ixia

Triple Play Testing with IxChariot

IxChariot

Contents

IXIA
26601 West Agoura Road, Calabasas, CA 91302
(877) FOR-IXIA

This Test Plan Primer contains a general outline for testing a particular technology. Not all the capabilities of Ixia technology have been exposed in this document. Please feel free to contact us if additional capabilities are required.

1. Introduction

Working at layers 4 through 7 off the OSI stack, IxChariot is an effective tool for evaluating the performance of network equipment that runs the triple-play (Voice, Video and Data) suite of protocols. In this test plan, we'll examine how IxChariot can be used to evaluate the performance of network equipment in a triple-play environment.

1.1 Objectives

We will create several baseline traffic types and measure the performance of a Device Under Test (DUT) when the traffic is run in isolation. Then we'll combine all traffic types and re-assess the performance in terms of throughput, latency and data loss. Finally, we'll adjust QoS parameters on certain traffic flows and implement QoS policies on the DUT to measure its ability to properly prioritize certain streams within a triple-play environment.

2. Internet Baseline Traffic

2.1 Objective and Setup

IxChariot is used to set up a set of background traffic types that will serve as a constant source of background Internet traffic. Traffic types will consist of web accesses, mail, ftp, P2P and various forms of business traffic.

2.2 Methodology

- Create 9 port pairs of IxChariot traffic that simulate Internet traffic. Ensure that the proper port numbers are utilized by the endpoints; for instance, HTTP requests target TCP port 80, POP3 transactions target port 110, etc. See table 1 for the scripts that are to be called out for each pair and the communication settings that are to be used for each pair. Unless noted in this table, all values are default. In all pairs, enter the pair comment as "Internet_Group_routine."
- Ensure that any DUT QoS policy is turned off.
- Run all pairs with a two minute duration and note the results, such as throughput, latency and lost data.
- Save this setup for later reference. File name = baseline.tst.

Script Filename	Protocol	TCP/UDP Port	User Delay	Transaction Delay	Response Delay
DNS.scr	UDP	53		10	10
FTPget.scr	TCP	20	1000	10	1000
FTPput.scr	TCP	20	1000	10	10
HTTP_Secure_Transaction.scr	TCP	443		10	
HTTPgif.scr	TCP	80		10	10
HTTPtext.scr	TCP	80		10	10
NNTP.scr	TCP	119		10	10
POP3.scr	TCP	110		10	
SMTP.scr	TCP	25		10	

Table 1. Internet Baseline Traffic setup

Test Setup | Throughput | Transaction Rate | Response Time | VoIP | One-Way Delay | Lost Data | Jitter | Raw Data Totals | Endpoint Configuration | Datagram

Group	Run Status	Timing... Compl...	Endpoint 1	Endpoint 2	Network Protocol	Service Quality	Script/Stream Filename	Pair Comment	Pair Group Name	Console k Endpoint 1
Pair 53	Finished	18	172.16.8.1	172.17.8.1	UDP		DNS.scr	Internet_Group_routine	No Group	10.0.2.1
Pair 49	Finished	147	172.16.8.1	172.17.8.1	TCP		FTPget.scr	Internet_Group_routine	No Group	10.0.2.1
Pair 55	Finished	183	172.16.8.1	172.17.8.1	TCP		FTPput.scr	Internet_Group_routine	No Group	10.0.2.1
Pair 56	Finished	283	172.16.8.1	172.17.8.1	TCP		HTTPS_Secure_Transactio...	Internet_Group_routine	No Group	10.0.2.1
Pair 50	Finished	59	172.16.8.1	172.17.8.1	TCP		HTTPgif.scr	Internet_Group_routine	No Group	10.0.2.1
Pair 57	Finished	86	172.16.8.1	172.17.8.1	TCP		HTTPtext.scr	Internet_Group_routine	No Group	10.0.2.1
Pair 54	Finished	7	172.16.8.1	172.17.8.1	TCP		NNTP.scr	Internet_Group_routine	No Group	10.0.2.1
Pair 52	Finished	79	172.16.8.1	172.17.8.1	TCP		POP3.scr	Internet_Group_routine	No Group	10.0.2.1
Pair 51	Finished	43	172.16.8.1	172.17.8.1	TCP		SMTP.scr	Internet_Group_routine	No Group	10.0.2.1

