

Summary Test H.323 Bounds Test Report

Paul Schopis pschopis@itecoho.org

I. Introduction

This report contains the results of H.323 traffic bounds testing conducted by ITEC-Ohio. Sniffed packet data by ITEC-NC will be presented in another report. These are experiments to quantify which network metrics are relevant for good H.323 video performance. Our overall objective is to use this information to develop a meaningful QoS strategy that supports H.323. Please refer to the document “H.323 Traffic Characterization Test Plan Draft” for specific test plan details.

II. Scope and Definitions.

The challenge of collecting meaningful data points for H.323 is that strictly acquiring statistics such as video bandwidth and frame rates does not necessarily correlate well to picture quality. H.323 video intentionally adjusts both to compensate for network conditions. Conversely, reporting picture quality is rather subjective and it is likely that different individual’s exposure to various video mediums will affect their interpretation [1]. Additionally, dissimilar devices manifest network failures differently. For example, many appliance H.323 devices are NTSC machines while H.323 NICs use a computer’s display. Their replay capabilities differ significantly. It is not within the scope of these tests to venture into the areas of Cognitive Science. Additionally, although we take some hints from ANSI Standard T1.801.03 we employ a simpler system. Our task is to collect and report several data sets and attempt to create a system for constructing reasonable conclusions. One method is to classify audio and video events and report their frequency and duration, thus creating a simple video quality index. Quality in this context does not mean a level acceptability, rather an index that correlates the number and type of video events with the tested network conditions. The general categories of events are:

Spatial Augmentation – Video artifacts are added to the picture. Objects appear that are not in the captured video such as video tiles.

Spatial Depreciation – Parts of the picture or objects in the picture are missing.

Temporal Distortion – Over time the “flow” of an event is distorted by missing data, in mild cases resulting in an inter-frame jerkiness. In more severe cases resulting in video freezing.

Audio Augmentation – Audio artifacts added to audio stream such as pops, clicks and hiss.

Audio Depreciation – Parts of the audio are missing.

In this report the following terms are used.

Tiling – H.323 displays the picture in chunks commonly called tiles. The

edges of the tiles become visible at times of network distress. It is a type of Spatial Augmentation.

Tile Pulse – Tiling at fairly regular intervals.

Frame Freeze – The picture locks for a period of time.

Clipped Speech – A very small portion of the audio is missing, commonly at the beginning or end of a word. Frequently associated with consonance. Generally it does not result in undiscernable speech.

Audio Drop Out – Entire syllables or words are missing.

Further as a general starting reference we assume information from two sources to be of importance. First is the following table from the ITU Y.1541 draft standard as a reasonable set of assumptions of network performance metrics for various classes of service.

	Single value (SV)	Short range (Class 0)	Medium Range Class 1 Interactive	Wide Range Class 2 Non-interactive	Class 3 Unspecified
One Way Delay	Measured Value Empty Network	Less than SV+ 50ms (150 ms)	Less than SV+ 250 ms (400 ms)	Less than SV+ 10s (1 s)	U
IPDV	Between 0 and T for one MTU transmitted at line speed	25ms (50ms)	50ms (50ms)	None 1 s	U
Packet Loss (Probability)	Nil	<10 ⁻⁴ (10 ⁻³)	<10 ⁻³ (10 ⁻³)	<0.1 (10 ⁻³)	U

Second, is the notion that the IPDV can be no greater than the display time needed to create one frame's worth of data [2]. Recommendations from Polycom fall in line with this concept. Using this as a measure, the IPDV is independent of bandwidth and dependent on the frame rate. At 30 frames per second this time interval is 1/30th of a second, which is approximately 33ms. As mentioned earlier H.323 adjusts frame rate to compensate for network performance.

III. General Test Procedure.

It is our expressed intention to create a testing methodology that is easily executed and reproduced. This allows the community of interest to verify these findings of these tests and to use them as a trouble-shooting tool during periods of suspected network trouble. The basic procedure is to initiate a video call and observe the number and duration of video artifacts that appeared during a sixty-second sampling period. The timer used was an electronic metronome. The type of artifact was recorded in addition to its frequency and duration. Two basic types of scenes were captured. The first was a still scene (no movement) to observe inherent defects caused by the tested network condition. The still scene was not a test pattern but an empty office, in order to observe a more “real world” performance scenario [1]. The previously described data was recorded. The second scene was a scene with movement to capture the period of time required for the video to restabilize. The description of the network emulator follows:

Operating System: Linux Mandrake 7.2 Kernel recompiled and optimized for the device to be a router.

CPU: Pentium III 733Mhz

Memory: 256 MB.

Motherboard: Asus CUSLC2-C AGP4X

NICS: Intel Etherpro 10/100.

Emulator Software: Nistnet 2.1.0

Note: Several Hardware and Software configurations were tested. For the sake of brevity only the platform passing all tests and actually used is described.

A: Baseline test of software and hardware configuration.

A1. Test Objective:

Verify test mechanisms for bounds test. Verify that test platform routes correctly and that it performs within acceptable limits.

A2. Entry Criteria:

None.

A3. Test Setup:

1. Configure Linux platform with Nistnet software for network performance emulation. Load the Nistnet Kernel module.
2. Connect each Ethernet port on Linux platform to a Smartbits e.g. one as a transmit and one as a receive port.

A4. Test Execution: Basic performance

1. Verify Linux platform is routing correctly via route command.
2. Send traffic at 5.38 Load (~384k data bits) of 10M ethernet interface.
3. Set packet size to 128 byte packets.

A5. Result:

All packets were routed correctly.

A6. Repeat test with following values for load, 15% ,50%, 75-80% , 90-100%. Repeat test with the following values for packet sizes 1000 and 1500 bytes.

Results: Device under test passed all 128 byte packets.

100% Load 128 Byte packets					
	Events	Rates	Events	Rates	
	1-01 LN-3101A	1-01 LN-3101A	1-02 LN-3101A	1-02 LN-3101A	
Tx Frames	10,000	0	0	0	
Rx Frames	0	0	10,001	0	
Rx Bytes	0	0	1,280,064	0	
Rx Triggers	0	0	0	0	
Collisions	0	0	0	0	
CRC Errors	0	0	0	0	
Alignment Errors	0	0	0	0	
OverSize	0	0	0	0	
Frag/UnderSize	0	0	0	0	
Tx From Stack	0		1		
Rx To Stack	0		1		
ARP Replies Sent	0		1		
ARP Requests Sent	0		0		
ARP Replies Received	0		0		
ARP Requests Received	0		1		
PING Replies Sent	0		0		
PING Requests Sent	0		0		
PING Replies Received	0		0		
PING Requests Received	0		0		
Signature Frames Sent	10,000		0		
Signature Frames Received	0		10,000		
Data Integrity Errors	0		0		
VLAN Frames Received	0		0		

Device under test started dropping 1000 byte packets at ~85% Load.

	Events	Rates	Events	Rates	
	1-01 LN-3101A	1-01 LN-3101A	1-02 LN-3101A	1-02 LN-3101A	
Tx Frames	10,000	0	0	0	

Rx Frames	1	0	10,006	0
Rx Bytes	130	0	9,999,805	0
Rx Triggers	0	0	0	0
Collisions	0	0	0	0
CRC Errors	0	0	1	0
Alignment Errors	0	0	0	0
OverSize	0	0	0	0
Frag/UnderSize	0	0	4	0
Tx From Stack	0		6	
Rx To Stack	0		6	
ARP Replies Sent	0		6	
ARP Requests Sent	0		0	
ARP Replies Received	0		0	
ARP Requests Received	0		6	
PING Replies Sent	0		0	
PING Requests Sent	0		0	
PING Replies Received	0		0	
PING Requests Received	0		0	
Signature Frames Sent	10,000		0	
Signature Frames Received	0		9,999	
Data Integrity Errors	0		0	
VLAN Frames Received	0		0	

Device under test started dropping 1500 byte packets at ~85% Load.

	Events	Rates	Events	Rates
	1-01 LN-3101A	1-01 LN-3101A	1-02 LN-3101A	1-02 LN-3101A
Tx Frames	10,000	0	0	0
Rx Frames	0	0	10,006	0
Rx Bytes	0	0	14,999,277	0
Rx Triggers	0	0	0	0
Collisions	0	0	0	0
CRC Errors	0	0	1	0
Alignment Errors	0	0	0	0
OverSize	0	0	0	0
Frag/UnderSize	0	0	5	0
Tx From Stack	0		7	
Rx To Stack	0		7	
ARP Replies Sent	0		7	

ARP Requests Sent	0		0		
ARP Replies Received	0		0		
ARP Requests Received	0		7		
PING Replies Sent	0		0		
PING Requests Sent	0		0		
PING Replies Received	0		0		
PING Requests Received	0		0		
Signature Frames Sent	10,000		0		
Signature Frames Received	0		9,999		
Data Integrity Errors	0		0		
VLAN Frames Received	0		0		

Conclusions:

Performance is acceptable since the maximum test load is approximately 15% and the device under test passes 100% packets sent at 15% load regardless of packet size.

A7. Test Execution: Nistnet reliability for drop.

1. Configure Nistnet to drop 5% of traffic.
2. Send traffic from the Smartbits into E0 and route out E1 to Smartbits.

A8. Expected Result:

It is expected that the Nistnet will drop the specified traffic.

A9. Repeat test using drop of 1%, 0.1%, 0.01% etc.

Drop tests							
1. Load 5.38% Drop 5%							
1	Signature Frames Sent	10,000		0			
	Signature Frames Received	0		9,507			
2	Signature Frames Sent	10,000		0			
	Signature Frames Received	0		9,497			
3	Signature Frames Sent	10,000		0			
	Signature Frames Received	0		9,540			
2. Load 5.38% Drop 1%							
1	Signature Frames Sent	10,000		0			
	Signature Frames Received	0		9,904			

2	Signature Frames Sent	10,000	0			
	Signature Frames Received	0	9,892			
3	Signature Frames Sent	10,000	0			
	Signature Frames Received	0	9,901			
3. Load 5.38% Drop 0.75						
1	Signature Frames Sent	10,000	0			
	Signature Frames Received	0	9,928			
2	Signature Frames Sent	10,000	0			
	Signature Frames Received	0	9,913			
3	Signature Frames Sent	10,000	0			
	Signature Frames Received	0	9,923			
4. Load 5.38% Drop 0.50%						
1	Signature Frames Sent	10,000	0			
	Signature Frames Received	0	9,944			
2	Signature Frames Sent	10,000	0			
	Signature Frames Received	0	9,953			
3	Signature Frames Sent	10,000	0			
	Signature Frames Received	0	9,955			
1	Signature Frames Sent	10,000	0			
	Signature Frames Received	0	9,984			
2	Signature Frames Sent	10,000	0			
	Signature Frames Received	0	9,971			
3	Signature Frames Sent	10,000	0			
	Signature Frames Received	0	9,967			
Repeat test with a larger data set for better resolution. Increase by one order of magnitude.						
100,000 packets						
1	Signature Frames Sent	100,000	0			
	Signature Frames Received	0	99,752			
2	Signature Frames Sent	100,000	0			

