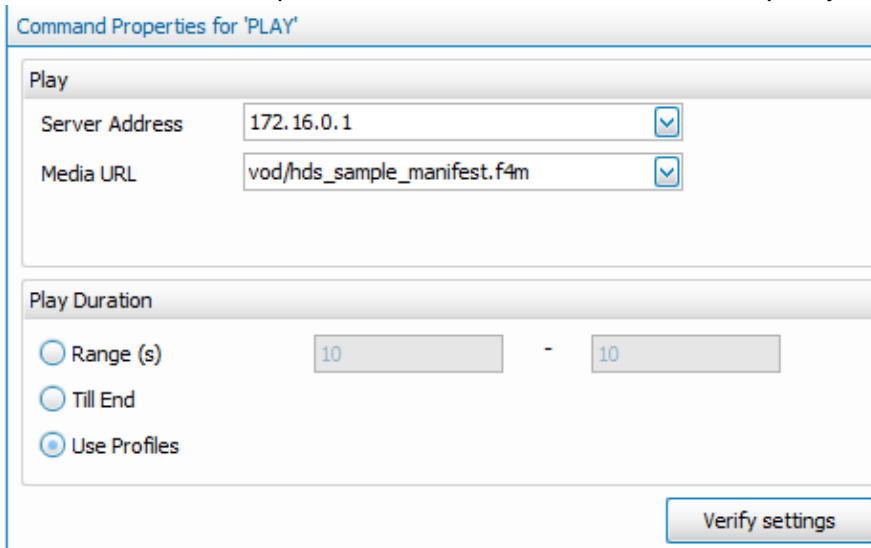
The screenshot shows a dialog box titled 'Command Properties for PLAY'. It has two main sections: 'Play' and 'Play Duration'. In the 'Play' section, there are two fields: 'Server Address' with the value '172.16.0.1' and a dropdown arrow, and 'Media URL' with the value 'vod/hds\_sample\_manifest.f4m' and a dropdown arrow. In the 'Play Duration' section, there are three radio button options: 'Range (s)' (unselected), 'Till End' (unselected), and 'Use Profiles' (selected). The 'Range (s)' option has two input boxes, both containing the number '10', separated by a hyphen. At the bottom right of the dialog is a 'Verify settings' button.

Figure 125. PLAY command configuration for basic HDS Player playback

8. Go to the **Settings** tab to set up user profiles and their upshift and downshift capabilities. Here you can also set the percentage of profiles across users.
9. Add a new profile in the **Stream Viewing Profiles** window:

# Test Case: Adobe® HTTP Dynamic Streaming with Pre-selected or Automatic Adaptive Bitrate Shifting

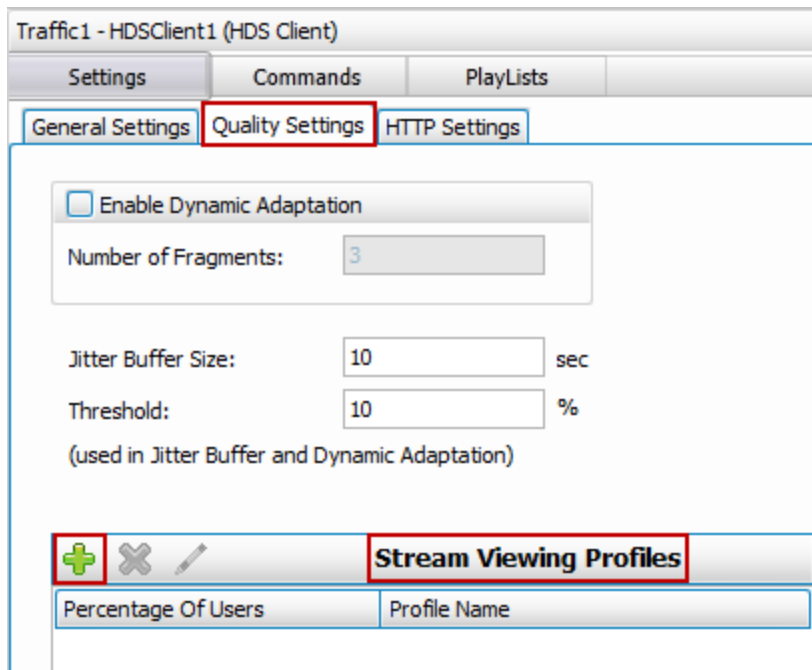


Figure 126. Streaming Viewing Profiles for bitrate shift profiles

- The 'Edit Viewing Profile' window opens. The table available here allows you to configure upshift, downshift, or jump to the lowest (default) bitrate or to the highest bitrate. For each shifting action, the user can set how many levels to shift up or down, and for how long to watch a particular stream after shifting to it.
- In this scenario, the server contains five levels of quality for the playback of the video file (150 kbps, 500 kbps, 700kbps, 1000 kbps, and 1500 kbps). This information is available in the corresponding video manifest file.
- The Viewing Profile shown in Figure 125 shifts up two levels and downshifts one level. A Play Duration of 0 means 'Play until end,' which is automatically determined from the manifest file.

Test Case: Adobe® HTTP Dynamic Streaming with Pre-selected or Automatic Adaptive Bitrate Shifting

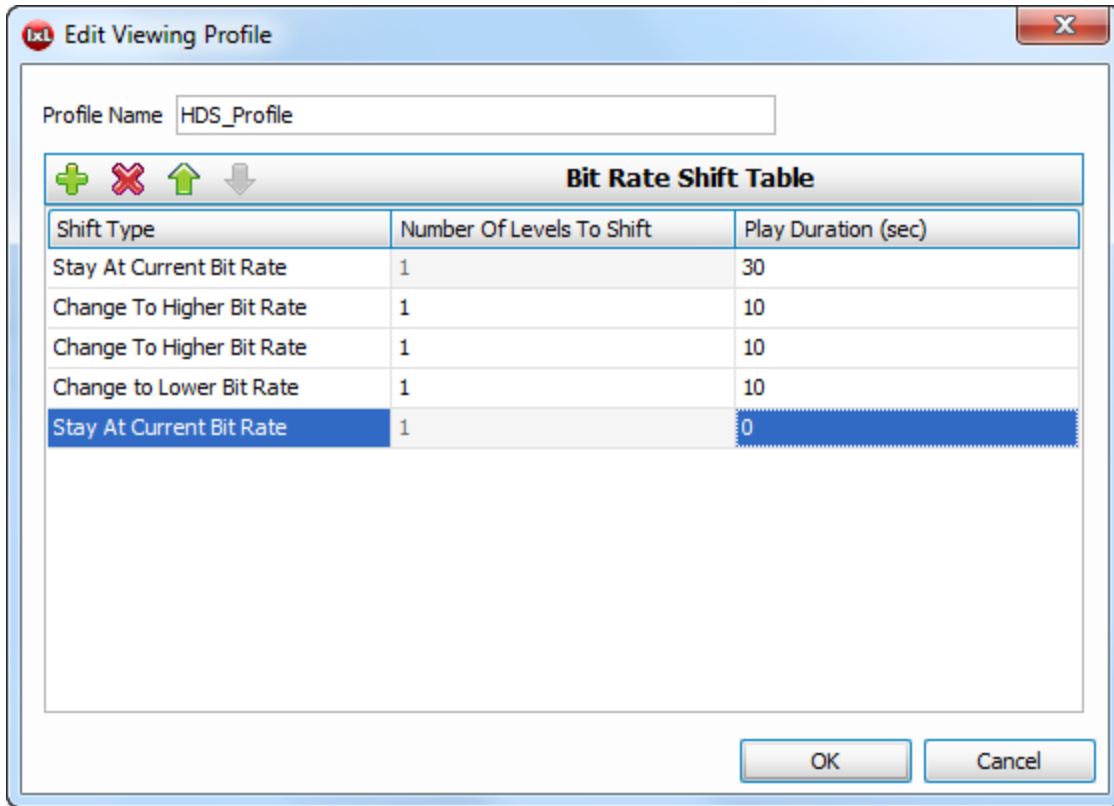


Figure 127. Configuring an adaptive viewing profile

10. Add a **STOP** command to ensure graceful teardown of sessions at the end of the test or when the test is stopped manually.

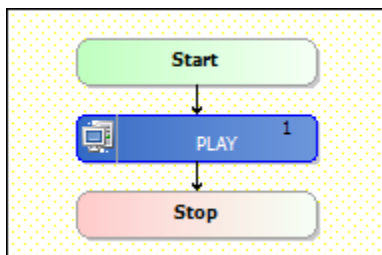


Figure 128. HDS Player command list example

11. After you set up the HDS Player activity, configure simulated user as the test objective and set the objective value to the desired number of users.
12. Add test port resources and run the test. Please refer to the Results Analysis and the Troubleshooting and Diagnostics sections for further details.

## Results Analysis

Metric	Key Performance Indicators	Statistics View
Performance Metrics	Simulated Users, Streams VoD/Live Attempted/Played Successful/Played With Error, Active VoD/Live Streams Total/RX/TX HTTP Throughput Bitrate shifts Streams/Fragments Bitrates	HDS Client - Objective HDS Client – Streams HDS Client - Active Streams HTTP Throughput HDS Client – Fragments Bitrate Distribution HDS Client – Playback Bitrates
Application Level Transactions	Fragments Requested/Request Successful/Request Failed,	HDS Client - Fragments HDS Client – Errors
Application Level Failure Monitoring	Manifest Error, Stream Response Error Manifest Requested/Request Successful/Request Failed, Successful	HDS Client - Manifest
TCP Statistics	TCP SYN, SYN-ACK, FIN, Retries, Timeouts TCP Connections Requests Failed, Resets	TCP Stats, TCP Failures
HTTP Level Failure Monitoring	HTTP Requests Failed, HTTP Session Timeouts/Rejected	HTTP Failures

Table 36. Results Analysis for HDS Player playback

# Test Case: Adobe® HTTP Dynamic Streaming with Pre-selected or Automatic Adaptive Bitrate Shifting

## Real-time Statistics

Adobe HDS activity provides TCP and HTTP level statistics, all Manifests and Fragments statistics, and VoD/Live streams status. Besides the usual performance indicators (refer to the [Basic Adobe HTTP Dynamic Streaming Client](#) test case), other types of statistics specifically related to bitrate shifting are available as well.

The **HDS Client – Bitrate Shift** view indicates in real time all attempted upshifts and downshifts.