Figure 1. Internet Baseline Traffic setup

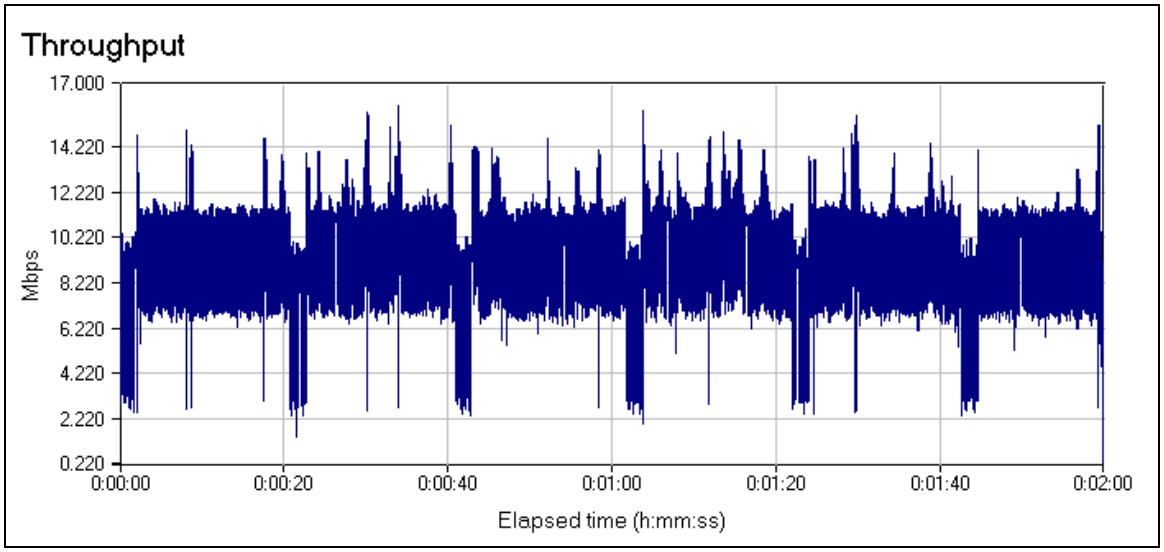

Figure 2. Internet Baseline with no QoS

3. Video Baseline Traffic

3.1 Objective and Setup

IxChariot emulates video streams through use of scripts within the Streaming script file directory. These streams will be used to simulate the behavior of video traffic through the DUT. Delay, jitter and throughput will be measured as part of this test.

3.2 Methodology

- Create a unicast port pair using the IPTVv.scr script. Make sure you specify the UDP network protocol. Retain all default settings for this stream, except change the bandwidth to 1.0Mbps. Enter the pair comment as "Video_routine."
- Duplicate this port pair but switch the source and destination addresses, creating bidirectional traffic.
- Adjust the run time for two minutes.
- Run the pairs with a two minute duration and note the results, such as throughput, latency and missing packets. Save the results for later comparison.
- Save this setup for later reference. File name = video.tst.

Test Setup | Throughput | Transaction Rate | Response Time | VoIP | One-Way Delay | Lost Data | Jitter | Raw Data Totals | Endpoint Configuration | Datagram

Group	Run Status	Timing... Compl...	Endpoint 1	Endpoint 2	Network Protocol	Service Quality	Script/Stream Filename	Pair Comment	Pair Group Name	Console k Endpoint
Pair 117	Finished	42	172.16.8.1	172.17.8.1	UDP		IPTVv.scr	Video_routine	No Group	10.0.2.1
Pair 118	Finished	42	172.17.8.1	172.16.8.1	UDP		IPTVv.scr	Video_routine	No Group	12.0.2.1

Figure 3. Video Baseline Traffic setup

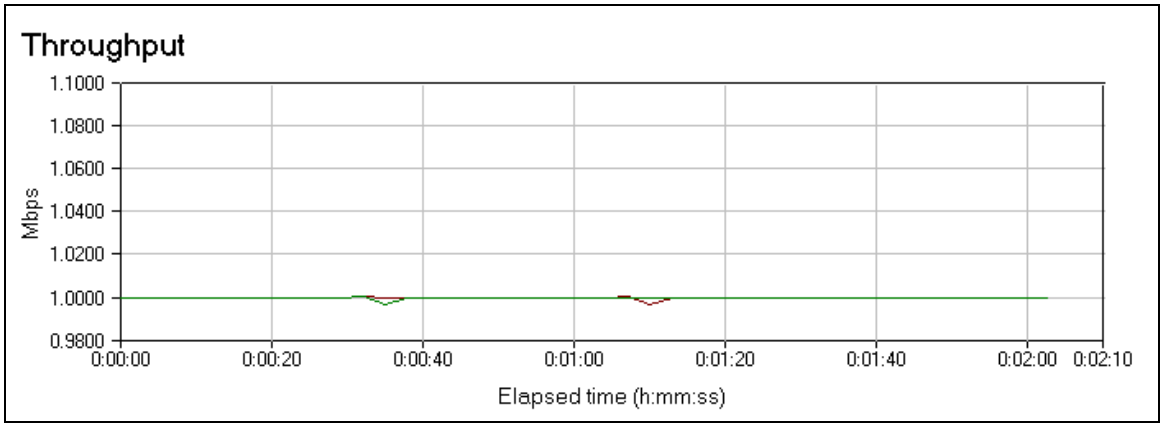

Figure 4. Video Baseline with no QoS

4. VoIP Traffic

4.1 Objective and Setup

IxChariot can emulate voice traffic using several different types of codec algorithms. In this instance, we create bi-directional voice traffic and measure the Mean Opinion Score (MOS) of voice conversations. This allows us to ascertain the quality of the voice calls and thus determine the effectiveness of the network for carrying voice traffic.