	Signature Frames Received	0		99,751		
3	Signature Frames Sent	100,000		0		
	Signature Frames Received	0		99,754		
6. Load 5.38% Drop 0.10%						
1	Signature Frames Sent	100,000		0		
	Signature Frames Received	0		99,891		
2	Signature Frames Sent	100,000		0		
	Signature Frames Received	0		99,905		
3	Signature Frames Sent	100,000		0		
	Signature Frames Received	0		99,894		
7. Load 5.38% drop 0.01%						
1	Signature Frames Sent	100,000		0		
	Signature Frames Received	0		99,990		
2	Signature Frames Sent	100,000		0		
	Signature Frames Received	0		99,988		
3	Signature Frames Sent	100,000		0		
	Signature Frames Received	0		99,990		
8. Load 15% Drop 5%						
1	Signature Frames Sent	100,000		0		
	Signature Frames Received	0		99,476		
2	Signature Frames Sent	100,000		0		
	Signature Frames Received	0		99,520		
3	Signature Frames Sent	100,000		0		
	Signature Frames Received	0		99,477		
9. Load 15% drop 1%						
1	Signature Frames Sent	100,000		0		
	Signature Frames Received	0		99,903		
2	Signature Frames Sent	100,000		0		
	Signature Frames Received	0		99,897		

	Received						
3	Signature Frames Sent	100,000		0			
	Signature Frames Received	0		99,898			
10. Load 15% drop 0.75%							
1	Signature Frames Sent	100,000		0			
	Signature Frames Received	0		99,235			
2	Signature Frames Sent	100,000		0			
	Signature Frames Received	0		99,257			
3	Signature Frames Sent	100,000		0			
	Signature Frames Received	0		99,240			
11. Load 15% drop 0.50%							
1	Signature Frames Sent	100,000		0			
	Signature Frames Received	0		99,498			
2	Signature Frames Sent	100,000		0			
	Signature Frames Received	0		99,504			
3	Signature Frames Sent	100,000		0			
	Signature Frames Received	0		99,494			
12 Load 15% drop 0.1%							
1	Signature Frames Sent	100,000		0			
	Signature Frames Received	0		99,906			
2	Signature Frames Sent	100,000		0			
	Signature Frames Received	0		99,890			
3	Signature Frames Sent	100,000		0			
	Signature Frames Received	0		99,907			
13. Load 15% drop 0.01%							
1	Signature Frames Sent	100,000		0			
	Signature Frames Received	0		99,989			
2	Signature Frames Sent	100,000		0			
	Signature Frames Received	0		99,990			

3	Signature Frames Sent	100,000		0		
	Signature Frames Received	0		99,988		

Conclusion: The Nistnet device correctly approximates configured drop rates.

A10. Repeat testing procedure for latency using 50ms, 100ms, and 500ms delay.

Note: All times are reported in microseconds.

8. Emulator turned on with targets entered.					5.38%			
	Chassis	Port	Stream	Frames Rcvd	Minimum	Maximum	Average	
1	1	1-01	1	10000	153.2	775.6	155.3	
2	1	1-01	1	10000	153.1	185	155.1	
3	1	1-01	1	10000	153.1	185.3	155.3	
9. Emulator running at 50ms delay 5.38% loaded								
	Chassis	Port	Stream	Frames Rcvd	Minimum	Maximum	Average	
1	1	1-01	1	10000	50208	51096.8	50272.4	
2	1	1-01	1	10000	50208.2	50775.2	50269.5	
3	1	1-01	1	10000	50207.6	50338.6	50269.3	
10. Emulator running at 50ms delay 15% loaded								
	Chassis	Port	Stream	Frames Rcvd	Minimum	Maximum	Average	
1	1	1-01	1	10000	50208.2	50735.5	50269.8	
2	1	1-01	1	10000	50208.3	50338.2	50269.8	
3	1	1-01	1	10000	50208.4	50331.5	50269.7	
11. Emulator running at 100ms delay 15% loaded								
	Chassis	Port	Stream	Frames Rcvd	Minimum	Maximum	Average	
1	1	1-01	1	10000	100133.9	100257.2	100195.5	
2	1	1-01	1	10000	100134	100279.8	100195.6	
3	1	1-01	1	10000	100133.9	103109.9	100240.8	
12. Emulator running at 100ms 5.38 % loaded								

	Chassis	Port	Stream	Frames Rcvd	Minimum	Maximum	Average
1	1	1-01	1	10000	100133.6	100263.3	100196.3
2	1	1-01	1	10000	100133.6	100263.3	100196.3
3	1	1-01	1	10000	100133.7	106601.7	100253.6
13. Emulator running at 500ms 5.38% load							
	Chassis	Port	Stream	Frames Rcvd	Minimum	Maximum	Average
1	1	1-01	1	10000	500149.8	500283.8	500211.3
2	1	1-01	1	10000	500149.6	500279.4	500211.3
3	1	1-01	1	10000	500149.5	500273.9	500211.1
14. Emulator running at 500ms 15% load							
	Chassis	Port	Stream	Frames Rcvd	Minimum	Maximum	Average
1	1	1-01	1	10000	500150.2	500278.1	500211.7
2	1	1-01	1	10000	500149.8	500287.6	500211.5
3	1	1-01	1	10000	500149.8	500287.6	500211.5

Conclusion: Nistnet device exceeds the required parameters of + or – 1 ms.

A11. Repeat testing procedure for jitter using 100ms delay and 11.5 sigma with no correlation and work downward to find approximately 10ms steps in the jitter window.

1. 15% Load 100ms 11.5 sigma				~60ms			
	Chassis	Port	Stream	Frames Rcvd	Minimum	Maximum	Average
1	1	1-01	1	10000	85400.4	146276.1	100117.2
2	1	1-01	1	10000	85401.7	146278.3	100113
3	1	1-01	1	10000	85392	146271.3	100090.9
2. 15% Load 100ms 9.5		Sigma	~50ms				

	Chassis	Port	Stream	Frames Rcvd	Minimum	Maximum	Average
1	1	1-01	1	10000	87965.8	138220.3	100114.9
2	1	1-01	1	10000	87965.6	138325.1	100186.7
3	1	1-01	1	10000	87940.1	138938.2	100126.1
3. 15% Load 100ms delay 8.5		7.9 sigma					
	Chassis	Port	Stream	Frames Rcvd	Minimum	Maximum	Average
1	1	1-01	1	10000	90026.7	131946.8	100159.6
2	1	1-01	1	10000	90068.3	131961.1	100158.4
3	1	1-01	1	10000	90046.6	131960.6	100151.7
4. 15% Load 100ms delay 7.7		7.5 sigma	~40ms				
	Chassis	Port	Stream	Frames Rcvd	Minimum	Maximum	Average
1	1	1-01	1	10000	90509.7	130393.9	100139.7
2	1	1-01	1	10000	90521.7	130296.7	100122.6
3	1	1-01	1	10000	90510.1	139337.7	100362.8
5. 15% Load 100ms delay 6.5			~35ms	7.0sigma			
	Chassis	Port	Stream	Frames Rcvd	Minimum	Maximum	Average
1	1	1-01	1	10000	91738.7	126297.8	100176.6
2	1	1-01	1	10000	91722.3	126310.4	100167.9
3	1	1-01	1	10000	91726.7	126291.5	100172.5
6. 15% Load 100ms delay 6.0 sigma			~32ms				
	Chassis	Port	Stream	Frames Rcvd	Minimum	Maximum	Average
1	1	1-01	1	10000	92451.9	124318.7	100190.5
2	1	1-01	1	10000	92489	124306.9	100151.4
3	1	1-01	1	10000	92465	124301.2	100210.6
7. 15% Load 100ms delay 5.5 sigma							
	Chassis	Port	Stream	Frames Rcvd	Minimum	Maximum	Average
1	1	1-01	1	10000	93143.6	122278.7	100218.2
2	1	1-01	1	10000	93144.5	122279.2	100182.6
3	1	1-01	1	10000	93135.7	122267.1	100177.9

8. 15% Load 100ms delay 5.6 sigma					~30ms			
	Chassis	Port	Stream	Frames Rcvd	Minimum	Maximum	Average	
1	1	1-01	1	10000	93069.2	122825.7	100173	
2	1	1-01	1	10000	93145.4	122851	100174.1	
3	1	1-01	1	10000	93128.9	122854	100206.2	
9. 15% Load 100ms delay 4.5 sigma					~23 ms			
	Chassis	Port	Stream	Frames Rcvd	Minimum	Maximum	Average	
1	1	1-01	1	10000	94458.3	118345.4	100171.4	
2	1	1-01	1	10000	94505.3	118389.9	100204.6	
3	1	1-01	1	10000	94506.4	118385	100184.2	
10. 15% Load 100ms		4.0 Sig	~21 ms	Jitter				
	Chassis	Port	Stream	Frames Rcvd	Minimum	Maximum	Average	
1	1	1-01	1	10000	95033.6	116304.5	100233.4	
2	1	1-01	1	10000	95176.4	116294	100201	
3	1	1-01	1	10000	95007.9	116295.8	100195.3	
11. 15% Load 100ms delay		3.9 sigma	~21ms jitter					
	Chassis	Port	Stream	Frames Rcvd	Minimum	Maximum	Average	
1	1	1-01	1	10000	95165.3	115995.9	100207.6	
2	1	1-01	1	10000	95192.5	115994.3	100194.4	
3	1	1-01	1	10000	95166.3	115996.5	100215	
12. 15 % Load 100ms delay 3.7 sigma								
~20ms jitter								
	Chassis	Port	Stream	Frames Rcvd	Minimum	Maximum	Average	
1	1	1-01	1	9999	95379	115168.5	100238.1	
2	1	1-01	1	10000	95446.2	115164.2	100205.1	
3	1	1-01	1	10000	95387.6	115106.2	100197.1	
13. 15% Load 100ms delay 2.0 sigma								
~10 ms jitter								
	Chassis	Port	Stream	Frames Rcvd	Minimum	Maximum	Average	
1	1	1-01	1	10000	97577.9	108311.8	100269.9	
2	1	1-01	1	9999	97596.9	108389.8	100211.4	
3	1	1-01	1	10000	97586.6	108904.6	100272.3	