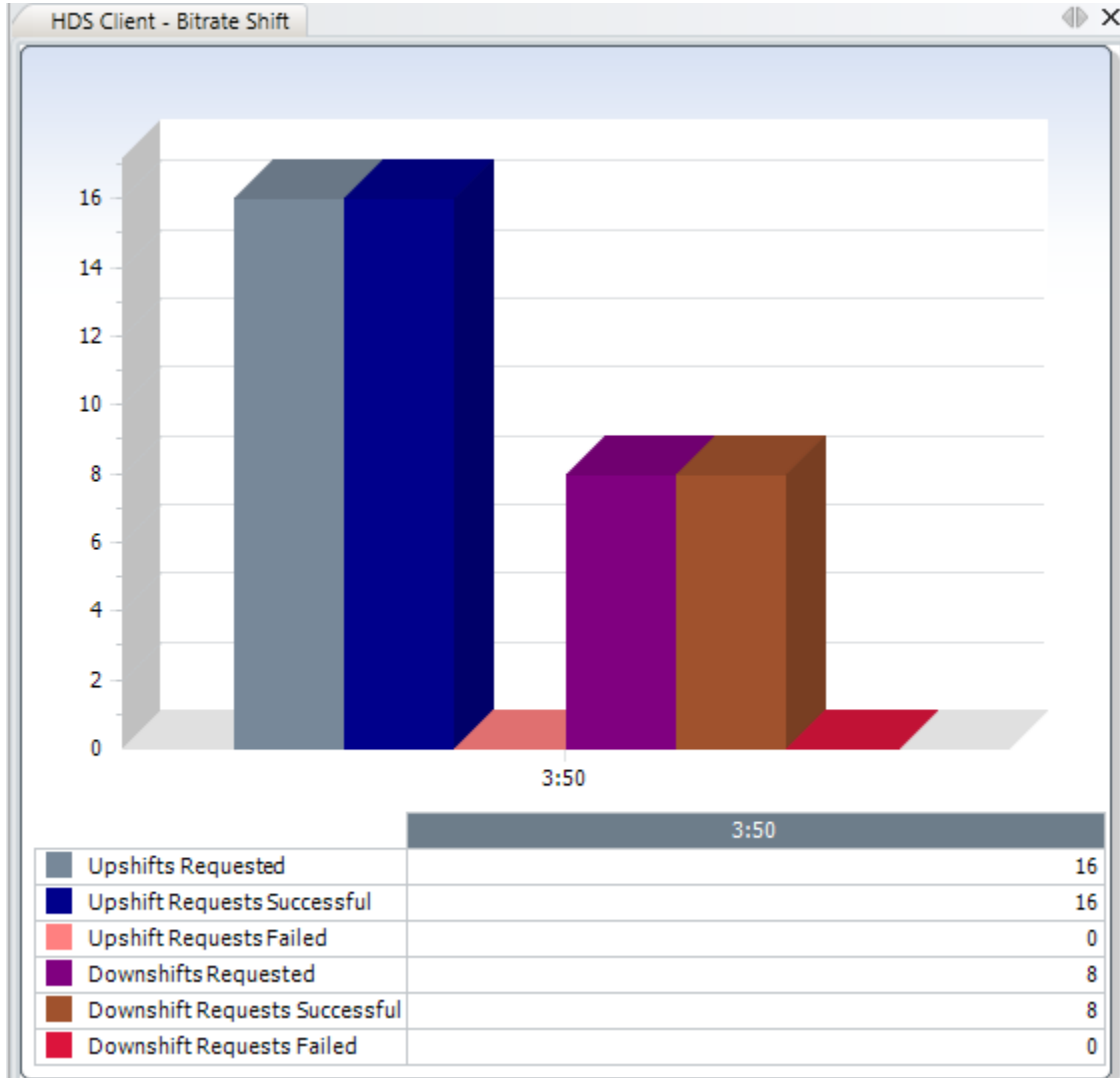


Figure 129. HDS Client – Bitrate Shift view

Test Case: Adobe® HTTP Dynamic Streaming with Pre-selected or Automatic Adaptive Bitrate Shifting

The **HDS Client – Fragments Bitrate Distribution** view groups all streamed fragments in buckets of different bitrate ranges.

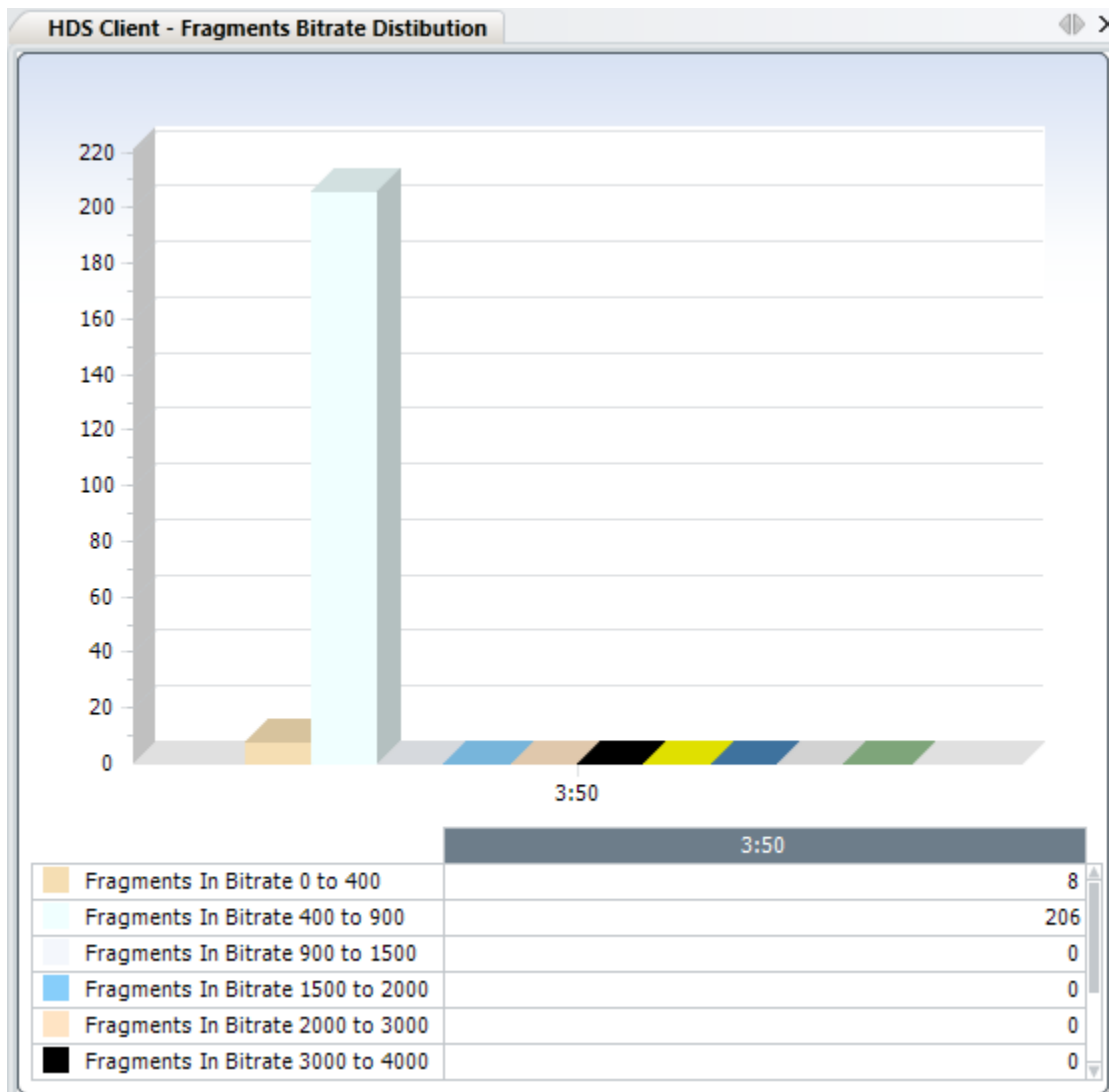


Figure 130. HDS Client – Streams view

Test Case: Adobe® HTTP Dynamic Streaming with Pre-selected or Automatic Adaptive Bitrate Shifting

The **HDS Client – Playback Bitrates** view groups all active streams in buckets of different bitrate ranges.

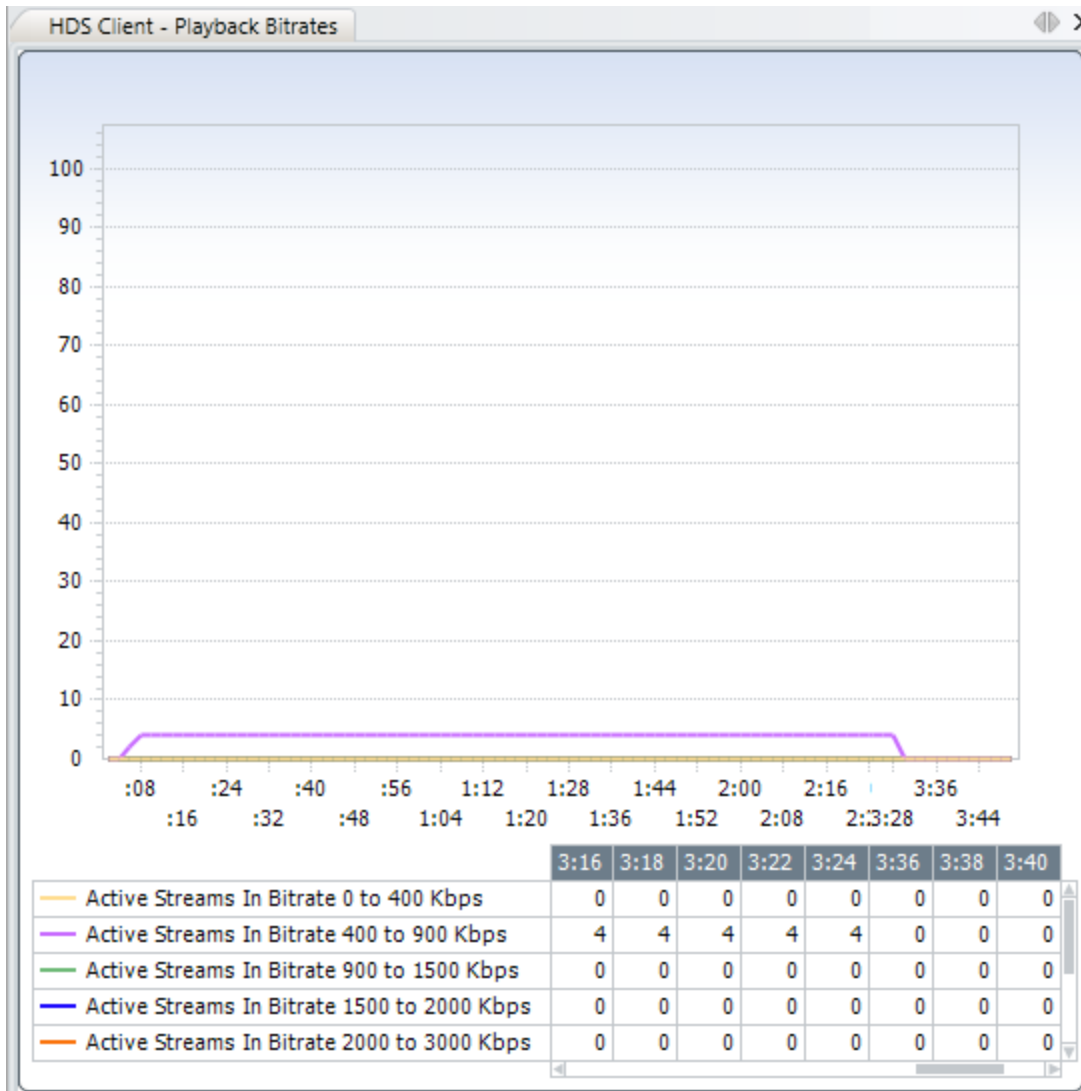


Figure 131. HDS Client – Playback Bitrates view

## Troubleshooting and Diagnostics

Issue	Diagnosis, Suggestions
No SYNs sent.	Test port cannot resolve gateway or server IP to MAC. Resolve routing or switch issue. Are VLANs required?
Manifest request fails.	Is proxy configured? Is it required? Is the media URL field correct? Ensure that in the PLAY command the manifest is requested, not the media file.
Partial fragment requests are failing.	The network may be dropping packets, the server may be congested and causing issues, the manifest file is not correctly constructed, or media files are missing from the server.

Table 37. Troubleshooting checklist





## Test Case: Basic Silverlight® Smooth Streaming Client

### Overview

Microsoft® Silverlight® is a development platform that has been rapidly adopted to create rich interactive applications, and for delivering high quality multimedia streaming to a variety of devices and platforms. The platform runs on .NET framework and it is compatible with a variety of operating systems, browsers, and mobile devices.