4.2 Methodology

- Create six VoIP pairs, with each pair using a unique codec type. (G.711u, G.711a, G.723.1-ACELP, G.723.1-MPMLQ, G.729, G.726) In all pairs, enter the pair comment as "VoIP_routine."
- Replicate the group of VoIP pairs, creating a total of twelve pairs.
- On the replicated pairs, switch the source and destination addresses, thus creating six bidirectional VoIP pairs.
- Replicate all 12 pairs 3 times, creating a total of 48 VoIP pairs, or 24 bidirectional pairs.
- Adjust the run time for two minutes.
- Run this script and collect results. Save the results for later comparison.
- Save this setup for later reference. File name = voip.tst.

Test Setup | Throughput | Transaction Rate | Response Time | VoIP | One-Way Delay | Lost Data | Jitter | Raw Data Totals | Endpoint Configuration | Datagram

Group	Run Status	Timing... Compl...	Endpoint 1	Endpoint 2	Network Protocol	Service Quality	Script/Stream Filename	Pair Comment	Pair Group Name	Con End
Pair 71	Finished	40	172.16.8.1	172.17.8.1	RTP		G.711a	VoIP_routine	No Group	10.0
Pair 72	Finished	40	172.16.8.1	172.17.8.1	RTP		G.711a	VoIP_routine	No Group	12.0
Pair 73	Finished	40	172.16.8.1	172.17.8.1	RTP		G.711a	VoIP_routine	No Group	10.0
Pair 74	Finished	40	172.16.8.1	172.17.8.1	RTP		G.711a	VoIP_routine	No Group	10.0
Pair 75	Finished	40	172.16.8.1	172.17.8.1	RTP		G.711a	VoIP_routine	No Group	10.0
Pair 76	Finished	40	172.16.8.1	172.17.8.1	RTP		G.711a	VoIP_routine	No Group	12.0
Pair 77	Finished	40	172.16.8.1	172.17.8.1	RTP		G.711a	VoIP_routine	No Group	12.0
Pair 78	Finished	40	172.16.8.1	172.17.8.1	RTP		G.711a	VoIP_routine	No Group	12.0
Pair 79	Finished	40	172.16.8.1	172.17.8.1	RTP		G.711u	VoIP_routine	No Group	10.0

Figure 5. Setup screen for VoIP Traffic

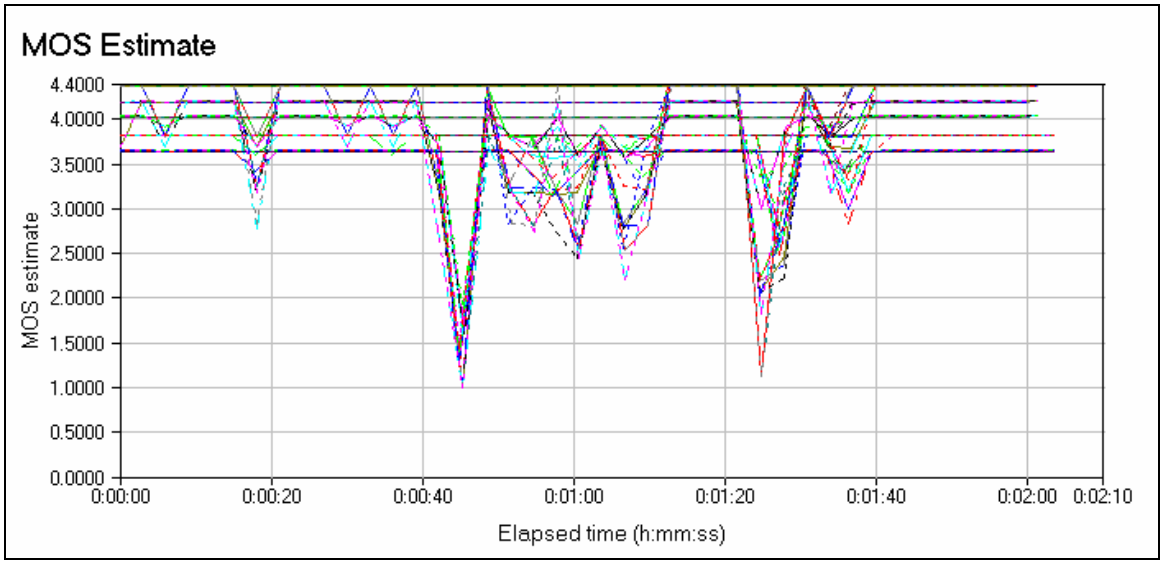

Figure 6. Sample results for VoIP Traffic Test

5. Running Combined Traffic

5.1 Objective and Setup

In this section of the test document, we will run a combination of the above traffic types. We will combine the traffic types by creating a new test window and then using cut-and-paste to copy all the port pair setup information into a single test. Then we'll duplicate the traffic, earmarking one for implementation of QoS policies and the other to continue running a routine QoS policy. Initially, we'll run both pairs and measure the outcome. Afterwards, we'll implement QoS policies on the duplicated pairs and implement the appropriate QoS policy on the DUT. Then we'll run the test again and compare the ports that have implemented QoS policies to the ports that have not implemented QoS.