14. 5.38% Load 100ms delay 11.5 sigma ~61ms jitter							
	Chassis	Port	Stream	Frames Rcvd	Minimum	Maximum	Average
1	1	1-01	1	10000	85375.5	146270.3	100060.1
2	1	1-01	1	10000	85382.2	146275.2	100104
3	1	1-01	1	10000	85372.2	146275.3	100114.8
15. 5.38 % Load 100ms delay 9.5 sigma ~ 50ms jitter							
	Chassis	Port	Stream	Frames Rcvd	Minimum	Maximum	Average
1	1	1-01	1	10000	1	138219.5	100116.9
2	1	1-01	1	10000	87962.8	145224.8	100185.6
3	1	1	1	10000	87970.4	138217.4	100143.2
16. 5.38% Load 100ms delay 7.9 sigma ~42 ms jitter							
	Chassis	Port	Stream	Frames Rcvd	Minimum	Maximum	Average
1	1	1-01	1	10000	90005.5	131869	100115.3
2	1	1-01	1	10000	90062.7	131865.5	100186.8
3	1	1-01	1	10000	90006.1	131870.4	100102.3
17. 5.38% Load 100ms delay 7.5 sigma ~40ms jitter							
	Chassis	Port	Stream	Frames Rcvd	Minimum	Maximum	Average
1	1	1-01	1	10000	90497.3	130281.3	100151.7
2	1	1-01	1	10000	90499.6	131032.5	100121.7
3	1	1-01	1	10000	90539.8	130281.8	100192.2
18. 5.38% Load 100ms delay 5.6 sigma ~30 ms jitter							
	Chassis	Port	Stream	Frames Rcvd	Minimum	Maximum	Average
1	1	1-01	1	10000	93000.9	122708.2	100186.5
2	1	1-01	1	10000	92956.4	122712.9	100152.7
3	1	1-01	1	10000	92939.6	122713.6	100169.7
19. 5.38% Load 100ms delay 3.7 sigma ~20 ms jitter							
	Chassis	Port	Stream	Frames Rcvd	Minimum	Maximum	Average

1	1	1-01	1	10000	95462.4	115025.4	100189
2	1	1-01	1	10000	95464.7	115026.1	100171.7
3	1	1	1	10000	95488.3	115028.1	100202
20. 5.38 % Load 100ms delay 2.0 sigma ~10ms jitter							
	Chassis	Port	Stream	Frames Rcvd	Minimum	Maximum	Average
1	1	1-01	1	10000	97611.6	108312.1	100208.3
2	1	1-01	1	10000	97610.6	108311.1	100209.1
3	1	1-01	1	10000	97609.5	108308.4	100204.8

Conclusion: Nistnet device approximates jitter and exceeds the + or – 1 ms acceptability parameter at the required load bounds.

B: Point to point H.323 client bounds test.

B1: Test objective:

Test performance bounds for Loss, Latency and Jitter for point-to-point H.323 client calls.

B2: Entry criteria:

Test A has been performed and there is at least a consistent functional basis for data interpretation.

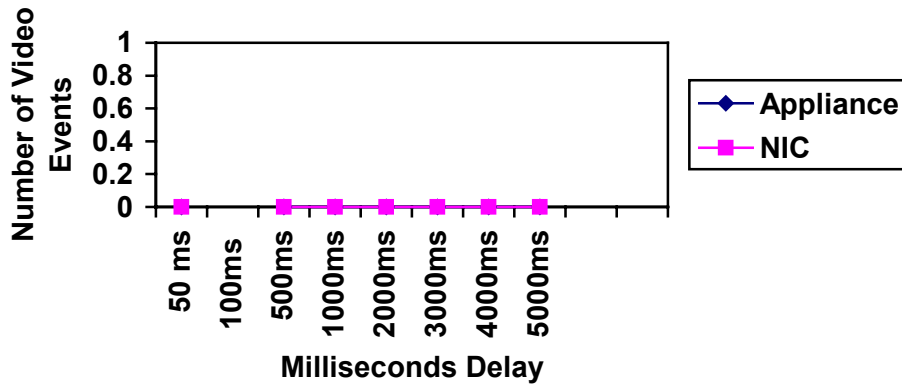
B3: Test set up:

Connect two H.323 clients back to back through the network emulation device (Nistnet).

B3. Test Execution:

1. Configure Nistnet platform to increase latency by 10ms.
2. Establish call between Clients at 768Kbps.
3. Observe call quality
4. Repeat calls with increasing latency every iteration by 10ms until call quality deteriorates.

Latency Errored Seconds



Latency for Point-to-Point Connections.

Conclusion for Latency:

Latency seems to have no effect on Video quality.

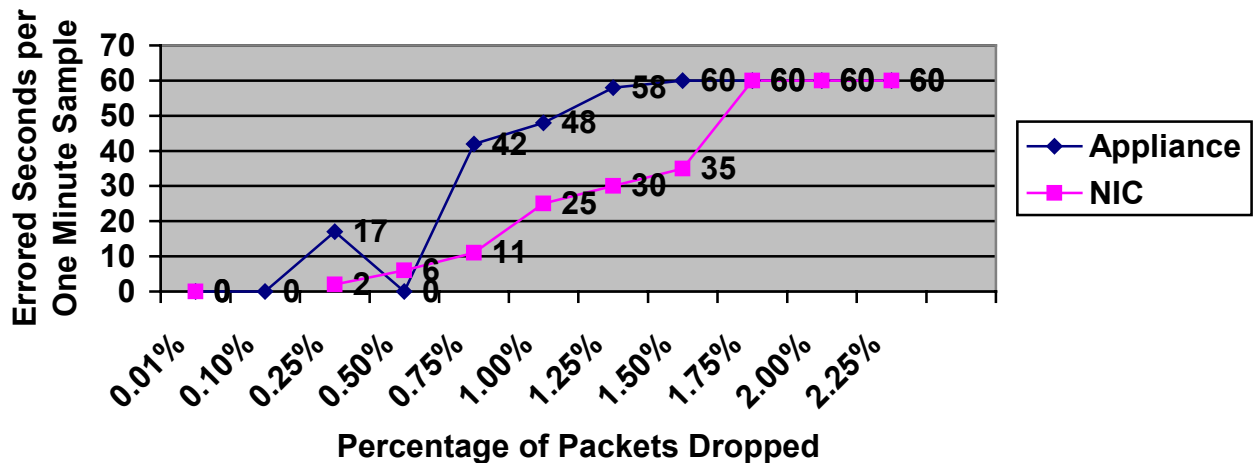
B4. Test Execution:

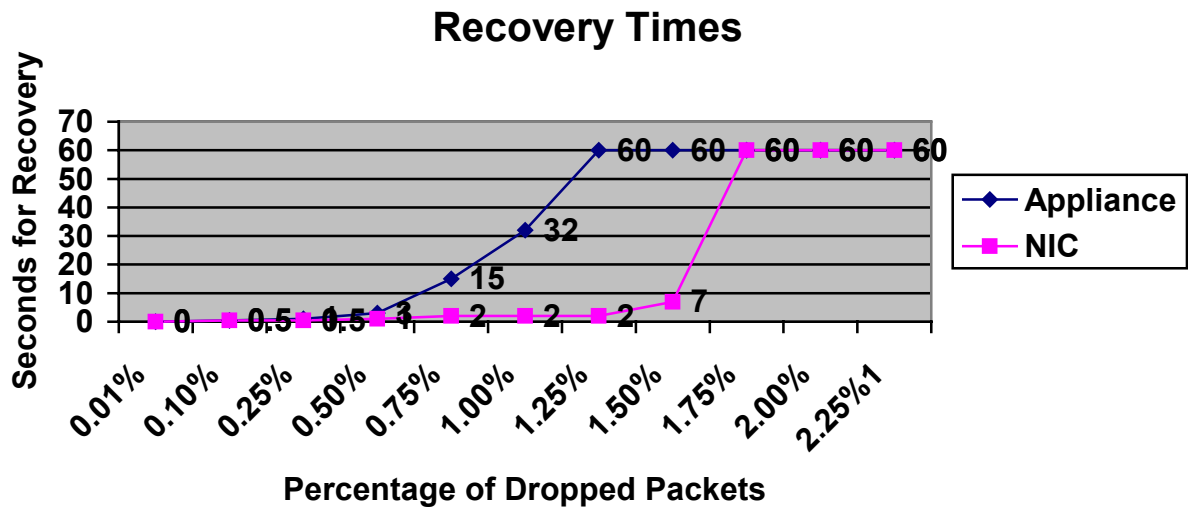
1. Configure Nistnet platform to increase drop starting at 0.01% to 2.25%.

The following data points for drop will be tested, 0.01%, 0.1%, 0.25%, 0.5%, 0.75%, 1%, 1.25%, 1.5%, 1.75%, 2.0%, 2.25%.

2. Establish call between Clients at 768Kbps.
3. Observe call quality.

Drop Errored Seconds





Observed Video Defects:

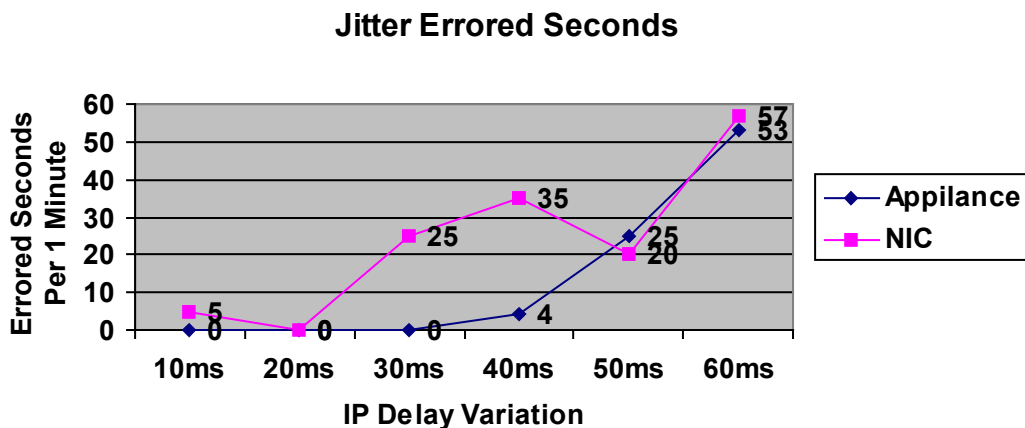
Tiling, frame freezing and jerkiness in motion were the observed defects with latter two being predominant. Clipped speech was noticeable at 1% loss with whole word drop out occurring at 1.25%.

Conclusions:

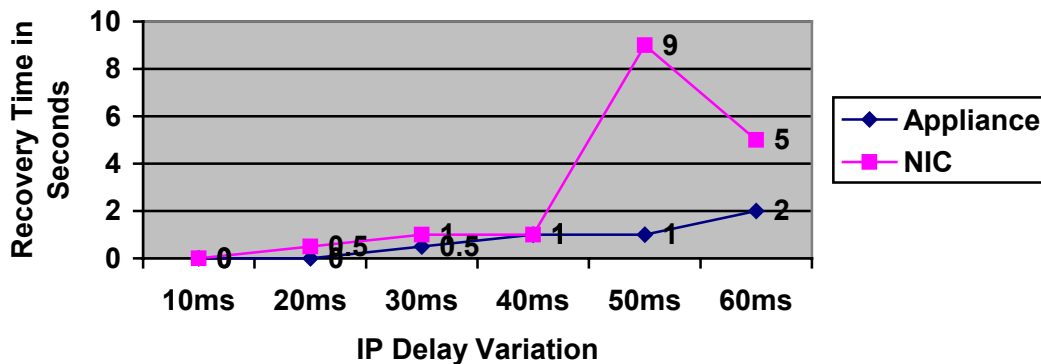
In point to point connections the drop rate should be below 1.0%. The NIC based unit performed better than the appliance unit.