Silverlight's most prevalent adoption comes from delivering high quality media, called Smooth Streaming. Specifically, Windows 2008 Server with IIS Smooth Streaming enables adaptive streaming of on-demand and Live streaming video by using HTTP.

IxLoad includes the complete basic Silverlight Player video streaming capability. It supports playback of on-demand and Live media, and it supports Smooth Streaming in a deterministic manner to create suitable up shift and down shift of media quality.

### Objective

Create a test profile in IxLoad to act as a Silverlight Player to stream content from a Silverlight (Smooth Streaming) compatible video server.

The test case will outline how to configure the Silverlight Streaming player in IxLoad and examine real-time statistics for key performance indicators. The PLAY command is used to retrieve the manifest and automatically parse it to play the media stream at the default bitrate until the next of test or end of stream.

## Setup

The following setup is used:

One Ixia test port is connected directly to a Silverlight Streaming Server (Windows 2008 running IIS7 with extensions). The topology shown in the following figure is representative of a more complex scenario with optional HTTP Proxy.

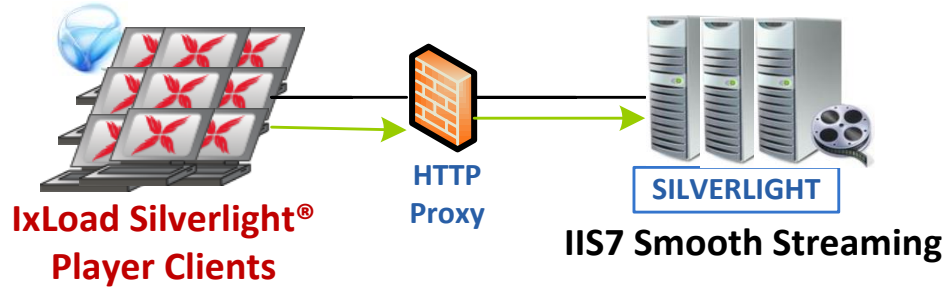


Figure 132. Silverlight Player setup to test Silverlight Compatible Server infrastructure

## Test Variables

### Test Tool Variables

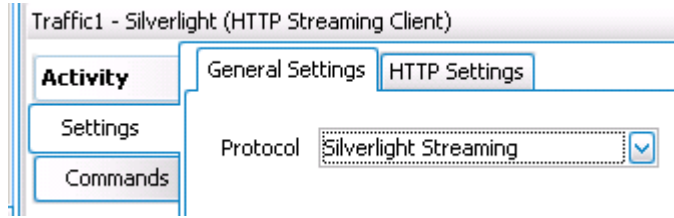
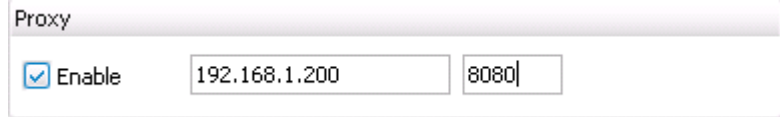
Parameters	Description
Silverlight Clients	100 IP addresses or more, use sequential or 'use all' IP addresses.
Silverlight parameters	<p>Change 'Protocol' in the <b>General</b> tab to 'Silverlight Streaming.'</p>  <p style="text-align: right;">Figure 133. Choosing Silverlight Streaming player</p>
TCP parameters	TCP RX and TX buffer at 32768 bytes.
Silverlight Player client command list	Use PLAY and configure based on the specific server and content name that is present on the server. See example in Step by Step.
Buffer configuration	<p>By default, 'Infinite' is used. All the segment 'GET' requests are sent at the beginning of the stream playback.</p> <p>With 'Finite,' the buffer size (in seconds) can be configured. With a finite buffer size, the segment 'GET' requests are timed to fill the buffer and sent continuously.</p>
Proxy support	<p>If clients connect through a content switch or proxy, IxLoad can send all HTTP packets to this configured <b>IP:port</b>.</p>  <p style="text-align: right;">Figure 134. – Configuring HTTP proxy for Silverlight Player</p>
Use of sequence generator	<p>Sequence generator is used above to expand <b>video[1-5].m4v</b> to video1.m4v through video5.m4v.</p> <p>Each user in the test will start with the next index, that is, user1 starts with video1.m4v, user2 starts with video2.m4v, and cycles through the sequence.</p>

Table 38. Test Tool Variables

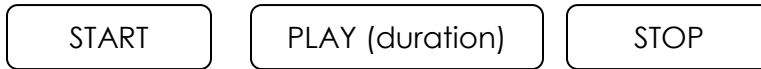
DUT Test Variables

Device(s)	Variation	Description
Silverlight Compatible Server	Enable on-demand and smooth streaming	Each server may use a different semantic to reach the Manifest or the adaptive streaming Manifest. Check and confirm that it works with a real Silverlight Player.

Table 39. DUT Variables

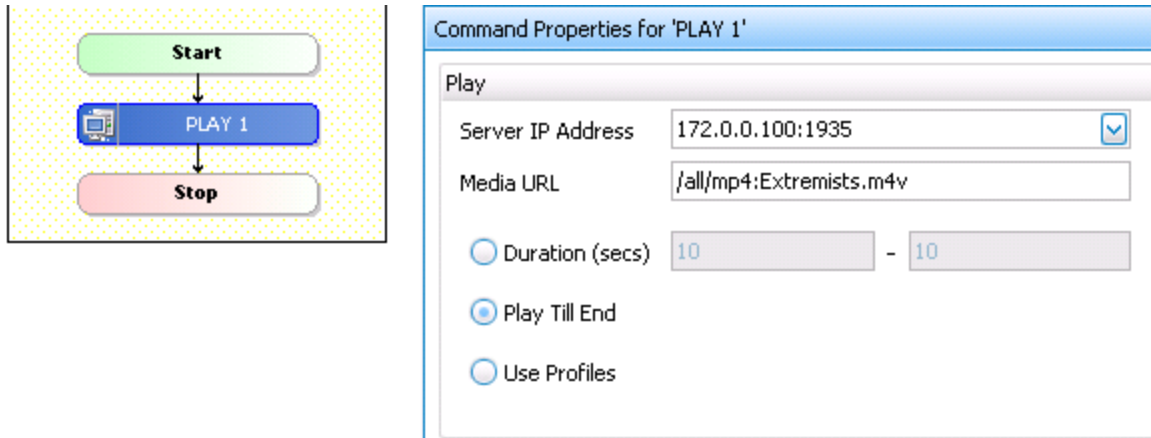
Step-by-Step Instructions

1. Start IxLoad. In the main window, the **Scenario Editor** window appears. All test configurations will be done here.
2. To get familiar with the IxLoad GUI, see the Getting Started Guide section.
3. Add the client NetTraffic object. Configure the client network with total IP count, gateway, and VLAN, if used.
4. For a step-by-step workflow, see Annex A.
5. Configure the Silverlight Player client. Add the **HTTP Streaming** activity to the client NetTraffic.
6. For basic **PLAYBACK**, configure the command list as follows:



7. Add the **PLAY** command and configure the following parameters:
  - Choose Silverlight Streaming for Protocol from the **General** tab.
  - Enter the destination server by IP or hostname (enable DNS to use this). If the server is listening on a non-standard port, it can be specified as IP:port.
  - Enter the media URL. The media name must follow the exact convention that the Silverlight server requires. Each implementation can vary. Do not specify the '/Manifest' in the end of the media URL. It is always automatically added. For example, '/all/mp4:media.m4v' would be sent as '/all/mp4:media.m4v/**Manifest.**'
  - Choose 'Play Till End' as the play duration.
  - The 'Duration' option is available to specify a range of time. Each user will randomly choose playback duration within this range.

## Test Case: Basic Silverlight® Smooth Streaming Client



**Figure 135.** – Configuration of PLAY command for Silverlight Player

8. Having set up the HTTP Streaming (Silverlight) activity, set the test objective. Configure simulated user as the test objective and set the objective value to the desired number of users.
9. To ensure that no interoperability or configuration issues exist (such as wrong content name), you can run a one simulated user test and enable capture on the port to see the packet exchange for possible issues.
10. Add test port resources and run the test. Refer to the Results Analysis and the Troubleshooting and Diagnostics section for further information.

## Results Analysis

Metric	Key Performance Indicators	Statistics View
Performance Metrics	Streaming Users, Played, Paused RX/TX Throughput, Audio/Video/Media Throughput	Silverlight Client - Active Streams Silverlight Client - Throughput Silverlight Client - Streams
Application Level Transactions  Application Level Failure Monitoring	Streams Played, Successful, Played with Error Manifest Requested, Successful Audio/Video/Media Fragments Requested, Successful	Silverlight Client - Fragments Silverlight Client - Failures
TCP Statistics	TCP SYN, SYN-ACK, Connections Established, Failed	TCP Stats, HTTP Failures

**Table 40. Results Analysis for Silverlight Player Playback**

## Real-time Statistics

The HTTP Streaming activity supports real-time statistics for TCP, HTTP, and the Silverlight application.

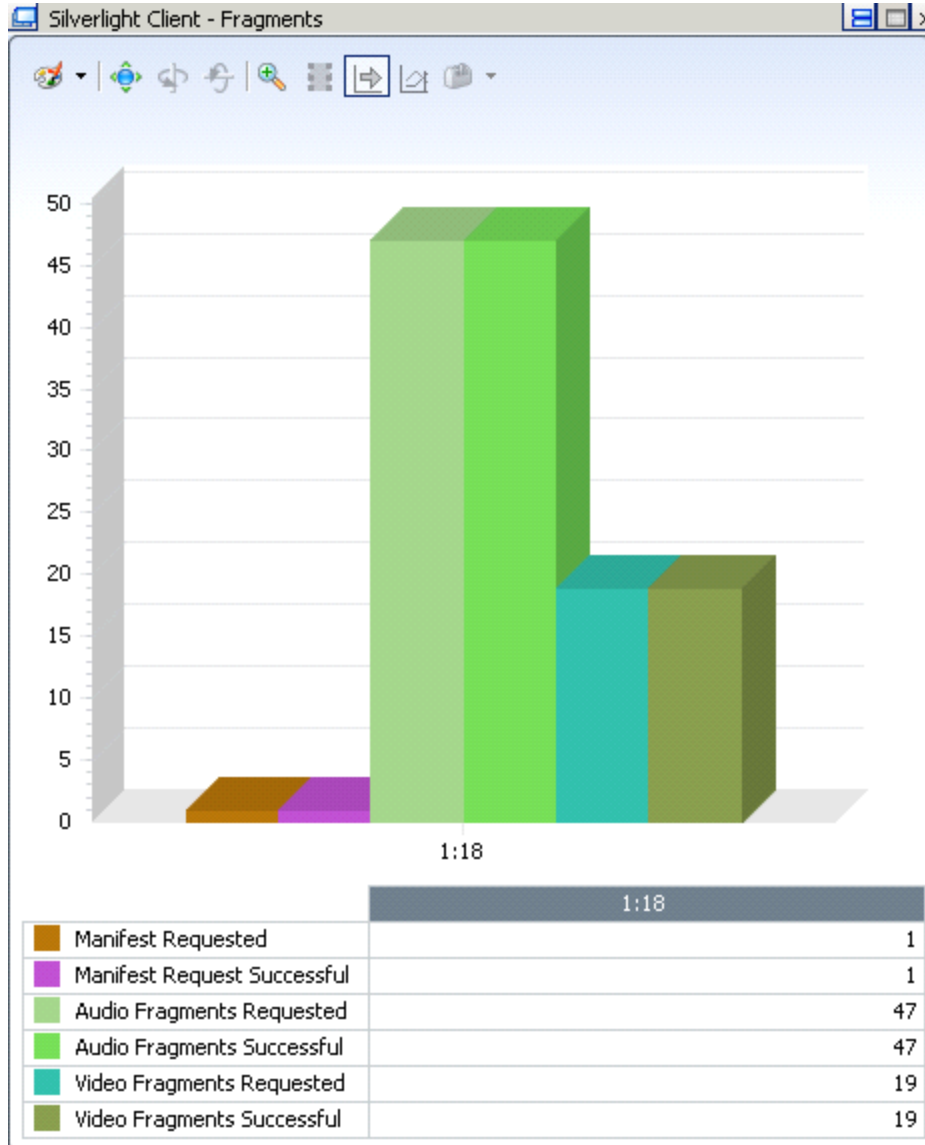


Figure 136. Silverlight Client – Fragments view

The **Silverlight Client - Fragments** view shows all the audio, video, and media fragments that were requested and how many failed or succeeded.