5.2 Methodology

- Create a new IxChariot test structure.
- Open up the previously stored tests: baseline.tst, video.tst and voip.tst.
- On each of the previously stored tests, highlight all defined port pairs, copy to the Windows clipboard and then paste into the new test.
- Within each group, duplicate all pairs one time. Within each of these pairs, change the pair comment as follows:
  - Internet_Group_routine changes to Internet_Group_flash
  - Video_routine changes to Video_flash
  - VoIP_routine changes to VoIP_flash_override
- Save the new test as InternetMixWithQoS.tst and then as InternetMixWithNoQoS.tst. This will preserve the setups for running under both the non-QoS and QoS environments.

5.2.1 Without QoS

- Recall the InternetMixWithNoQoS.tst setup that was saved above. (This might already be on your screen if you saved the files in the order that was mentioned above.)
- Ensure the run parameter is set to run this test for two minutes.
- Run the test and store the results.
- Note that you can display the graphs by group comment.

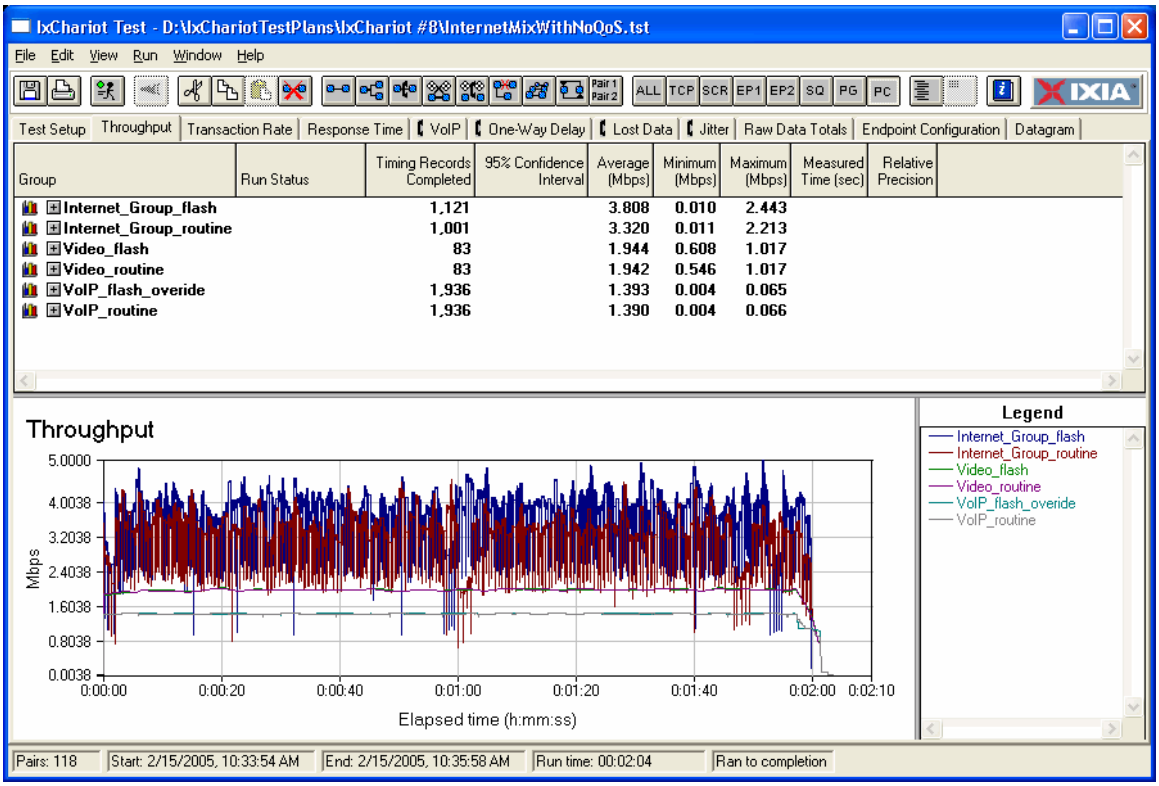

Figure 7. Throughput when running without QoS

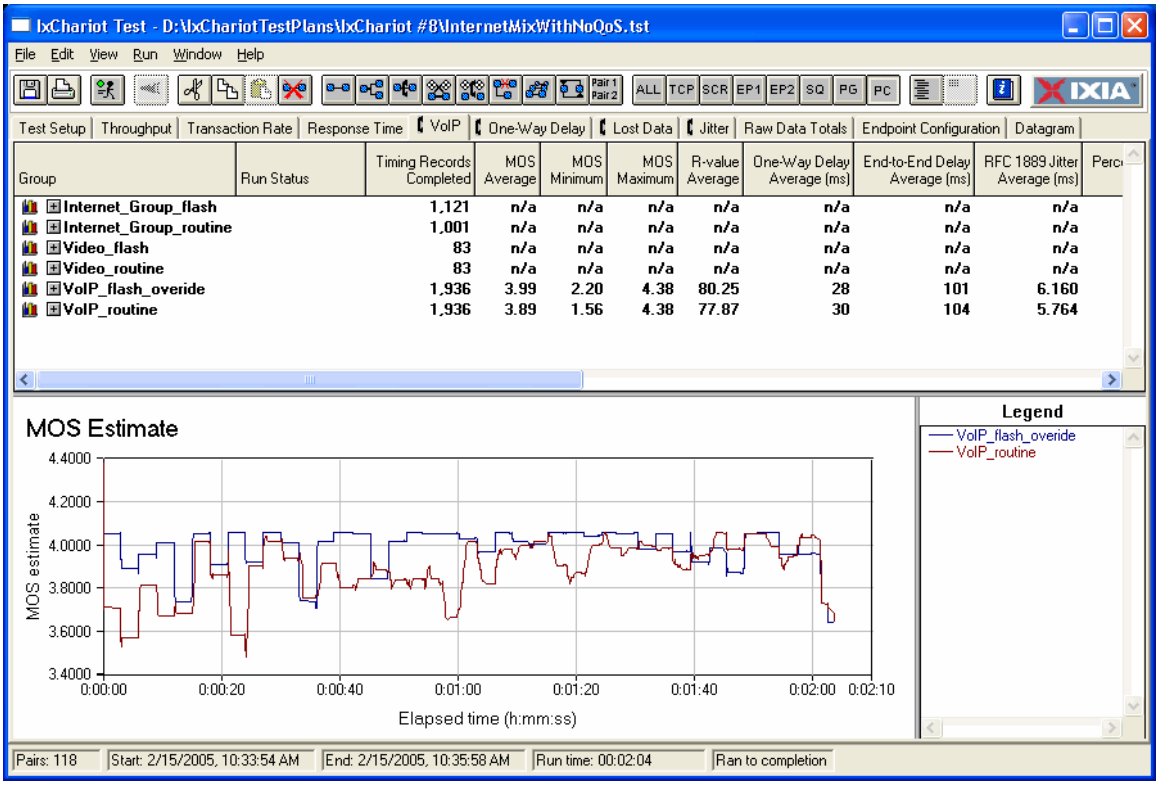

Figure 8. VoIP MOS score when running without QoS

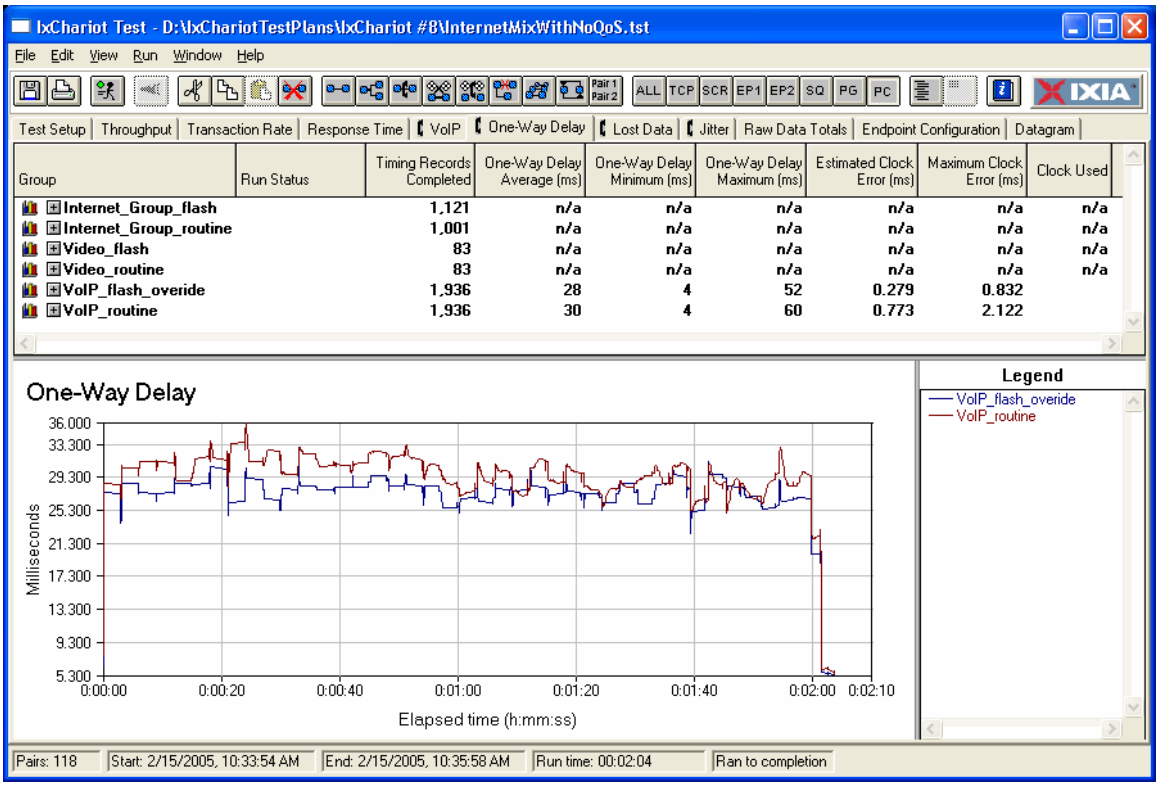

Figure 9. VoIP One-Way Delay when running without QoS

5.2.2 With QoS

- Recall the InternetMixWithQoS.tst setup that was saved previously.