B5. Test Execution:

1. Configure Nistnet platform to increase jitter by 10ms.
2. Establish call between Clients at 768Kbps.
5. Observe call quality
6. Repeat calls with increasing jitter every iteration by 10ms until call quality deteriorates.



Jitter Recovery Time



Observed Video Defects:

Tile Pulse as IPDV increased on the NIC unit while the Appliance showed pixel shift. Ghost image and picture freezing were both present in both units. Speech clipping was noticeable at 40ms and drop out occurred at 50ms.

Conclusions:

While recovery times seemed to be quite quick when compared to drop the overall performance was quite bad especially for the NIC device. The notion of IPDV being no greater than the required frame time seems to hold true.

C: Point to point H.323 client to MCU bounds test.

C1. Test objective:

Test performance bounds for latency between an MCU and client.

C2. Test set up:

Connect an H.323 client back to back to an MCU through the network emulation device (Nistnet).

C3. Test Execution:

1. Configure Nistnet platform to increase latency by 10ms.
2. Establish call between Clients at 768Kbps.
3. Observe call quality
4. Repeat calls with increasing latency every iteration by 10ms until call quality deteriorates.

Observed Video Defects:

Very few artifacts were observed. The NIC unit did have some mild Tiling when the video scene had quick movement. Recovery time was well under one second and was not observed until there was 100ms latency. The recovery time remained constant up through a latency of 2000ms.

Conclusion:

Latency is not much of a factor between an MCU and Client.

C5. Test objective:

Test performance bounds for loss and between an MCU and client.

C6. Test set up:

Connect an H.323 client back to back to an MCU through the network emulation device (Nistnet).

C7. Test Execution:

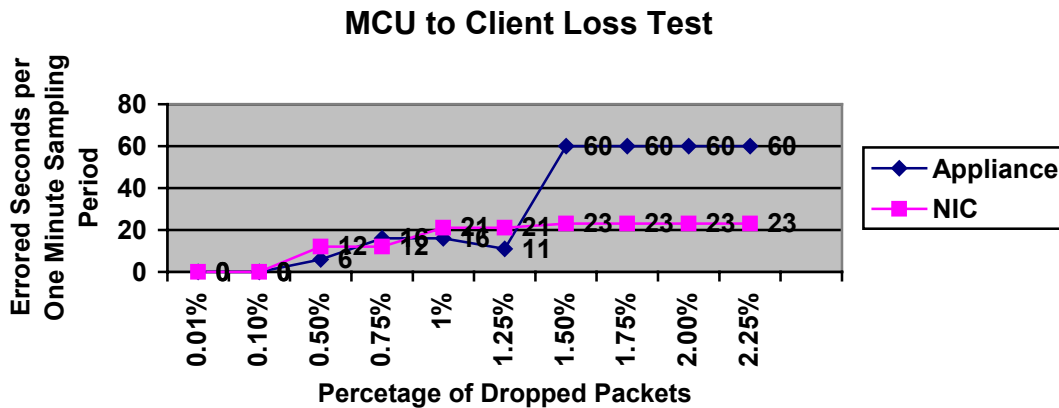
1. Configure Nistnet platform to increase drop starting at 0.01% to 2.25%.
The following data points for drop will be tested, 0.01%, 0.1%, 0.5%, 0.75%, 1%, 1.25%, 1.5%, 1.75%, 2.0%, 2.25%.
2. Establish call between Clients at 768Kbps.
4. Observe call quality.

Observed Video Defects:

Congestion warning on both clients, tiling as loss increased, split image on NIC device.

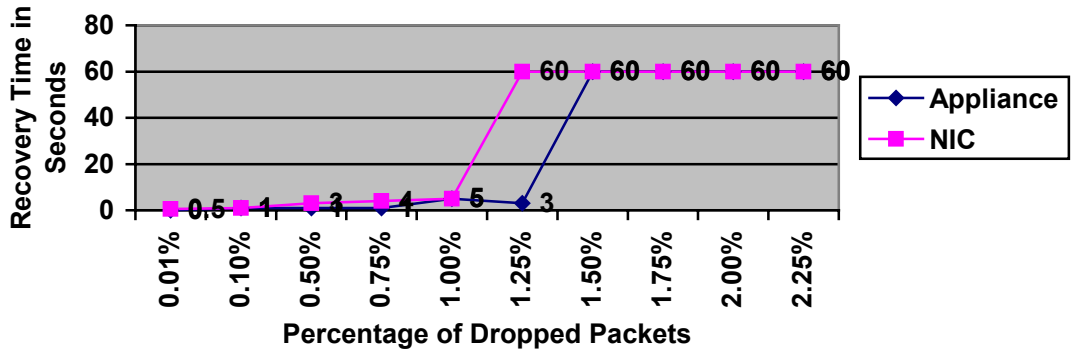
Conclusion:

On both clients the loss rate between 1% and 1.25% renders nearly chronic failure as the client never recovers completely.



NOTE: The calls failed completely for the Appliance at 1.5% packet drop.

MCU to Client Recovery Time for Loss



C1. Test objective:

Test performance bounds for jitter and between an MCU and client.

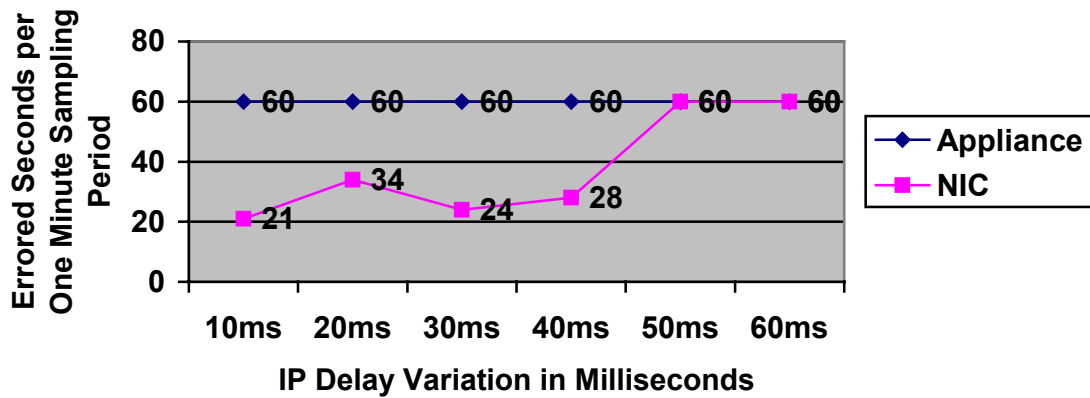
C2. Test set up:

Connect an H.323 client back to back to an MCU through the network emulation device (Nistnet).

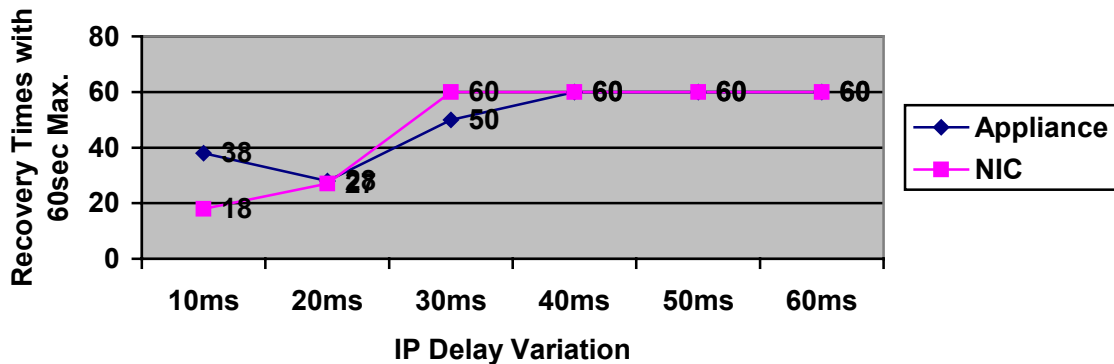
C3. Test Execution:

1. Configure Nistnet platform to increase jitter by 10ms.
2. Establish call between Clients at 768Kbps.
5. Observe call quality
6. Repeat calls with increasing jitter every iteration by 10ms until call quality deteriorates.

MCU to Client Jitter Errored Seconds



MCU to Client Recovery Time for Jitter



Observed Video Defects:

Image splitting, Tile freezing, Frame freezing and Ghost Images. Defects quickly max out as they start to overlap quickly.

Conclusions:

Jitter between an MCU and end point is quite disruptive, particularly when jitter exists on both the send and receive channels. The one-way case will be tested in a later test.

D: Switch Propagation Delay Test

D1. Test Objective:

The purpose of this test is to determine the worst case propagation delay introduced by the switch that will be used for the cascading of MCUs in subsequent tests.

D2. Test Set Up.

Plug two ports of Smartbits back to back with an ethernet cross over cable

D3. Test Execution.

Pass traffic between ports at 10 Mbps at 1 MTU (1500 bytes).

D4. Findings

1. Smartbits back to back

Chassis	Port	Stream	Frames Rcvd	Minimum	Maximum	Average
1	1	1-01	1	10000	0.5	0.6
						0.5

The Smartbits has a worst case of 0.6 ms and average 0.5 ms propagation and processing delay.

D5. Test Set Up.

Plug two ports of a Cisco 2924XL into the Smartbits into 10/100 ports.

D6. Test Execution.

1. Configure ports on the 2924XL and the ports on the Smartbits to match the transmit and receive speeds and duplex settings of the MCUs that will be used in test E.
2. Pass traffic at 10Mbps using one MTU packet size. From one port to the other on the Smartbits.
3. Note delay in traffic.

D7. Results:

2. Switch Test

Chassis	Port	Stream	Frames Rcvd	Minimum	Maximum	Average
1	1	1-01	1	10000	1221.1	1978.2
2	1	1-01	1	10000	1220.9	1818.1
3	1	1-01	1	10000	1220.9	1820.8

Conclusion:

As a worst case scenario the maximum expected delay is 1.978 ms and an average of 1.572 ms.

E: Multiple clients connected via MCU test.

E1. Objective:

The purpose of this test is to see the effects of latency using an MCU for one way network problems.

E2. Test Set Up.

1. Plug one client directly into the Network Emulation device.
2. Plug the Cisco 2924XL into the second port of the Nistnet device.
Note: each Nistnet port is a different subnet.
3. Plug the MCU and the other client into the Cisco Switch.
4. Configure a common conference on the MCU and register it with the on board GateKeeper.
5. Register both clients with the GateKeeper.

E3. Test Execution.

1. Configure Nistnet platform to increase latency by 10ms.
2. Establish call between Clients Through MCU at 768Kbps.
3. Observe call quality
4. Repeat calls with increasing latency every iteration by 10ms until call quality deteriorates.

E4. Results.

We tested with 50ms through 2000ms delay and as in previous tests there was little effect. Starting at 100ms there was brief tiling with movement that recovered in less than 1 second. This remained the same up through 2000ms. There was no effect on still scenes.