## Test Case: Basic Silverlight® Smooth Streaming Client

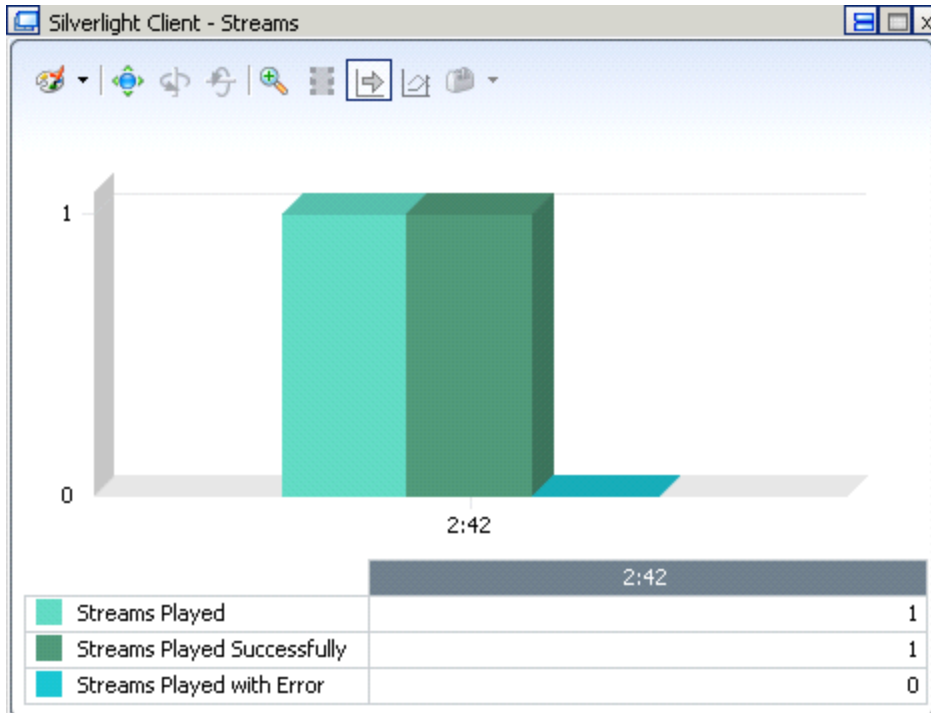


Figure 137. Silverlight Client – Streams view

The **Silverlight Client - Streams** view updates at the end of a stream playback. For example, if the stream duration is 120 seconds, this view will update after 120 elapses. If one or more fragment requests fail, the test will continue to run, but the 'Streams Played with Error' will indicate such transient and important error conditions.

## Troubleshooting and Diagnostics

Issue	Diagnosis, Suggestions
No SYNs sent.	Test port cannot resolve gateway or server IP to MAC. Resolve routing or switch issue. Are VLANs required?
Manifest request fails.	Is proxy configured? Is it required? Is the media URL correct?  Ensure that the '/Manifest' is not in the URL.
Partial fragment requests are failing.	The network may be dropping packets, or server may have congestion and causing issues, or the manifest file is not correctly constructed, or media files are missing from the server.

**Table 41.** Troubleshooting checklist for Silverlight Player playback



## Test Case: Silverlight® Smooth Streaming with Pre-selected Bitrate Shifts

### Overview

Silverlight's most prevalent adoption comes from delivering high quality media, called Smooth Streaming. Specifically, Windows 2008 Server with IIS Smooth Streaming enables adaptive streaming of on-demand and Live streaming video by using HTTP.

Smooth Streaming allows the Silverlight Player to dynamically adapt to the nature of the transport network by changing the quality of the video with no user input. For example, if the playback starts with average quality, and there is sufficient bandwidth, the Silverlight Player automatically up shifts to a stream of higher quality. If adverse network conditions exist, such as limited bandwidth or packet drops, the Silverlight Player automatically down shifts to a lower acceptable quality. The heuristics of determining network conditions and when to shift up or down is built into the Silverlight Player. A video server that is compatible with this technology is required, such as Windows 2008 IIS7 with Smooth Streaming extensions.

IxLoad includes the Silverlight Player video smooth streaming capability. It supports playback of on-demand and Live media, and it supports Smooth Streaming to switch to streams of higher or lower quality.

### Objective

Create a test profile in IxLoad to act as a Silverlight Player to stream content and use adaptive bitrate shifts to emulate different users up shifting or down shifting to different levels of media quality during playback.

The test case will outline how to set up the Silverlight player in IxLoad to use profiles to create a scenario to up shift and down shift to different video streams during playback. Profiles are used to create percentage of users taking different shift actions. The shift (up or down) is created using a bit-rate shift table that is deterministic and repeatable.

## Setup

The following setup is used:

One Ixia test port is connected directly to a Silverlight Streaming Server (Windows 2008 running IIS7 with extensions). The topology shown in the following figure is representative of a more complex scenario with optional HTTP Proxy.

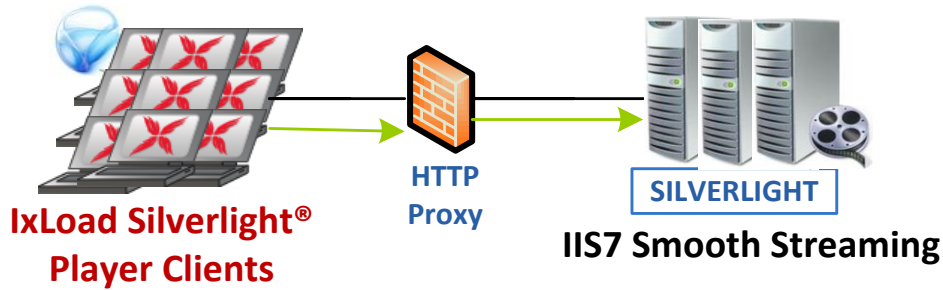


Figure 138. Silverlight Player setup to test Silverlight Compatible Server infrastructure

## Test Variables

### Test Tool Variables

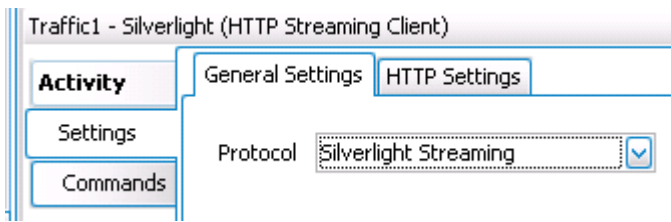
Parameters	Description
Silverlight Clients	100 IP addresses or more, use sequential or 'use all' IP addresses.
Silverlight parameters	Change 'Protocol' in the <b>General</b> tab to 'Silverlight Streaming.' 
TCP parameters	TCP RX and TX buffer at 32768 bytes.
Silverlight Player client command list	Use PLAY and configure based on the specific server and content name that is present on the server. See example in Step by Step.
Play Duration	Choose the Use Profiles option. This option is used to create a percentage based distribution

Figure 139. – Choosing Silverlight Player

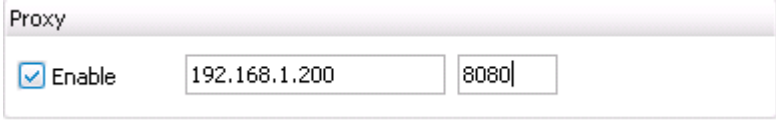
Parameters	Description
	and uses a shift table to create up-shift and down-shift, number of levels, and duration for playback for each shift. See details in Step by step.
Buffer configuration	By default, 'Infinite' is used. All the segment 'GET' requests are sent at the beginning of the stream playback.  With 'Finite,' the buffer size (in seconds) can be configured. With a finite buffer size, the segment 'GET' requests are timed to fill the buffer and sent continuously.
Proxy support	If clients connect through a content switch or proxy, IxLoad can send all HTTP packets to this configured <b>IP:port</b> .   <p style="text-align: center;">Figure 140. Setting HTTP proxy for Silverlight Player</p>

Table 42. Test Tool Variables

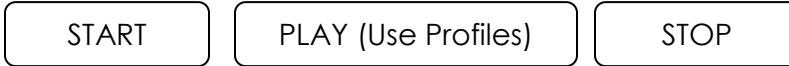
### DUT Test Variables

Device(s)	Variation	Description
Silverlight Compatible Server	Enable on-demand and smooth streaming	Each server may use a different semantic to reach the Manifest or the adaptive streaming Manifest. Check and confirm that it works with a real Silverlight Player.

Table 43. DUT Variables

### Step-by-Step Instructions

1. Start IxLoad. In the main window, the **Scenario Editor** window appears. All test configurations will be done here.
2. To get familiar with the IxLoad GUI, see the Getting Started Guide section.
3. Add the client NetTraffic object. Configure the client network with total IP count, gateway, and VLAN, if used.
4. For a step-by-step workflow, see Annex A.
5. Configure the Silverlight Player client. Add the **HTTP Streaming** activity to the client NetTraffic.
6. For **Smooth Streaming PLAYBACK**, configure the command list as follows:



7. Add the **PLAY** command and configure the following parameters:

- Choose Silverlight Streaming for Protocol from the **General** tab.
- Enter the destination server by IP or hostname (enable DNS to use this). If the server is listening on a non-standard port, it can be specified as IP:port.
- Enter the media URL. The media name must follow the exact convention that the Silverlight server requires. Each implementation can vary. Do not specify the '/Manifest' in the end of the media URL. It is always automatically added. For example, '/all/mp4:media.m4v' would be sent as '/all/mp4:media.m4v/Manifest.'
- Choose 'Use Profiles' as the play duration. This option allows multiple groups of users to up-shift or down-shift to different media quality.

Figure 141. Configuring PLAY command to use Profiles for bit-rate shifting

8. Go to the **Settings** tab to set up profiles of users and their up-shift and down-shift capabilities. Here, you can also set percentage of profiles across users.
9. Add a new profile in the Stream Viewing Profile.

Stream Viewing Profiles	
Percentage Of Users	Profile Name
50	

Figure 142. Stream viewing profile for bit-rate shift profiles

The 'Edit Viewing Profile' window appears. This table allows you to configure up shift, down shift, jump to lowest (default) bitrate, and jump to highest bitrate. For each shift, the user can set how many levels to shift up or down, and for how long to watch a particular stream after shifting to it. The 'Stay at Current Bit Rate' must always be first, with any desired duration.

In this scenario, the server contains four levels of quality for the playback of Big Buck Bunny video (450 kbps, 750 kbps, 1100 kbps, and 1500 kbps).