- Define a new QoS parameter within IxChariot. This is performed through the main IxChariot window, under the menu Tools | Edit QoS Templates …. Create a new IP TOS template named "Flash Override" and set its precedence to "100 – Flash Override." Similarly, create another QoS parameter called "Flash," and set its precedence to "001 – Flash."
- In each of the voice pairs labeled "VoIP_flash_override," adjust the Service Quality to use the "Flash Override" template we defined earlier. This can be done quickly by highlighting all VoIP pairs and then right-clicking and selecting the "Edit…" option from the drop-down menu. Then within the edit dialog box, select the service quality to be "Flash Override."
- Similarly, in each of the video pairs labeled "Video_flash", adjust the Service Quality to use the "Flash" template we defined earlier.
- In each of the pairs within the "Internet_Group_flash" change the Service Quality to use the "Flash" template.
- Ensure that all remaining QoS settings are blank. This will provide a "Routine" QoS level.
- Change the policy on the DUT so that "Flash Override" has highest precedence, followed by "Flash," followed by "Routine."
- Run the test for two minutes and save the results.
- You should now be able to see and compare the QoS traffic from the non-QoS traffic in the results.

In the figures below, you can see the differences between packet pairs running with QoS and those running without QoS. In each test case, there is a clear separation of values that shows that the traffic marked for higher priority was indeed handled with higher priority.