E5. Objective:

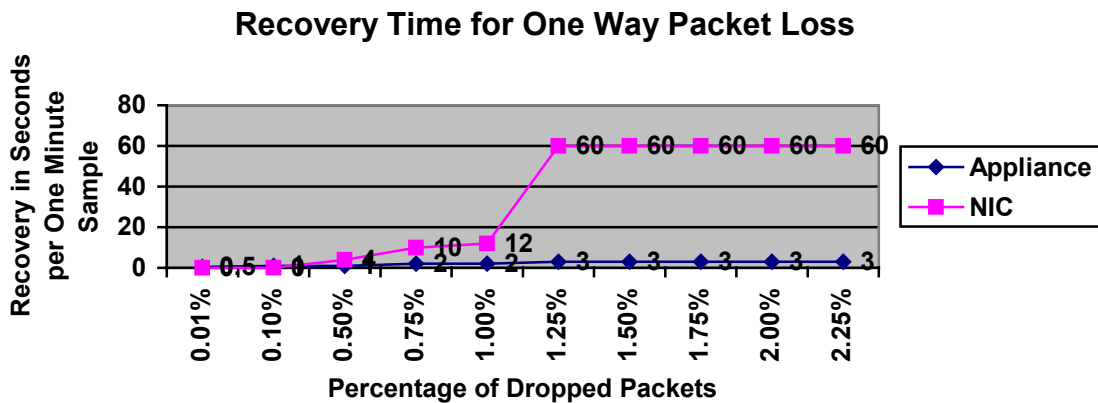
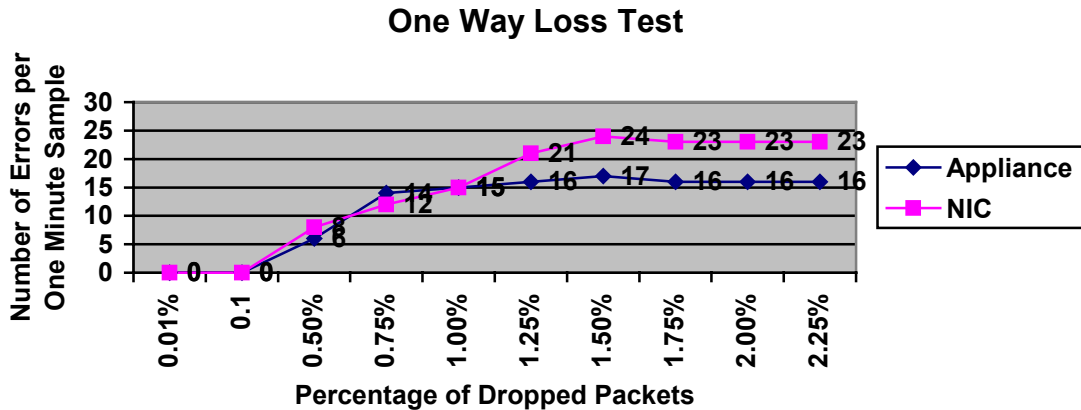
The purpose of this test is to see the effects of loss using an MCU for one way network problems. Secondly, since the errors are introduced between the Appliance and the MCU and the NIC is on the same subnet as the MCU. We wish to evaluate the MCU's role as a replay buffer.

E6. Test Set Up.

Test set up is the same as section E2.

E7. Test Execution.

1. Configure Nistnet platform to increase drop starting at 0.01% to 2.25%.
The following data points for drop will be tested, 0.01%, 0.1%, 0.5%, 0.75%, 1%, 1.25%, 1.5%, 1.75%, 2.0%, 2.25%.
2. Establish call between Clients via MCU at 768Kbps.
3. Observe call quality.



Observed Video Defects:

Ghost images, frame freezing and jerky video. Recovery time on the NIC device grew quite large quickly. The Appliance device seemed to fair better overall.

Conclusions:

The Appliance device seemed to fair better overall. The MCU seemed to help in total of errored seconds but helped little in a role as a replay buffer for recovery. Recovery times were similar to point-to-point connections.

E8. Objective:

The purpose of this test is to see the effects of jitter using an MCU for one-way network problems.

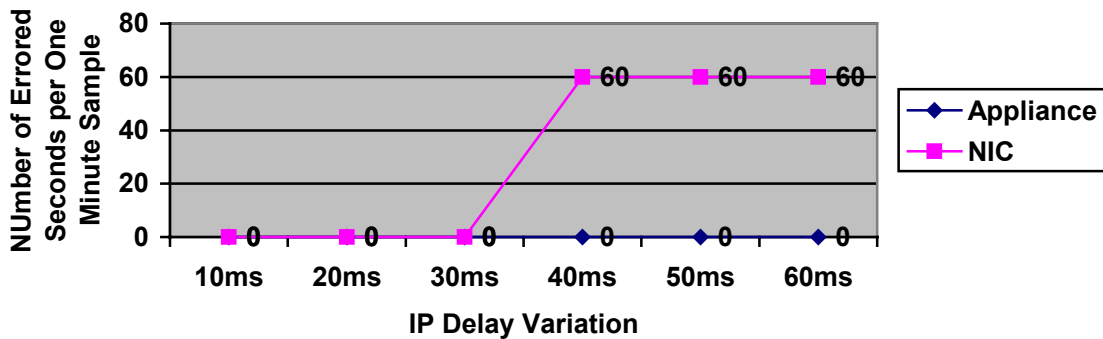
E9. Test Set Up.

Test set up is the same as section E2.

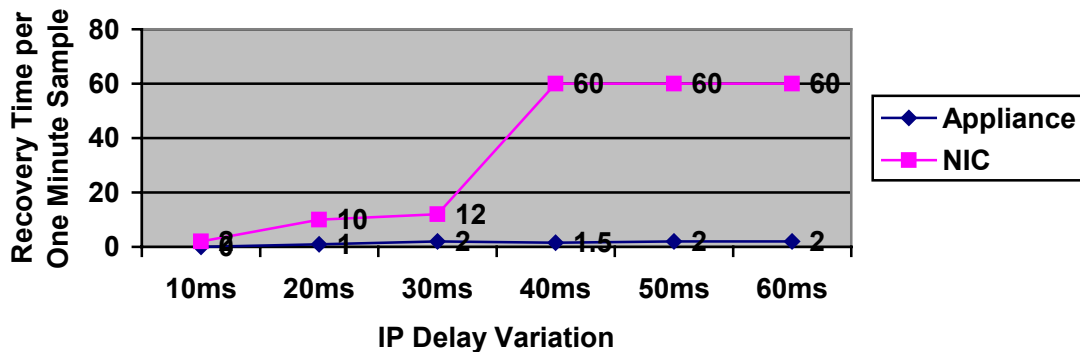
E10. Test Execution.

1. Configure Nistnet platform to increase jitter by 10ms.
2. Establish call between Clients at 768Kbps.
- 3.Observe call quality
- 4.Repeat calls with increasing jitter every iteration by 10ms until call quality deteriorates.

One Way Jitter with MCU



One Way Jitter With MCU



Observed Video Defects:

Tiling, Ghost Image, Image Smearing and Frame Freezing.

Conclusions:

The observed video events at first gleaning are somewhat surprising. Since the errored seconds are sampled in a still scene it appears the MCU has somewhat of a smoothing effect. However when motion is introduced the NIC experiences near total failure. It is interesting to note the failure occurs in close proximity to the theoretical bound of 33ms for 30 frames per second which was requested conference frame rate.

F: Two Way Network Failure Via MCU Test

F1. Objective.

The objective of is to examine the effects of two way latency network problems when the clients are connected via an MCU.

F2. Test Set up.

It is the same as test E2.

F3. Test Execution.

1. Configure Nistnet platform to increase latency by 10ms.
2. Establish call between Clients Through MCU at 768Kbps.
3. Observe call quality
4. Repeat calls with increasing latency every iteration by 10ms until call quality deteriorates.

F4. Test Results.

Test results were the same as reported in test E4. Latency does not appear to be an issue.

F5. Objective:

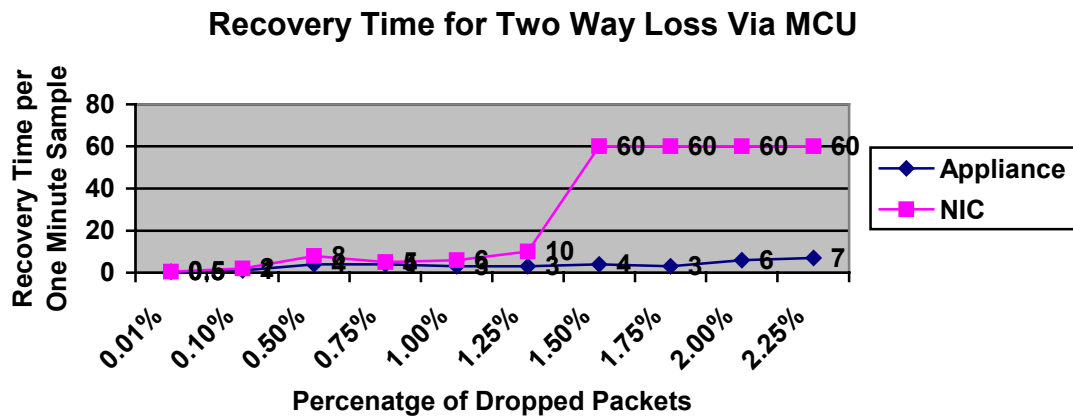
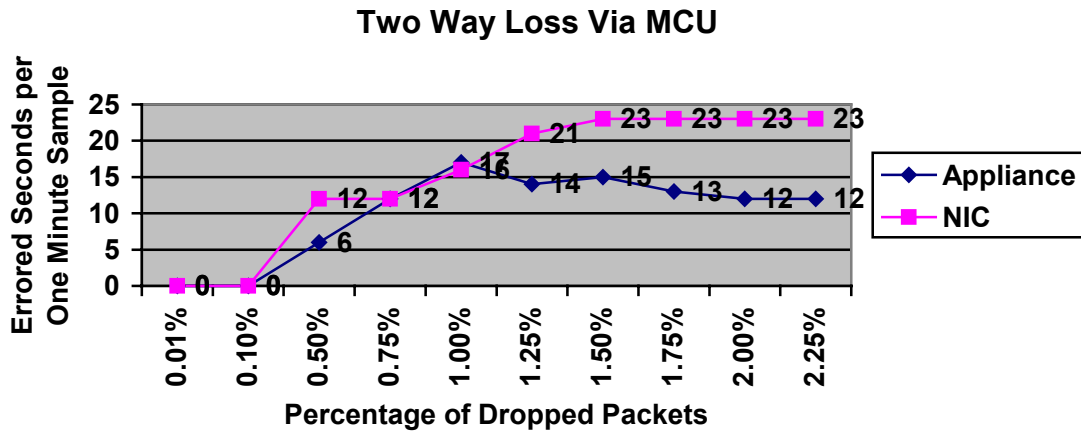
The purpose of this test is to see the effects of loss using an MCU with two way network problems.

F6. Test Set Up.

Test set up is the same as section E2.