Test Case: Silverlight® Smooth Streaming with Pre-selected Bitrate Shifts

Name	Size
bigbuckbunny.smil	1 KB
bigbuckbunny_450.mp4	35,192 KB
bigbuckbunny_750.mp4	57,021 KB
bigbuckbunny_1100.mp4	82,462 KB
bigbuckbunny_1500.mp4	111,537 KB

Figure 143. View of content that is available for playback on the Media Server

The following Viewing Profile shifts up three levels, and down shifts to default.

A Play Duration of 0 means 'Play until end,' which is automatically determined from the Manifest.

**Edit Viewing Profile**

Profile Name:

Bit Rate Shift Table		
Shift Type	Number Of Levels To Shift	Play Duration
Stay At Current Bit Rate	1	30
Change To Higher Bit Rate	1	30
Change To Higher Bit Rate	1	30
Change To Higher Bit Rate	1	30
Stay At Current Bit Rate	1	0

Figure 144. Configuring an adaptive viewing profile

10. Set the playback buffer to 30 seconds.

Enable Buffer

Infinite  
 Finite  seconds

Figure 145. Playback buffer configuration



11. IxLoad includes a powerful 'User Monitoring' capability, which provides an easy way to visualize how a 'monitored' user requests to shift to different streams over time. Enable it by going to the **Settings** tab.



The image shows a dialog box titled "Enable User Monitoring". Inside the dialog, there is a checked checkbox labeled "Enable" and a text input field containing the number "0".

**Figure 146.** Configuration of the "User Monitoring" feature to track bitrate shifting

Note that user index 0 is the first user. If a test has 99 users, the index value is 98.

12. Having set up the HTTP Streaming (Silverlight) activity, set test objective. Configure simulated user as the test objective and set the objective value to the desired number of users.
13. To ensure that no interoperability or configuration issues exist (such as wrong content name), you can run a one simulated user test and enable capture on the port to see the packet exchange for possible issues.
14. Add test port resources and run the test. Refer to the Results Analysis and the Troubleshooting and Diagnostics section for further information.

## Results Analysis

Metric	Key Performance Indicators	Statistics View
Performance Metrics	Streaming Users, Played, Paused RX/TX Throughput, Audio/Video/Media Throughput	Silverlight Client - Active Streams Silverlight Client - Throughput Silverlight Client - Streams
Application Level Transactions	Streams Played, Successful, Played with Error	Silverlight Client - Fragments Silverlight Client - Failures
Application Level Failure Monitoring	Manifest Requested, Successful Audio/Video/Media Fragments Requested, Successful	
Smooth Streaming Shifting	Total Upshifts, Total Downshifts	Silverlight Client - Bitrate Shift
TCP Statistics	TCP SYN, SYN-ACK, Connections Established, Failed	TCP Stats, HTTP Failures

Table 44. Results Analysis

## Real-time Statistics

The HTTP Streaming activity supports real-time statistics for TCP, HTTP, and the Silverlight application.

The **Silverlight Client - User Monitoring** view shows how the monitored user is up-shifting to different quality streams.



Figure 147. Silverlight Streaming User Monitoring view

Test Case: Silverlight® Smooth Streaming with Pre-selected Bitrate Shifts

The **Silverlight Client - Bitrate Shift** view indicates in real-time all attempted up shifts and down shifts.

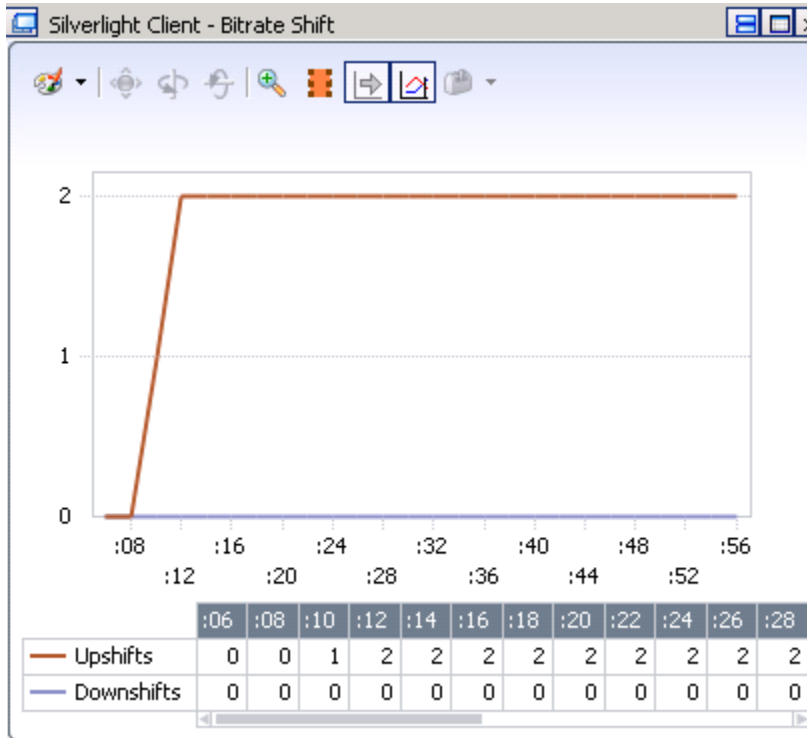


Figure 148. Silverlight Client – Bitrate Shift view

For other performance indicators, refer to the 'Basic Silverlight Streaming Client' test case.



## Test Case: Basic HTTP Live Streaming (HLS) Client

### Overview

HTTP Live Streaming (HLS) is a media streaming specification, developed by Apple® Inc that uses HTTP as the transport. Devices such as iPhone, iPad, and Apple compatible platforms support this streaming technology. The 'Live' is misleading in the name, as this technology works for on-demand and Live streaming

HLS supports streaming media that is segmented into smaller chunks of data, to improve delivery and user experience. An Extended M3U Playlist format file is used that contains the media segments to download.

The HLS specification is documented in the following draft RFC: (draft-pantos-http-live-streaming-04, <http://tools.ietf.org/html/draft-pantos-http-live-streaming-04>).

IxLoad includes HLS Player emulation capability. It supports playback of on-demand and Live media.

### Objective

Create a test profile in IxLoad to act as a Apple HLS Player to stream content from a Apple HLS Compatible Video Server.

### Setup

The following setup is used: One Ixia test port is connected directly to an Apple HLS Compatible Streaming Server.

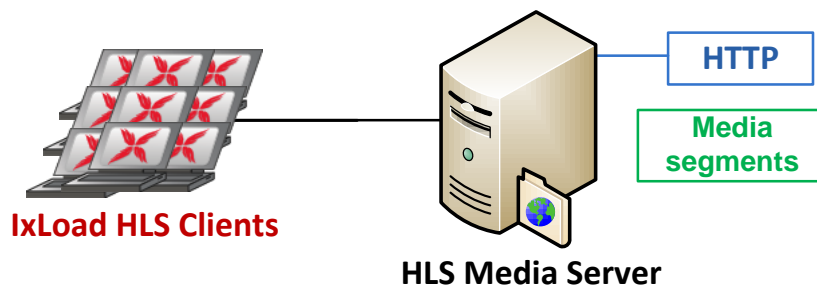


Figure 149. HLS Player setup to test HLS Compatible Server infrastructure

## Test Variables

### Test Tool Variables

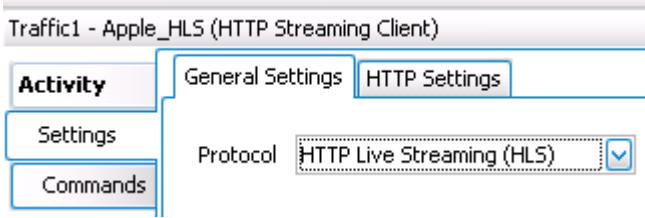

Parameters	Description
HLS Clients	100 IP addresses or more, use sequential or 'use all' IP addresses.
HLS parameters	<p>Change 'Protocol' in the <b>General</b> tab to 'HTTP Live Streaming (HLS).'</p>  <p style="text-align: center;"><b>Figure 150. Setting the HLS Player capability</b></p>
TCP parameters	TCP RX and TX buffer at 32768 bytes.
HLS Player client command list	Configure PLAY based on the specific server and content name present on the server. See example in Step by Step.
Buffer configuration	<p>By default, 'Infinite' is used. All the segment 'GET' requests are sent at the beginning of the stream playback.</p> <p>With 'Finite,' the buffer size (in seconds) can be configured. With a finite buffer size, the segment 'GET' requests are timed to fill the buffer and sent continuously.</p>
Proxy support	<p>If clients connect through a content switch or proxy, IxLoad can send all HTTP packets to this configured <b>IP:port</b>.</p>  <p style="text-align: center;"><b>Figure 151. Enabling HTTP proxy for HLS Player</b></p>
Use of sequence generator	<p>Sequence generator is used above to expand <b>video[1-5].m4v</b> to video1.m4v through video5.m4v.</p> <p>Each user in the test will start with the next index, that is, user1 starts with video1.m4v, user2 starts with video2.m4v, and cycles through the sequence.</p>

Table 45. –Test Tool Variables

DUT Test Variables

Device(s)	Variation	Description
HLS Compatible Server	Enable on-demand streaming	Each server may use a different semantic to reach the playlist file. Check and confirm that it works from an iPhone/iTouch/iPad device.

Table 46. DUT Variables

Step-by-Step Instructions

1. Start IxLoad. In the main window, the **Scenario Editor** window appears. All test configurations will be done here.
2. To get familiar with the IxLoad GUI, see the Getting Started Guide section.
3. Add the client NetTraffic object. Configure the client network with total IP count, gateway, and VLAN, if used.
4. For a step-by-step workflow, see Annex A.
5. Configure the Silverlight Player client. Add the **HTTP Streaming** activity to the client NetTraffic.
6. For basic **PLAYBACK**, configure the command list as follows:



7. Add the **PLAY** command and configure the following parameters:
  - Choose HTTP Live Streaming (HLS) for Protocol from the **General** tab.
  - Enter the destination server by IP or hostname (enable DNS to use this). If the server is listening on a non-standard port, it can be specified as IP:port.
  - Enter the media URL. The media name must follow the exact convention that the HLS server requires. Each implementation can vary. For example, '/all/mp4:media.m4v/playlist.m3u8'
  - Choose 'Play Till End' as the play duration.



## Test Case: Basic HTTP Live Streaming (HLS) Client

- The 'Duration' option is available to specify a range of time. Each user will randomly choose playback duration within this range.

The screenshot shows a configuration window titled "Play". It contains the following elements:

- Server IP Address:** A text field containing "172.0.0.100:1935" with a dropdown arrow on the right.
- Media URL:** A text field containing "/all/mp4:Extremists.m4v/playlist.m3u8".
- Duration (secs):** A range configuration with two input boxes, both containing "10", separated by a hyphen.
- Radio Buttons:** Three radio buttons are present:
  - Play Till End
  - Use Profiles
  - Duration (secs)

**Figure 152. Configuring PLAY command for HLS Player**

8. Having set up the HTTP Streaming [HTTP Live Streaming (HLS)] activity, configure the test objective. Configure simulated user as the test objective and set the objective value to the desired number of users.
9. To ensure that no interoperability or configuration issues exist (such as wrong content name), you can run a one simulated user test and enable capture on the port to see the packet exchange for possible issues.
10. Add test port resources and run the test. Refer to the Results Analysis and the Troubleshooting and Diagnostics section for further information.