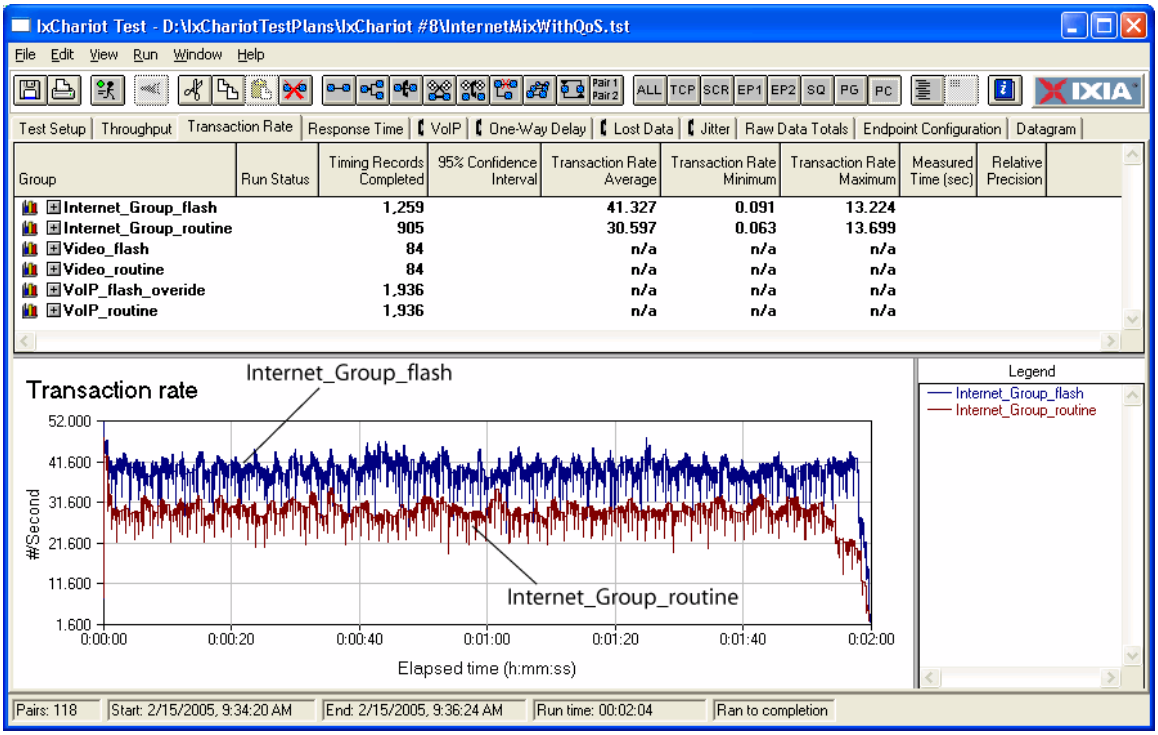

Figure 10. Comparison of Internet traffic throughput with different QoS levels

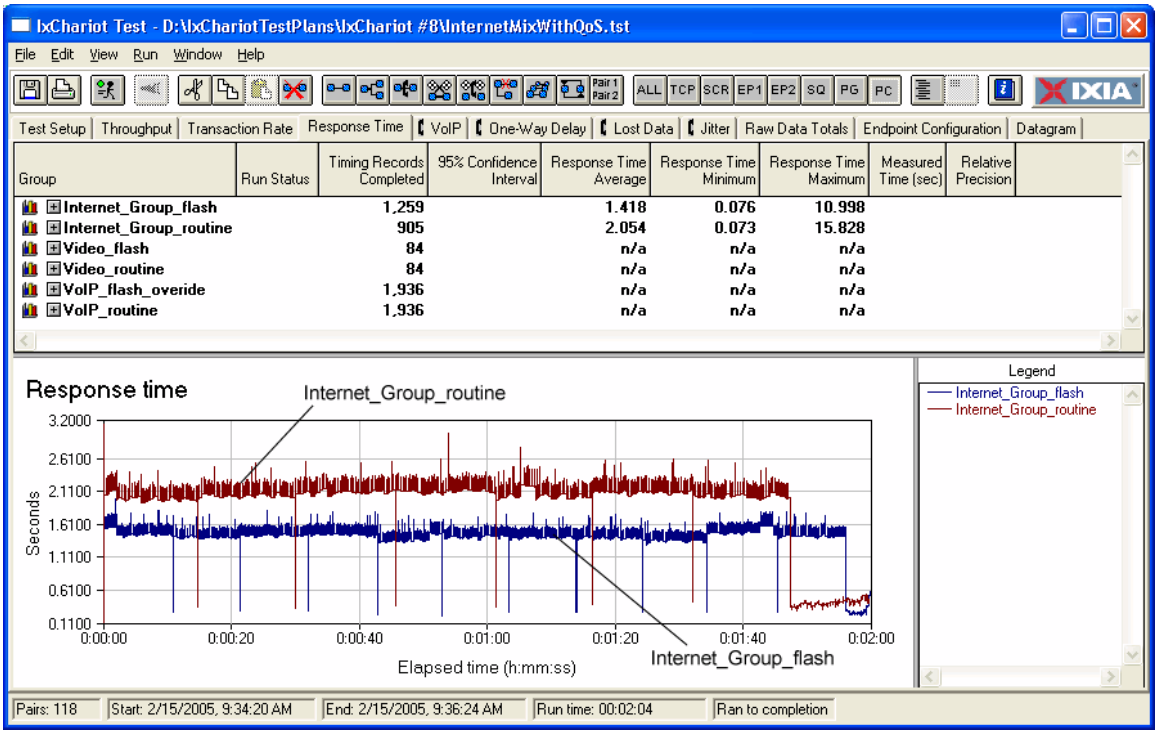

Figure 11. Comparison of Response Time with different QoS levels

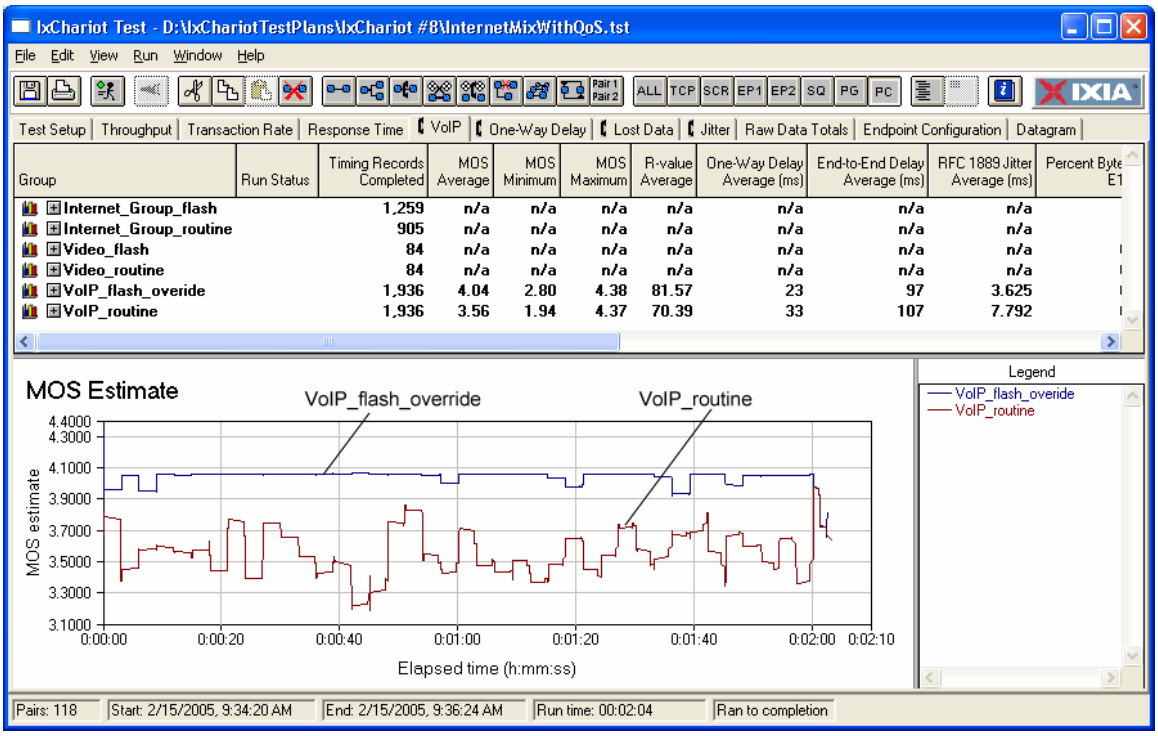

Figure 12. VoIP MOS estimates for different QoS levels

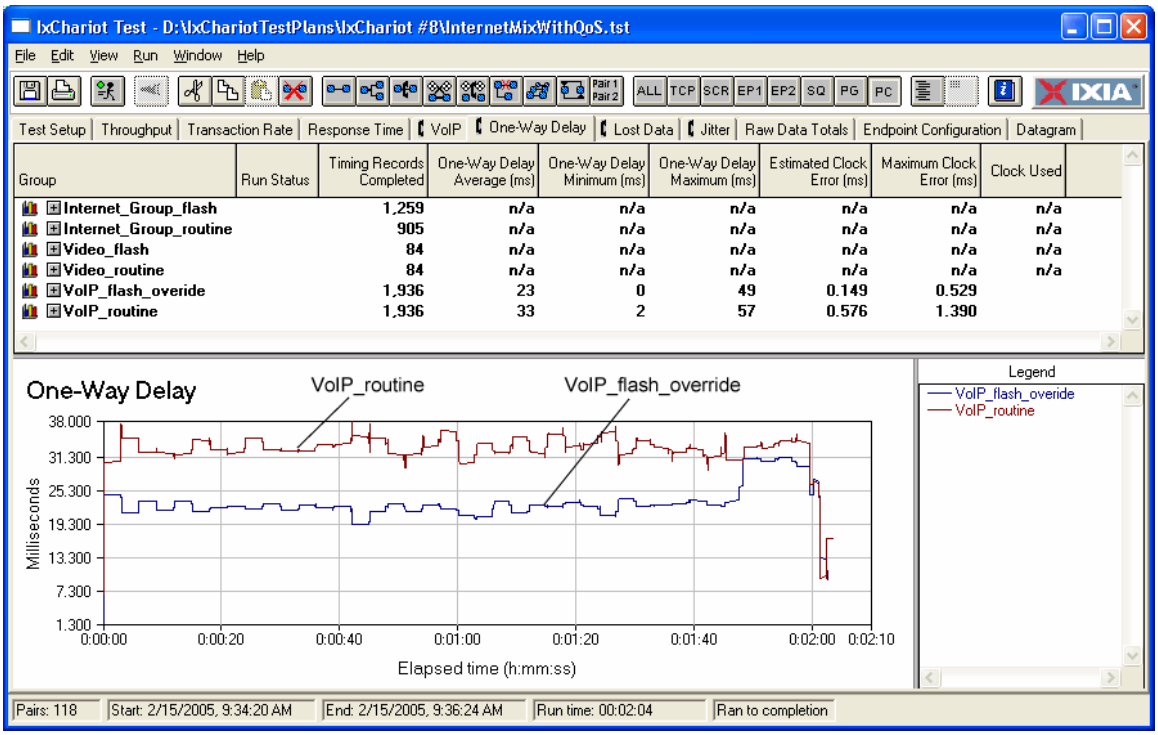

Figure 13. VoIP Delay characteristics with different QoS characteristics

5.2.3 Mixing QoS types

It is recommended that you repeat the above scenarios with different types of QoS and traffic types to verify that the DUT correctly differentiates between the two traffic types. Specifically, one should prioritize data transfers, such as the Internet Baseline traffic we developed in section 2, to have the lowest priority. Data services are generally not adversely affected by having to wait for results. Video traffic, as was developed in section 3 may be the next highest priority. A few missing frames of video will not seriously degrade its performance, as long as the audio track (which is typically sent as a stream) does not get broken. Finally, VoIP traffic, as developed in section 5, would typically have the highest priority, as voice services are very sensitive to latency and corrupted data problems.

6. Conclusion

IxChariot can be used to create many different types of layer 4-7 traffic, and each traffic pattern can be assigned different characteristics. Besides generating generic baseline Internet traffic, IxChariot can create a rich set of different traffic types that effectively test a DUT's ability to correctly handle these different types of traffic. For instance, IxChariot can simulate HTTP, FTP, email, video streams, audio streams and VoIP. IxChariot can measure DUT response in terms of latency, throughput, missing packets, transaction rate and Mean Opinion Score (MOS). Additionally, IxChariot traffic can be assigned different QoS parameters, allowing it to test a DUT's ability to correctly implement QoS policies on its traffic.