F7. Test Execution.

1. Configure Nistnet platform to increase drop starting at 0.01% to 2.25%.
The following data points for drop will be tested, 0.01%, 0.1%, 0.5%, 0.75%, 1%, 1.5%, 1.75%, 2.0%, 2.25%.
2. Establish call between Clients via MCU at 768Kbps.
3. Observe call quality.



Observed Video Defects:

Tiling, Frame Freezing, and Congestion Notification. Extreme Tiling was apparent in the Appliance unit although recovery was good.

Conclusions:

The defect occurrence and duration are similar to the one way test. Loss effects would appear to be independent of single or double duplexing of the loss.

F8. Objective:

The purpose of this test is to see the effects of jitter using an MCU for two- way network problems.

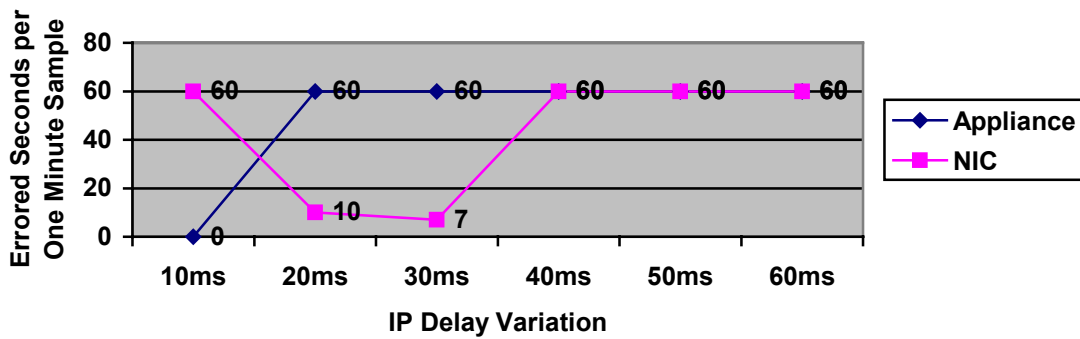
F9. Test Set Up.

Test set up is the same as section E2.

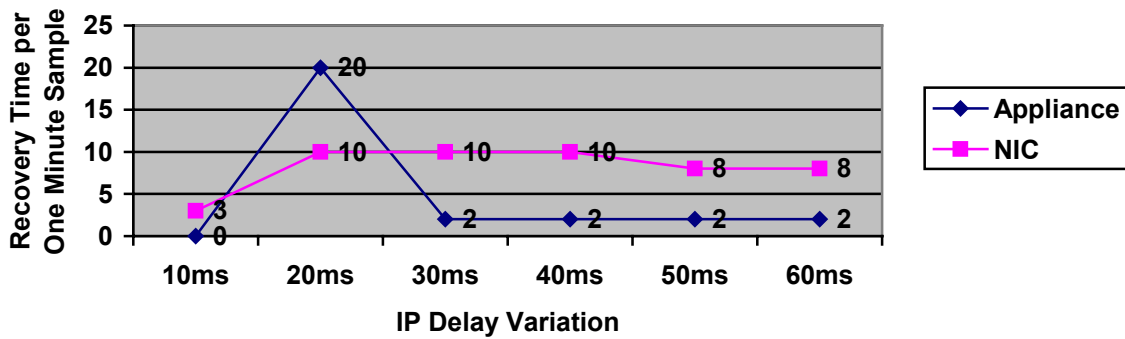
F10. Test Execution.

1. Configure Nistnet platform to increase jitter by 10ms.
2. Establish call between Clients at 768Kbps.
- 3.Observe call quality
- 4.Repeat calls with increasing jitter every iteration by 10ms until call quality deteriorates.

Two Way Jitter with MCU



Two Way Jitter with MCU Recovery Times



Observed Video Defects:

Tiling, Frame Freezing, and Missing Tiles, also there was a lot of jerky motion. Initially the NIC codec had whole minutes of missing tiles.

Conclusion:

The behavior of the clients was the most erratic of all tests so far. Both quickly went into chronic failure although recovery was relatively fast. The events were cascading together so it was difficult at times to tell when one event ended and a new one began. It would appear the MCU causes a more erratic performance when the network failure is bi-directional.

G: One-Way Cascaded MCU test.

G1. Objective:

The purpose of this test is to see the effects of latency using a cascaded MCU for one way network problems.

G2. Test Set Up.

1. Plug each client directly into the Cisco 2924XL
2. Create two separate VLANs so that each client is on a separate subnet.
2. Plug each VLAN of the Cisco 2924XL into the ports of the Nistnet device.
Note: Each Nistnet port is a different subnet.
3. Plug the MCUs into the Cisco 2924XL so that the each client and MCU pair are on different subnets and must be routed via the Nistnet router.
4. Configure a common conference on one MCU and register it with the on board GateKeeper. Register the other MCU to the same GateKeeper
5. Register both clients with the GateKeeper.

G3. Test Execution.

1. Configure Nistnet platform to increase latency by 10ms.
2. Establish call between Clients Through MCU at 768Kbps.
3. Observe call quality
4. Repeat calls with increasing latency every iteration by 10ms until call quality deteriorates.

G4. Test Results.

Test results were the same as reported in test E4. Latency does not appear to be a relevant metric.

G5. Objective:

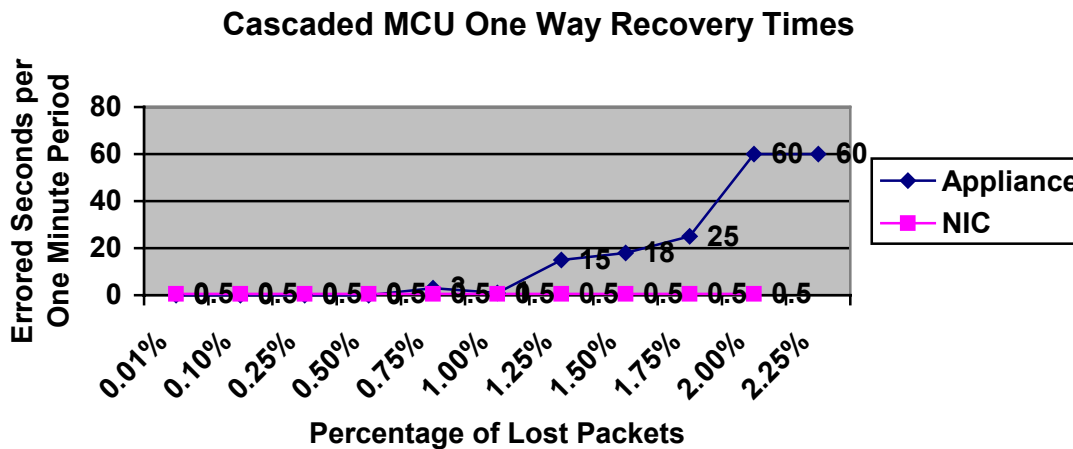
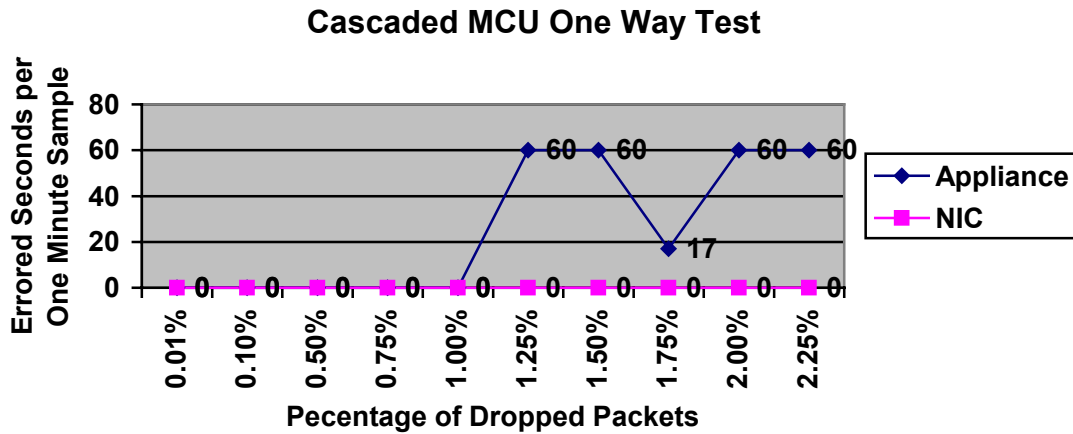
The purpose of this test is to see the effects of loss using an MCU for two way network problems. Secondly, since the errors are introduced between the Appliance and the MCU and the NIC are on the same subnet as the MCU. We wish to evaluate the MCU's role as a replay buffer.

G6. Test Set Up.

Test set up is the same as section E2.

G7. Test Execution.

1. Configure Nistnet platform to increase drop starting at 0.01% to 2.25%.
The following data points for drop will be tested, 0.01%, 0.1%, 0.5%, 0.75%, 1%, 1.25%, 1.5%, 1.75%, 2.0%, 2.25%.
2. Establish call between Clients via MCU at 768Kbps.
3. Observe call quality.



Observed Video Defects:

Congestion warning, smeared images with movement, frame freezing, and ghost images.

Conclusions:

The statistics seem to show that in terms of loss the cascaded MCUs are not quite as sensitive, however since frame freezing is difficult to perceive during still shots the reported statistics may be misleading. The more salient feature is the MCU apparently smoothes out the return path. As noted in previously a client could have a clean return path but if it has a compromised forward path the receiver will turn down the performance parameters and thus compromise the video quality. In this case it is apparent the NIC did not “hear” the distress of the Appliance.

G8. Objective:

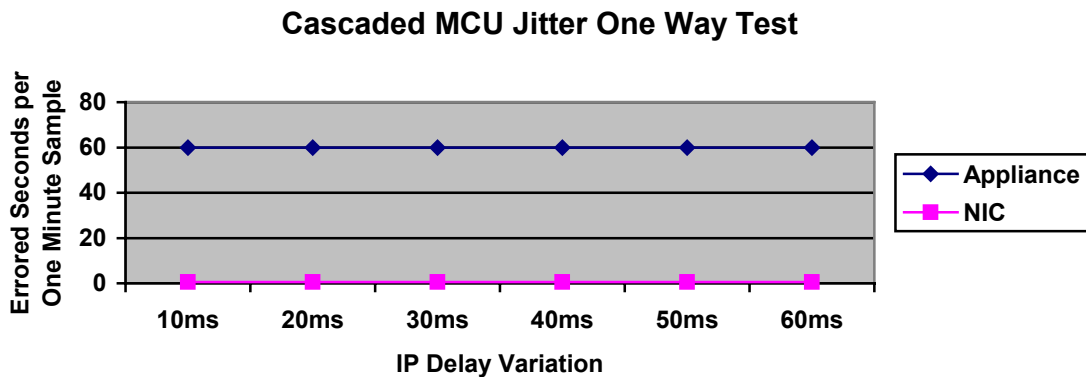
The purpose of this test is to see the effects of jitter using an MCU for one-way network problems.

G9. Test Set Up.