## Results Analysis

Metric	Key Performance Indicators	Statistics View
Performance Metrics	Streaming Users, Played, Paused RX/TX Throughput, Audio/Video/Media Throughput	HLS Client - Active Streams HLS Client - Throughput HLS Client - Streams
Application Level Transactions  Application Level Failure Monitoring	Streams Played, Successful, Played with Error Manifest Requested, Successful Audio/Video/Media Fragments Requested, Successful	HLS Client - Streams HLS Client - Failures
Playlist Requests	Primary, Secondary, and Dynamic Play Requests Requested, Successful, Failed	HLS Client - Playlist
TCP Statistics	TCP SYN, SYN-ACK, Connections Established, Failed	TCP Stats, TCP Failures

Table 47. Results Analysis for HLS Player

## Real-time Statistics

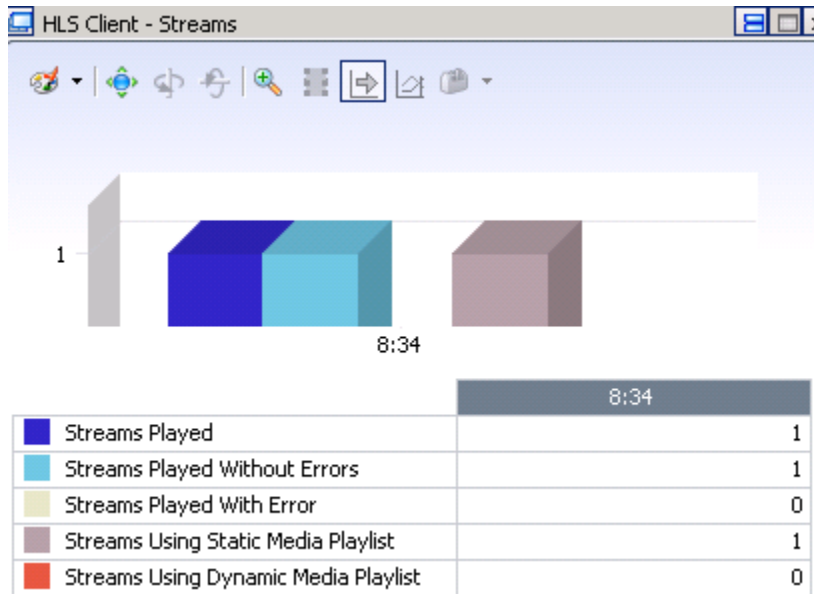


Figure 153. **HLS Client – Streams view**

The **HLS Client - Streams** view contains statistics for stream playlist with and without errors. It also indicates the total streams that played using static playlist (on-demand) and dynamic playlist (Live).

## Test Case: Basic HTTP Live Streaming (HLS) Client

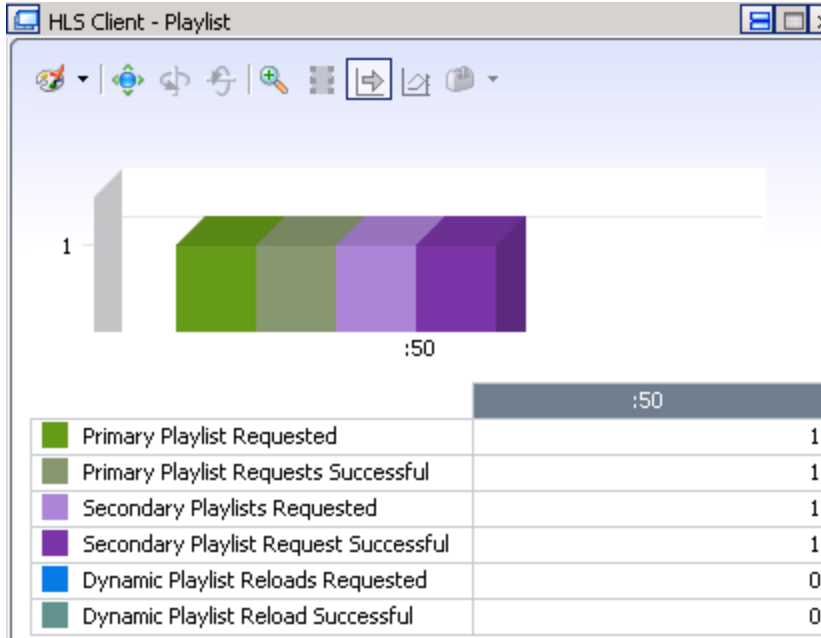


Figure 154. HLS Client – Playlist view

The **HLS Client - Playlist** view contains a number of different playlist types requested, successful, and failed. If there are issues with the filename or any other playlist related errors, this view can be used to identify them.

- Primary Playlist: Is used for static loading of on-demand content.
- Secondary Playlist: A primary playlist can refer to this playlist to locate content.
- Dynamic Playlist: Is used for Live streaming, and server sends this playlist asynchronously to the client, which processes this to know next segments to request.

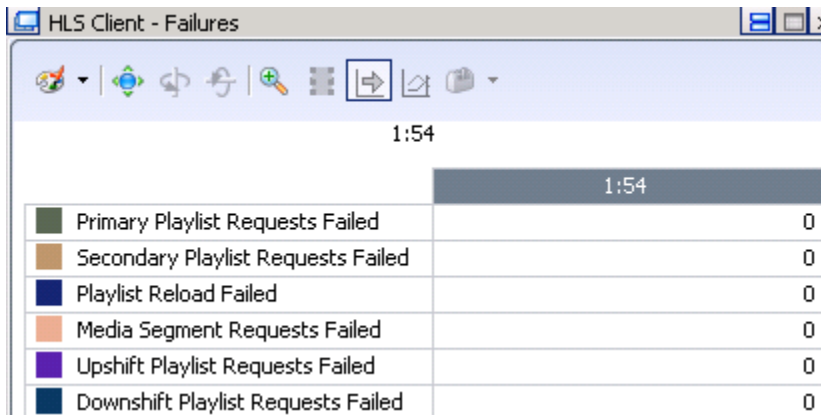


Figure 155. HLS Client – Failures view

The **HLS Client - Failures** view indicates any issues related to playlist requests or media segment requests failing.

## Troubleshooting and Diagnostics

Issue	Diagnosis, Suggestions
No SYNs sent.	Test port cannot resolve gateway or server IP to MAC. Resolve routing or switch issue. Are VLANs required?
Playlist request fails.	<p>Is proxy configured? Is it required? Is the media URL correct?</p> <p>Ensure the '/Playlist.m3u8' is included in the URL.</p>
Partial media requests are failing.	The network may be dropping packets, or server may have congestion and causing issues, or the Playlist.m3u8 file is not correctly constructed, or media files are missing from the server.

**Table 48.** Troubleshooting checklist for HLS Player playback

## Test Case: HTTP Live Streaming with Pre-Selected Adaptive Bitrate Shifting

### Overview

HTTP Live Streaming (HLS) is a media streaming specification, developed by Apple® Inc that uses HTTP as the transport. Devices such as iPhone, iPad, and Apple compatible platforms support this streaming technology. The 'Live' is misleading in the name, as this technology works for on-demand and Live streaming.

HLS supports streaming media that is segmented into smaller chunks of data, to improve delivery and user experience. An Extended M3U Playlist format file is used that contains the media segments to download.

HLS supports media streaming with different bitrates and quality, and the HLS client observes network conditions to automatically up shift or down shift to appropriate quality of media stream.

The HLS specification is documented in the following draft RFC: (draft-pantos-http-live-streaming-04, <http://tools.ietf.org/html/draft-pantos-http-live-streaming-04>).

IxLoad includes the HLS Player emulation capability. It supports playback of on-demand and Live media and adaptive bitrate shifts to switch to streams of higher or lower quality.

### Objective

Create a test profile in IxLoad to act as an Apple HLS Player to stream content and use adaptive bitrate shifts to emulate different users up shifting or down shifting to different levels of media quality during playback.

The test case will outline how to set up the Apple HLS player in IxLoad to use profiles to create a scenario to up shift and down shift to different video streams during playback. Profiles are used to create percentage of users taking different shift actions. The shift (up or down) is created using a bit-rate shift table that is deterministic and repeatable.

## Setup

The following setup is used:

One Ixia test port is connected directly to an Apple HLS Compatible Streaming Server.

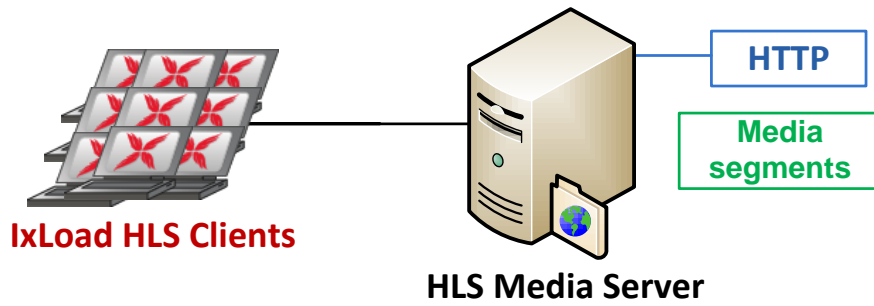


Figure 156. HLS Player setup to test HLS Compatible Server infrastructure

## Test Variables

### Test Tool Variables

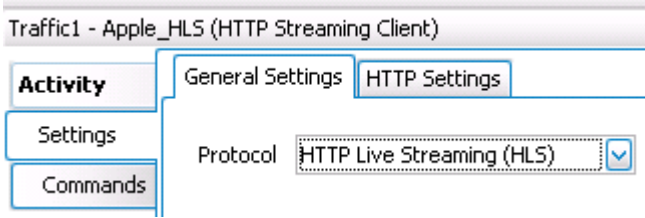

Parameters	Description
HLS Clients	100 IP addresses or more, use sequential or 'use all' IP addresses.
HLS parameters	<p>Change 'Protocol' in the <b>General</b> tab to 'HTTP Live Streaming (HLS).'</p>  <p style="text-align: center;"><b>Figure 157. Setting HLS Player emulation</b></p>
TCP parameters	TCP RX and TX buffer at 32768 bytes.
HLS Player client command list	Configure PLAY based on the specific server and content name present on the server. See example in Step by Step.
Buffer configuration	<p>By default, 'Infinite' is used. All the segment 'GET' requests are sent at the beginning of the stream playback.</p> <p>With 'Finite,' the buffer size (in seconds) can be configured. With a finite buffer size, the segment 'GET' requests are timed to fill the buffer and sent continuously.</p>
Proxy support	<p>If clients connect through a content switch or proxy, IxLoad can send all HTTP packets to this configured <b>IP:port</b>.</p>  <p style="text-align: center;"><b>Figure 158. HTTP proxy setting for HLS playback</b></p>
Use of sequence generator	<p>Sequence generator is used above to expand <b>video[1-5].m4v</b> to video1.m4v through video5.m4v.</p> <p>Each user in the test will start with the next index, that is, user1 starts with video1.m4v, user2 starts with video2.m4v, and cycles through the sequence.</p>



Table 49. Test tool variables

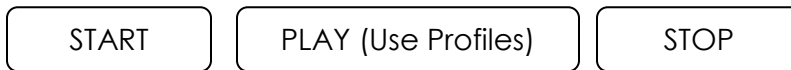
DUT Test Variables

Device(s)	Variation	Description
HLS Compatible Server	Enable on-demand streaming	Each server may use a different semantic to reach the playlist file. Check and confirm that it works from an iPhone/iTouch/iPad device.