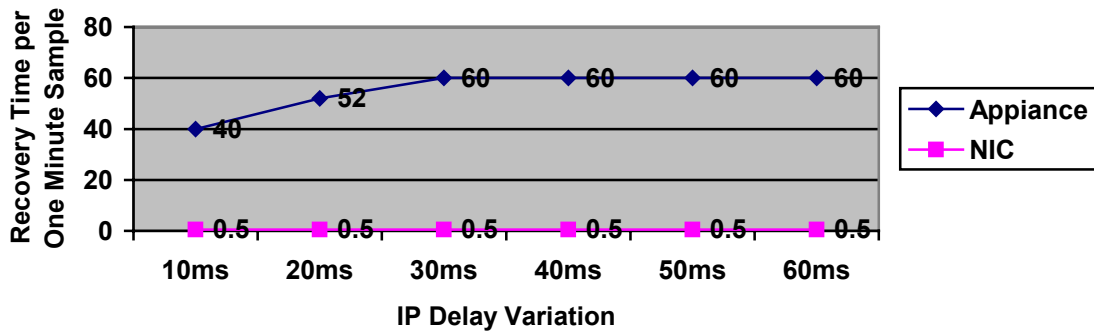
Test set up is the same as section E2.

G10. Test Execution.

1. Configure Nistnet platform to increase jitter by 10ms.
2. Establish call between Clients at 768Kbps.
3. Observe call quality
4. Repeat calls with increasing jitter every iteration by 10ms until call quality deteriorates.



Cascaded MCU Jitter One Way Test



Observed Video Defects:

The Appliance device was the receiver and behaved very badly from the very beginning. The reverse path was largely unaffected. The artifacts seen were chronic tiling and freezing rendering the video unusable.

Conclusions:

Jitter between cascaded MCUs is very bad for performance. Our hypothesis is that the MCU acts as a jitter amplifier because it is trying to transcode on the fly. When jitter hits the MCU it tries to adjust to the new speed but since the jitter produced is randomly distributed it can't lock down packet scheduling. This needs to be investigated further.

H: Two-Way Cascaded MCU Tests.

H1. Objective:

The purpose of this test is to see the effects of latency using a cascaded MCU for two-way network problems.

H2. Test Set Up.

3. Plug each client directly into the Cisco 2924XL
4. Create two separate VLANs so that each client is on a separate subnet.
2. Plug each VLAN of the Cisco 2924XL into the ports of the Nistnet device.

Note: each Nistnet port is a different subnet.

3. Plug the MCUs into the Cisco 2924XL so that the each client and MCU pair are on different subnets and must be routed via the Nistnet router.
4. Configure a common conference on one MCU and register it with the on board GateKeeper. Register the other MCU to the same GateKeeper
5. Register both clients with the GateKeeper.

H3. Test Execution.

1. Configure Nistnet platform to increase latency by 10ms.
2. Establish call between Clients Through MCU at 768Kbps.
3. Observe call quality

4.Repeat calls with increasing latency every iteration by 10ms until call quality deteriorates.

H4. Test Results.

Test results were the same as reported in test E4. Latency does not appear to be a relevant metric.

H5. Objective:

The purpose of this test is to see the effects of loss using an MCU for two-way network problems. Secondly, since the errors are introduced between the Appliance and the MCU and the NIC is on the same subnet as the MCU. We wish to evaluate the MCU's role as a replay buffer.

H6. Test Set Up.

Test set up is the same as section E2.

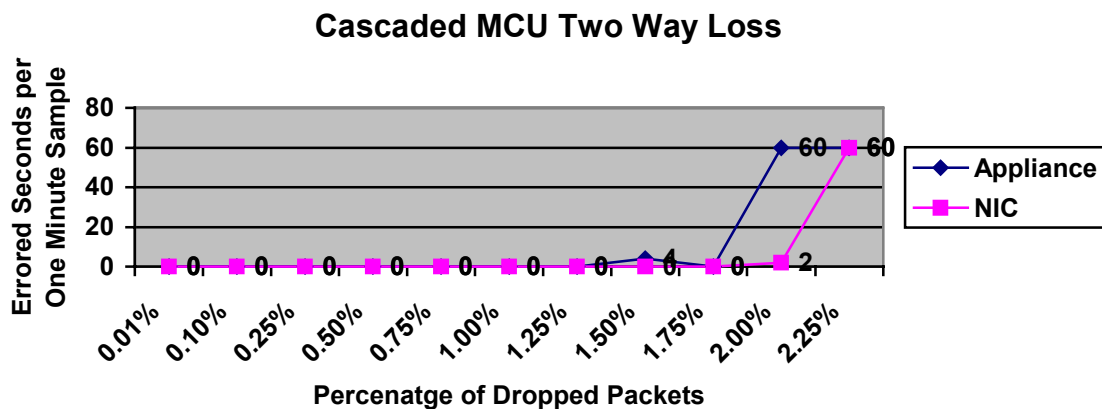
H7. Test Execution.

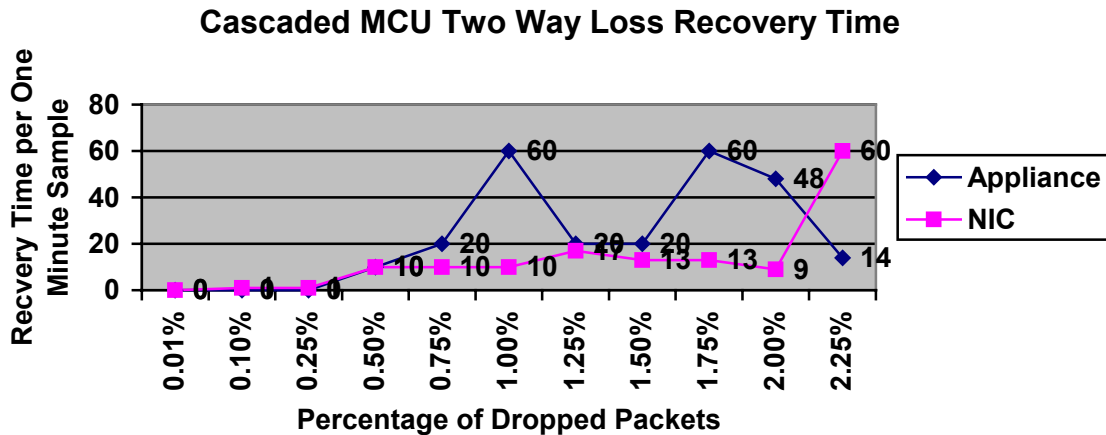
1. Configure Nistnet platform to increase drop starting at 0.01% to 2.25%.

The following data points for drop will be tested, 0.01%, 0.1%, 0.5%, 0.75%, 1%, 1.25%, 1.5%, 1.75%, 2.0%, 2.25%.

2. Establish call between Clients via MCU at 768Kbps.

3. Observe call quality.





Observed Video Defects:

Audio drop out occurs at 1.5% loss. Video has a surreal flowing effect as it deteriorates.

Conclusion:

Cascading in the packet loss scenario is rather erratic although for the still scene tests it is much more tolerant of loss than point-to-point connections.

H8. Objective:

The purpose of this test is to see the effects of jitter using an MCU for two-way network problems.

H9. Test Set Up.

Test set up is the same as section E2.

G20. Test Execution.

1. Configure Nistnet platform to increase jitter by 10ms.
2. Establish call between Clients at 768Kbps.
3. Observe call quality
4. Repeat calls with increasing jitter every iteration by 10ms until call quality deteriorates.

Observed Video Defects:

We did not graph the errored seconds or recovery times. There was chronic tiling, frame freezing and dropped tiles. The rate of error changed somewhat but errors never cleared completely. Even at 10ms jitter the video was not usable.

Conclusion:

The failure was quite bad for any jitter. As stated in previous sections the MCU not only does not help. It appears to aggravate the negative effects of jitter.

IV. Recommendations.

We have tested a variety of scenarios to better understand the operational relationships and bounds for successful H.323 video conferencing. H.323 attempts to enforce its own network defect detection and attempts to adjust accordingly by backing off frame rates and consequently bandwidth. These back off mechanisms not only help during network fault detection but also improve performance during normal operation. It uses a VBR like behavior to reduce latency [2]. Additionally, it uses RTP and RTCP to protect audio streams [3,4] thus creating a deterministic bit rate. The work started here should help provide a framework for seeking network layer solutions as well as spawn new areas for further investigation. The work presented here is by no means complete.

Our QoS recommendations are to keep jitter below 30ms and loss below 1% for normal operation. By normal operation we mean point-to-point connections and multiple clients connected to a single MCU. For the Megaconference the requirements are even more stringent. IPDV must be at an absolute minimum, it must be well below 10ms. Several approaches are being discussed.

First, traffic shaping may be desirable for cascaded MCUs, however it may need to be on a per flow basis to be effective e.g. it will only be effective on the flow connecting one MCU to another [5]. How deployable this is at the network layer is questionable. Abilene probably cannot currently support it due to GSR line card limitations. However, it is worth investigating. This solution also is dependent on participating Gigapop's ability to support it as well as the type of equipment deployed for the Member's Meeting in Austin. Another method is place play back buffer devices at the end points. The correct performance metrics must be validated experimentally but there is an existing Network Calculus proof supporting the viability of this approach [5]. There are off the shelf devices whose makers claim said devices have the necessary features.

Another option may be to turn off transcoding. Again, this must be verified experimentally. The MCU vendors must be consulted to explore this as a factual solution.

The third option would be to co-locate all cascaded MCUs. This would minimize the probability of jitter but with two constraints. First, the switch or switches must be able to handle the load. Secondly, jitter reduction is at the expense of having a single point of failure.

V. Acknowledgments.

ITEC-Ohio would like to thank Spirent Communications and Polycom for their generous support without which these tests and many other experiments would not be possible.

I wish to also thank our research assistant Anjaneya Prasad Calyam for his help in the testing and academic research phases of these experiments.

Bibliography

- [1] S. Wolf. Measuring the End-to-End Performance of Digital Video Systems. IEEE Transactions on Broadcasting, Vol. 43, No.3, pp 320-328. September 1997.
- [2] M. Baldi, Y. Ofek. End-to-end Delay of Videoconferencing over Packet Switched Networks. IEEE, February 1998. Pp.1084-1092.
- [3] H. ElGebaly. Charecterization of Multimedia Streams of an H.323 Terminal. Intel Technology Journal, August 2, 2001.
- [4] H. ElGebaly. Reactive Mechanisms for Recovering Audio Performance in MultiMedia Conferencing Over Packet Switched Networks. Intel Technology Journal, Q3, 1999.
- [5] J.-Y. Le Boudec, P. Thiran. Network Calculus, A Theory of Deterministic Queuing Systems for the Internet. Springer-Verlag, Heidelberg Germany. July 2001.

Appendix A

Encode/Decode Time Measurement of a Polycom 512 Unit

-Anjaneya Prasad Calyam

3rd August 2001

Objective:

To measure the time it takes for two Videoconferencing Units to encode and decode a still video and a Metronome generated pulsating audio when calls are made between the two units in 3 modes: 'point-to-point', 'via an MCU' and 'Via 2 cascaded MCUs'.

Setup:

1. Two Videoconferencing Units were connected to a switch. A Set of experiments were conducted using two Polycom 512s and another set of experiments were conducted using one Polycom and one Vcon. The results were comparable in both the cases. The MCUs used in the experiments were Radvision MCUs that were connected to the same switch to which the two Videoconferencing units were connected, as shown in Fig.1
2. Calls were made in 3 modes:
 - (a) Point to point
 - (b) Via an MCU
 - (c) Via 2 cascaded MCUs
3. The dialing speeds for the calls were: 256kbps, 384kbps, 512kbps and 768kbps
4. Metronome Dial setting: 113 with a constant volume in all the test cases. The setting of 113 was chosen so that the trace on the oscilloscope was intelligible.
5. Microphone of Videoconferencing Unit 2 (shown in Fig. 1) was muted and the Metronome output was fed to the Videoconferencing Unit 1.
6. An oscilloscope was used to generate traces of the metronome output before and after the encode/decode through the two Videoconferencing Units.
7. The assumption in this experiment is that the propagation delay and delay of the signal through the switch are negligible.

Major Findings:

1. The Encode + Decode delay in a point-to-point setting is ~240ms. This delay is independent of the dialing speed.
2. Each MCU or each cascaded MCU in a Videoconference adds ~120-200ms extra delay. As the dialing speed increases, the MCU delay increases.

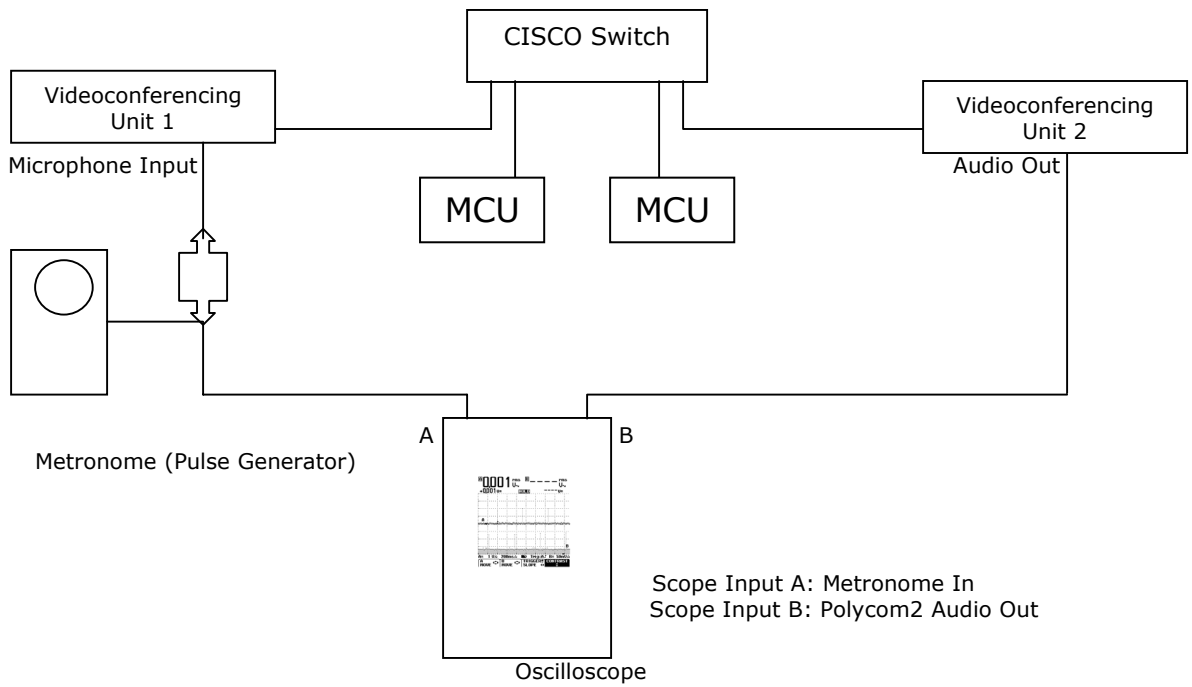


Fig. 1 EXPERIMENT SETUP

Encode/Decode Test Results

The waveforms for the 256kbps and 512kbps have not been recorded in this document. But the results in those settings were comparable to the results shown below.

Setup:	384K Point-to-Point
Metronome Setting:	113
Waveform:	<p>The waveform shows two channels, A and B. Channel A displays a series of pulses, and Channel B displays a series of dashes. A scale bar indicates a duration of 240ms. The display shows '0057' and '+0001U'. The bottom of the screen shows settings: A= 1 U/A, 200ms/A, Trig:AJ, B= 50mU/A. Controls for MOVE, TRIGGER SLOPE, and CONTRAST are visible.</p>
Comment:	240ms Delay

Setup:	384K One MCU
Metronome Setting:	113
Waveform:	<p>The waveform shows two channels, A and B. Channel A displays a series of pulses, and Channel B displays a series of dashes. The display shows '0001' and '+0001U'. The bottom of the screen shows settings: A= 1 U/A, 200ms/A, Trig:AJ, B= 50mU/A. Controls for MOVE, TRIGGER SLOPE, and CONTRAST are visible.</p>
Comment:	360ms Delay

Setup:	384K Cascaded MCU
Metronome Setting:	113
Waveform:	<p>The screenshot displays two waveforms, A and B, on a grid. Channel A shows a series of pulses with a period of approximately 200ms. Channel B shows a similar series of pulses, but with a significant delay relative to channel A. The top display shows '0.027 rms' for channel A and '----- rms' for channel B. The bottom display shows 'A= 1 U/A 200ms/A Trig:A/B B= 50mU/A' and control buttons for 'MOVE', 'TRIGGER SLOPE', and 'CONTRAST'.</p>
Comment:	500ms Delay

Setup:	768K Point-to-Point
Metronome Setting:	113
Waveform:	<p>The screenshot displays two waveforms, A and B, on a grid. Channel A shows a series of pulses with a period of approximately 200ms. Channel B shows a similar series of pulses, but with a significant delay relative to channel A. The top display shows '0.007 rms' for channel A and '----- rms' for channel B. The bottom display shows 'A= 1 U/A 200ms/A Trig:A/B B= 50mU/A' and control buttons for 'MOVE', 'TRIGGER SLOPE', and 'CONTRAST'.</p>
Comment:	240ms Delay

Setup:	768K One MCU
Metronome Setting:	113
Waveform:	<p>A: 0.0000 rms U_~ +0.0001 U_m HOLD</p> <p>B: 0L rms U_~ - 0L U_m</p> <p>A: 1 U/A 200ms/A Trig:A J B: 50mU/A</p> <p>A MOVE B MOVE TRIGGER SLOPE CONTRAST</p>
Comment:	470ms Delay

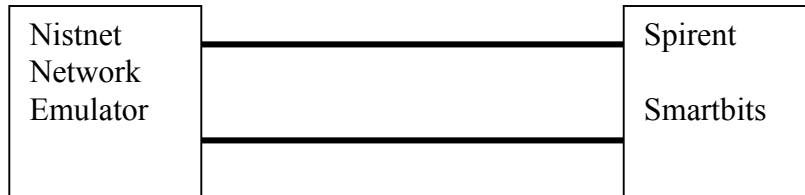
Setup:	768K Cascaded MCU
Metronome Setting:	113
Waveform:	<p>A: 0.035 rms U_~ +0.0001 U_m HOLD</p> <p>B: --- rms U_~ --- U_m</p> <p>A: 1 U/A 200ms/A Trig:A J B: 50mU/A</p> <p>A MOVE B MOVE TRIGGER SLOPE CONTRAST</p>
Comment:	630ms Delay

Note:

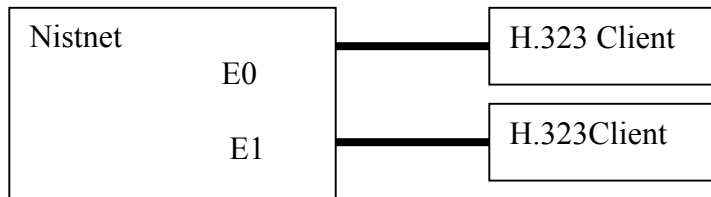
The experiment described above was conducted for Dr. Bob Dixon's presentation titled- "First-Ever Multi-Site Internet Musical Concert", at the Internet2 Members Meeting, March 2001, Washington, DC.

Appendix B Network Diagrams

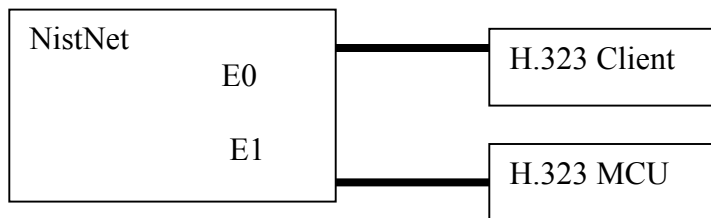
Test Series A “Network Emulator Verification Tests”.



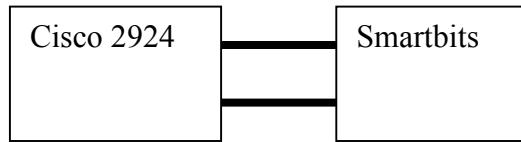
Test Series B “ Point-to-Point Client Tests”.



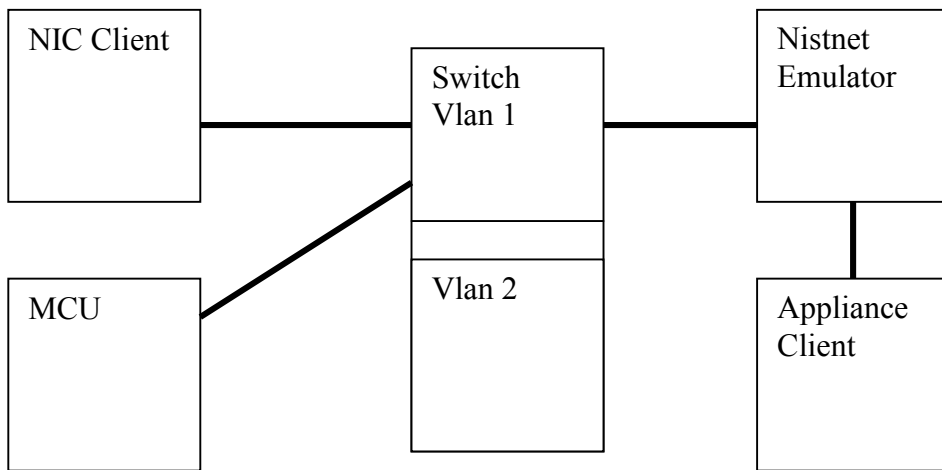
Test Series C “Client to MCU Tests”.



Test Series D “Switch Propagation Delay Test”.



Test Series E & F “Multiple Clients Connected via MCU Test”.



Test Series G & H “Cascaded MCU Tests”.