Table 50. DUT Variables

Step-by-Step Instructions

1. Start IxLoad. In the main window, the **Scenario Editor** window appears. All test configurations will be done here.
2. To get familiar with the IxLoad GUI, see the Getting Started Guide section.
3. Add the client NetTraffic object. Configure the client network with total IP count, gateway, and VLAN, if used.
4. For a step-by-step workflow, see Annex A.
5. Configure the HTTP Live Streaming (HLS) client. Add the **HTTP Streaming** activity to the client NetTraffic.
6. For adaptive streaming **PLAYBACK**, configure the command list as follows:



7. Add the **PLAY** command and configure the following parameters:
  - Choose Silverlight Streaming for Protocol from the **General** tab.
  - Enter the destination server by IP or hostname (enable DNS to use this). If the server is listening on a non-standard port, it can be specified as IP:port.
  - Enter the media URL. The media name must follow the exact convention that the Silverlight server requires. Each implementation can vary. Do not specify the '/Manifest' in the end of the media URL. It is always automatically added. For example, '/all/mp4:media.m4v' would be sent as '/all/mp4:media.m4v/playlist.m3u8'
  - Choose 'Use Profiles' as the play duration. This option allows multiple groups of users to up-shift or down-shift to different media quality.

## Test Case: HTTP Live Streaming with Pre-Selected Adaptive Bitrate Shifting

Play

Server IP Address 172.0.0.100:1935

Media URL /all/smil:bigbuckbunny.smil/playlist.m3u8

Duration (secs) 10 - 10

Play Till End

Use Profiles

Figure 159. PLAY command with Profiles for bit-rate shifting playback

8. Go to the **Settings** tab to set up profiles of users and their up-shift and down-shift capabilities. Here, you can also set percentage of profiles across users.
9. Add a new profile in the Stream Viewing Profile.

Percentage Of Users	Profile Name
50	

Figure 160. HLS Stream Viewing Profiles

The **Edit Viewing Profile** window appears. This table allows you to configure up shift, down shift, jump to lowest (default) bitrate, and jump to highest bitrate. For each shift, the user can set how many levels to shift up or down, and for how long to watch a particular stream after shifting to it. The 'Stay at Current Bit Rate' must always be first, with any desired duration. In this scenario, the server contains four levels of quality for the playback of Big Buck Bunny video (450 kbps, 750 kbps, 1100, kbps, and 1500 kbps).

Name	Size
bigbuckbunny.smil	1 KB
bigbuckbunny_450.mp4	35,192 KB
bigbuckbunny_750.mp4	57,021 KB
bigbuckbunny_1100.mp4	82,462 KB
bigbuckbunny_1500.mp4	111,537 KB

Figure 161. View of content that is available for playback on the Media Server

The following Viewing Profile up shifts three levels and down shifts to default. A play duration of 0 means 'Play until end,' which is automatically determined from the manifest.

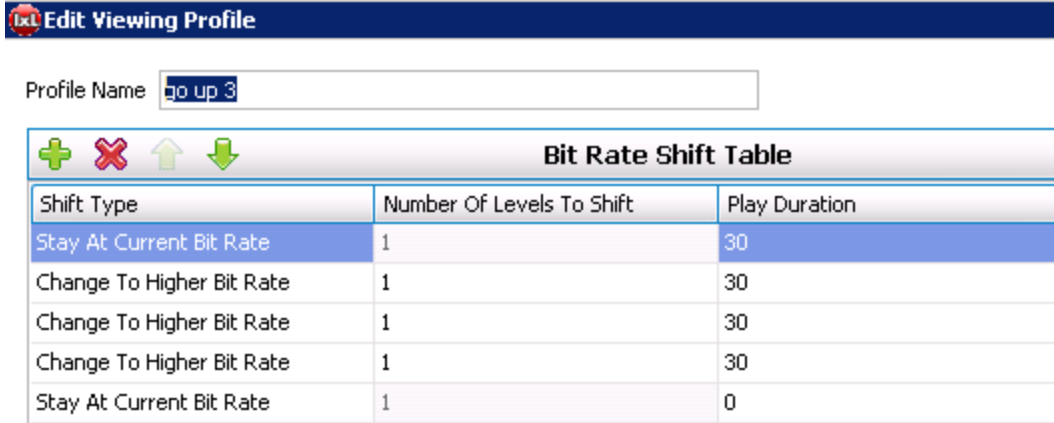


Figure 162. Configuring bit-rate shift profile

10. Set the playback buffer to 30 seconds.



Figure 163. Playback buffer configuration

11. IxLoad includes a powerful 'User Monitoring' capability, which provides an easy way to visualize how a 'monitored' user requests to shift to different streams over time. Enable it by going to the **Settings** tab, and enabling it as follows:



Figure 164. Configuring User Monitoring

Note that user index 0 is the first user. If a test has 99 users, the index is 98.

12. Having set up the HTTP Streaming (Silverlight) activity, configure the test objective. Configure simulated user as the test objective and set the objective value to the desired number of users.
13. To ensure that no interoperability or configuration issues exist (such as wrong content name), you can run a one simulated user test and enable capture on the port to see the packet exchange for possible issues.
14. Add test port resources and run the test. Refer to the Results Analysis and the Troubleshooting and Diagnostics section for further information.

## Results Analysis

Metric	Key Performance Indicators	Statistics View
--------	----------------------------	-----------------

Test Case: HTTP Live Streaming with Pre-Selected Adaptive Bitrate Shifting

Performance Metrics	Streaming Users, Played, Paused RX/TX Throughput, Audio/Video/Media Throughput	HLS Client - Active Streams HLS Client - Throughput HLS Client - Streams
Application Level Transactions  Application Level Failure Monitoring	Streams Played, Successful, Played with Error Manifest Requested, Successful Audio/Video/Media Fragments Requested, Successful	HLS Client - Streams HLS Client - Failures
Playlist Requests	Primary, Secondary, and Dynamic Play Requests Requested, Successful, Failed	HLS Client - Playlist
Media Segments	Media Segments Requested, Successful, Failed	HLS Client - Media Segments
TCP Statistics	TCP SYN, SYN-ACK, Connections Established, Failed	TCP Stats, TCP Failures

Table 51. Results Analysis for Silverlight Player playback

## Real-time Statistics

The HTTP Streaming activity supports real-time statistics for TCP, HTTP, and the HLS application.

The **HLS Client - User Monitoring** view shows how the monitored user is up-shifting to higher quality streams.



Figure 165. HLS Client - User Monitoring

For other performance indicators, refer to the 'Basic HLS Client' test case.

## Annex A: Configuring IP and Network Settings

In the Scenario Editor, add a NetTraffic Object. This object can contain network configurations and activities (protocols).



Figure 166. New NetTraffic Object

Click Network1 to configure the IP, TCP, and other network configuration parameters.

On the IP stack, configure the desired number of static IP addresses. If VLANs are required, configure it by selecting the MAC/VLAN stack and configuring the details.

Stack-1										
<div style="border: 1px solid gray; padding: 5px;"> <div style="border: 1px solid gray; padding: 2px; margin-bottom: 2px;">  IP-1                 </div> <div style="border: 1px solid gray; padding: 2px;">  MAC/VLAN-1                 </div> </div>										
	Enabled	Name	Status	IP Type	Address	Mask	Increment	Count	Gateway	
▶ 1	<input checked="" type="checkbox"/>	IP-R4	Unconfigured	IPv4	192.168.1.2	16 0.0.0.1		1000	192.168.1.1	

Figure 167. - IP stack

VLAN							
	Enabled	Name	Status	First ID	Increment every # addresses	Increment By	Unique Count
▶ 1	<input checked="" type="checkbox"/>	VLAN-R4	Unconfigured	101	1	1	1000

Figure 168. VLAN settings



## Annex B: Configuring TCP Parameters

The TCP settings shown should be configured as per the test tool Input Parameters for the specific test case.

Select the NetTraffic for the TCP configurations. Select the network object to open the Stack Manager window on the bottom.

Click **TCP/IP**. Configure the Receive and Transmit Buffer Size based on what is set in the Input Parameters for TCP/IP settings.

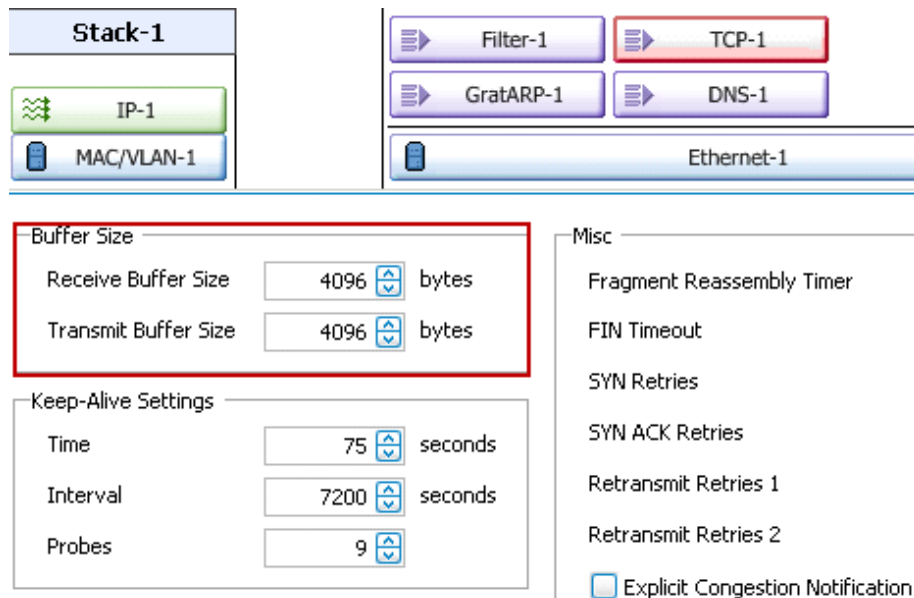


Figure 169. - Buffer size settings

Other TCP/IP configurations should not be changed from their defaults.





## Annex C: Configuring HTTP Servers

Add the HTTP server activity to the server NetTraffic object.

To configure the HTTP server parameters, pages, and responses, select the HTTPServer1 object to open the configuration pane on the bottom.

Networks and Traffic - New Traffic Flow

Originate Terminate

Network1 Traffic1

Network2 Traffic2 HTTPServer1

Traffic2\_HTTPServer1

HTTP | SSL | Web Pages | Advanced Options

Enable Integrity Check

Docroot Settings

Docroot File :

Integrity Check Option:

HTTP Server Web Pages

Examples...

	Page	Response	Payload Type	Fixed Size or From-To (Bytes)
1	/1b.html	200_OK	Range	1-1

Figure 170. HTTP server configuration

Configure the HTTP server as outlined in the Input Parameters section.



## Annex D: Configuring HTTP Clients

Add the HTTP client activity to the client NetTraffic object.

To configure the HTTP parameters, and pages to request, select the 'HTTPClient1' object to open the configuration pane on the bottom.

Configure the HTTP behavior on the **HTTP** tab. Refer to the Input Parameters section.

The version of HTTP to use, TCP connections per user, and transactions per TCP are configured here.

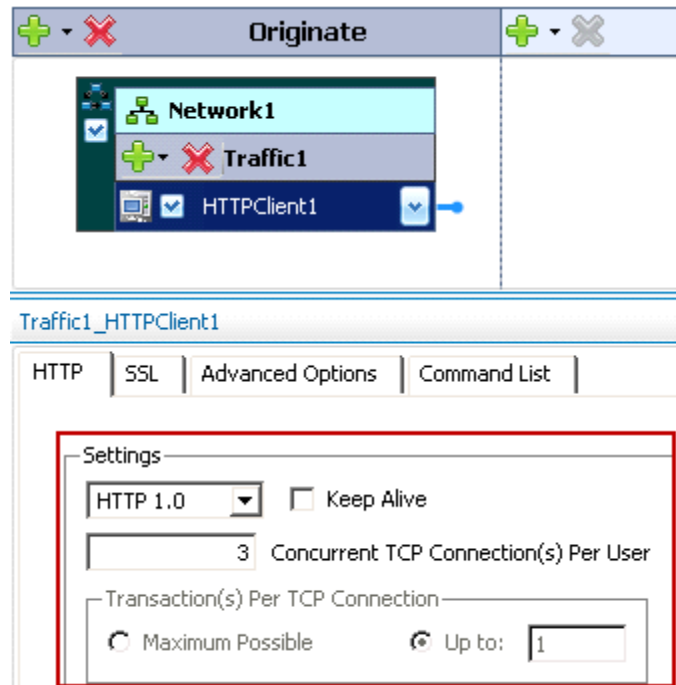


Figure 171. HTTP client configuration

Go to the Command List to configure the list of HTTP commands to use. The specific commands that should be used for the specific test objective type are outlined in the Input Parameters.

## Annex D: Configuring HTTP Clients

For example, when testing for CPS, the page size is 1b.html.

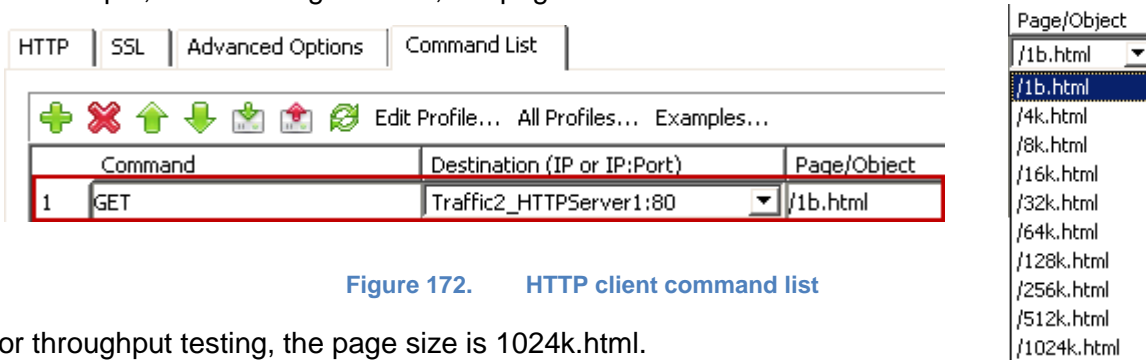


Figure 172. HTTP client command list

For throughput testing, the page size is 1024k.html.

Note the Traffic2\_HTTPServer1:80 in the Destination field. IxLoad supports this symbolic destination to associate traffic to the server object. It allows dynamic configuration of traffic distribution from the clients to servers across several ports, without manual configuration.

The use of source IP addresses for a test can depend on the test requirements. To maximize the test tool's performance, this is set to the default configuration of Use Consecutive IPs. This means that every simulated user will use the IPs as needed from the network configuration.

To change it, click the Traffic1 object and change the Use Source IP Rule (per port) setting.

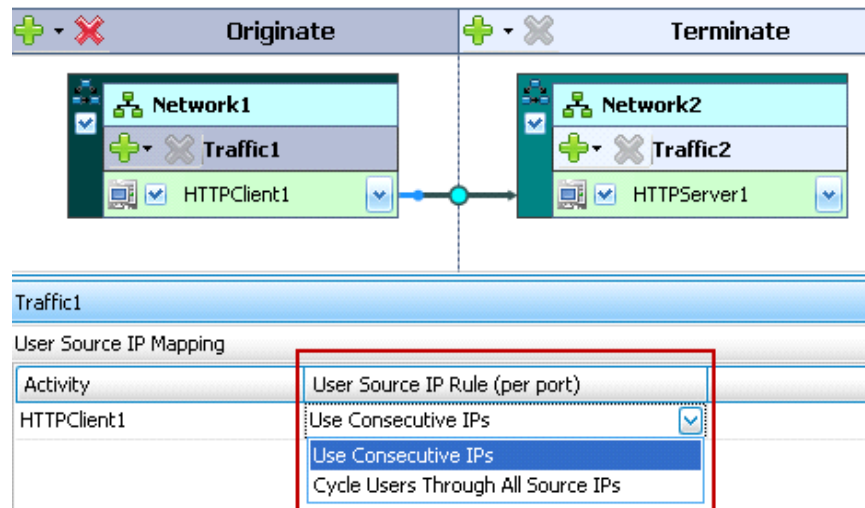


Figure 173. User Source IP Rule (per port)

Using Cycle Users Through All Source IPs allows all IP addresses to be used. This may be desirable, however, note that performance from the test tool may vary. Consider running a baseline test port-to-port to determine the test tool's limit before performing a test with this feature.

## Annex E: Setting the Test Load Profile and Objective

Open the Timeline and Objective window from the Test Configuration left pane.

Each NetTraffic will be listed, with one or more activities under it. Set the Objective Value to the desirable value, based on what the target performance is.

Network Traffic Mapping	Objective Type	Objective Value	Timeline
New Traffic Flow			
Traffic1@Network1	Simulated Users	10000	Timeline1
HTTPClient1	Simulated Users	10000	Timeline1
Traffic2@Network2	N/A	N/A	<Match Longest>
HTTPServer1	N/A	N/A	<Match Longest>

Figure 174. Setting the objective value

Set the ramp up and the overall test duration. Use the Input Parameters section.

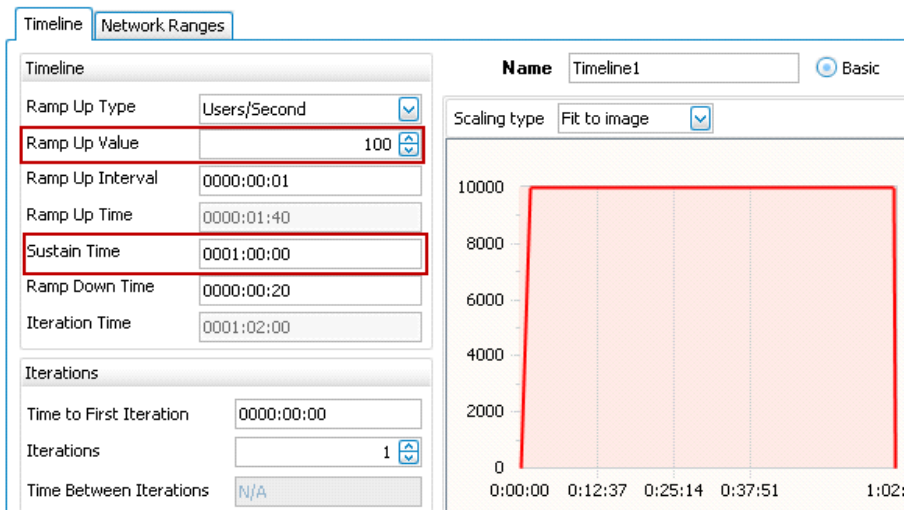


Figure 175. Setting the ramp up and overall test duration

The Number of Ports required highlights the ports required to accomplish the test objective. The computations here are conservative and we recommend you to use Ixia's guidance in addition to the ports required listed here.

Number of Ports Required							
	TXS-128MB	10G-LM	10G-LSM	ELM-1GB	CPM	XMV	ASM XMV12X
<a href="#">Refresh</a>	6	3	2	2	2	2	2

Figure 176. Number of ports required



## Annex F: Adding Test Ports and Running Tests

Go to the **Port Assignments** window, and add the chassis that will be used.

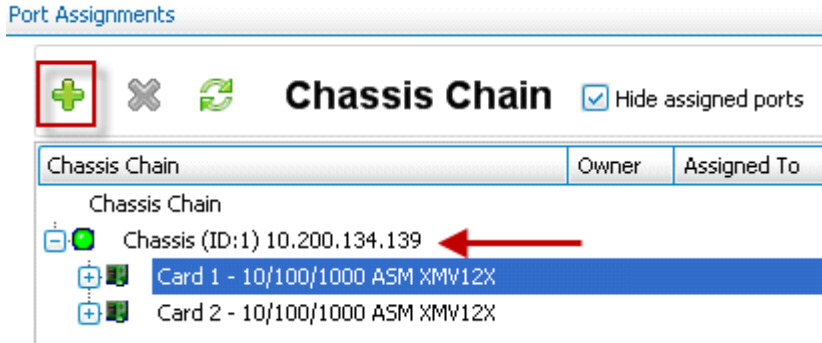


Figure 177. - Adding a chassis

The test ports are assigned at the NetTraffic level. In the simplest case in which HTTP client and server traffic is being emulated, there will be two NetTraffics. Add the required number of ports to each NetTraffic object. Use the arrows to add or remove the available ports to the Assigned Ports.

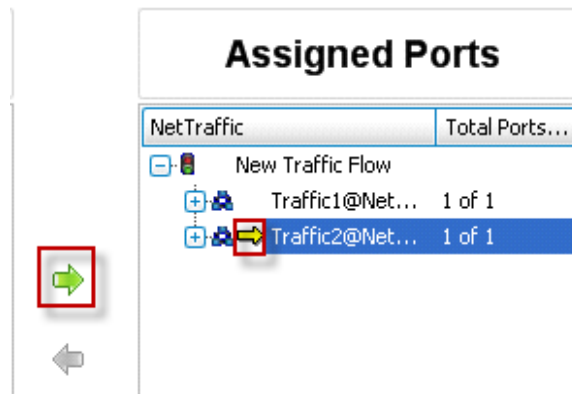





Figure 178. Assigning ports

At this point, the test is ready to be executed.



Figure 179. Executing a test

The PLAY  button starts the test. The RED  button stops the test. The DOWN  arrow downloads the configuration to the test ports, but does not initiate any traffic. The UP arrow deconfigures the test ports.





## Contact Ixia

Corporate Headquarters  
Ixia Worldwide Headquarters  
26601 W. Agoura Rd.  
Calabasas, CA 91302  
USA  
+1 877 FOR IXIA (877 367 4942)  
+1 818 871 1800 (International)  
(FAX) +1 818 871 1805  
[sales@ixiacom.com](mailto:sales@ixiacom.com)

Web site: [www.ixiacom.com](http://www.ixiacom.com)  
General: [info@ixiacom.com](mailto:info@ixiacom.com)  
Investor Relations: [ir@ixiacom.com](mailto:ir@ixiacom.com)  
Training: [training@ixiacom.com](mailto:training@ixiacom.com)  
Support: [support@ixiacom.com](mailto:support@ixiacom.com)  
+1 877 367 4942  
+1 818 871 1800 Option 1 (outside USA)  
online support form:  
<http://www.ixiacom.com/support/inquiry/>

EMEA  
Ixia Technologies Europe Limited  
Clarion House, Norreys Drive  
Maiden Head SL6 4FL  
United Kingdom  
+44 1628 408750  
FAX +44 1628 639916  
VAT No. GB502006125  
[salesemea@ixiacom.com](mailto:salesemea@ixiacom.com)

Renewals: [renewals-emea@ixiacom.com](mailto:renewals-emea@ixiacom.com)  
Support: [support-emea@ixiacom.com](mailto:support-emea@ixiacom.com)  
+44 1628 408750  
online support form:  
<http://www.ixiacom.com/support/inquiry/?location=emea>

Ixia Asia Pacific Headquarters  
21 Serangoon North Avenue 5  
#04-01  
Singapore 5584864  
+65.6332.0125  
FAX +65.6332.0127  
[Support-Field-Asia-Pacific@ixiacom.com](mailto:Support-Field-Asia-Pacific@ixiacom.com)

Support: [Support-Field-Asia-Pacific@ixiacom.com](mailto:Support-Field-Asia-Pacific@ixiacom.com)  
+1 818 871 1800 (Option 1)  
online support form:  
<http://www.ixiacom.com/support/inquiry/>