



Application Sharing in the Virtual Classroom

Testing Framework

This document is for the actual testing of the different application and screen sharing solutions.

Gerhold Benjamin Kooper
11/19/2010

1. Introduction

This document is to outline the testing factors and issues to test on the different screen and applications sharing tools. It would also outline how these tests would be done.

In this document the attributes to be tested for the analysis discussed in the Analysis Framework is discussed. The document would also outline how to tests the different attributes and the software and tools used for each of these tests.

2. The Testing Environment Setup

The setup under which the testing would be done is shown below.

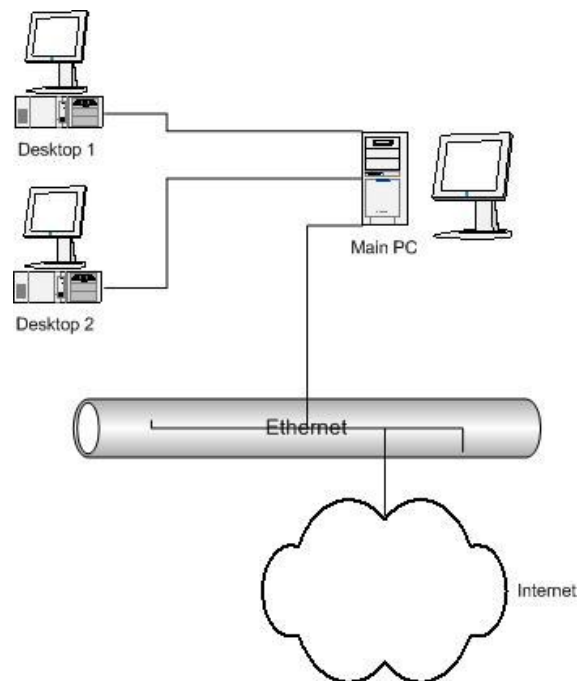


Figure 1: Environment Setup

Main PC

- Intel Quad Processor, 2.66GHz x 4
- 3 GiB Physical Memory, 1GiB Virtual Memory
- Three NIC: eth0 1 Gbps, eth1?, eth2 ?
- Ubuntu 10.04 Lucid, 2.6.32-25-generic kernel

The desktops are able to access internet by using the masquerade function of the iptables software. The main pc can also manipulate the bandwidth between it and the Desktops by using the iproute kernel module script.

2 Nanoware PCs

- Intel(R) Atom 1.60GHZ x2
- 2 GiB Physical Memory
- 1 NIC ?
- Ubuntu 9.04 , Windows Vista Home Edition

3. Pre Test

There are some tests or investigation to be done before doing the testing of the application and screen sharing systems. These tests would help in setting the tests criteria.

3.1.Network Mapping

This investigation is to determine the physical network layout to address issues regarding hosted solutions on different continents.

This may be done by using route trace which would show the different routers the IP packet passes to get to the server. As per the IP protocol design, the same route may not be followed at all times, so this has to be taken into account. The IP address of each router can then be entered into the website www.ip2location.com to get the geographical location.

The network mapping could also be determined more accurately by obtaining the network map from the Internet Service Providers in South Africa and Namibia.

3.2.Proxy Affect on Latency

The ping test provides the network latency to the host being pinged. This however may not reflect the same latency on the application protocol layer. The HTTP protocol data has to pass through the proxy server and add delay.

Have also to find out whether there are policies by the Rhodes IT Department that throttle network traffic of certain nature such as streaming audio/video. Also to see if ssh traffic is slowed down by throttling.

3.3.Computer Performance

Initial trials of application sharing system on the Atom Processor show high processor activity when even just moving the mouse around or activities causing frequent changes on screen. Have to investigate significance of processor speed and type.

4. Software

This part is to test the issues concerning the software itself with assumption that the bandwidth and network connection is ideal and has no problems.

4.1. Latency

In the screen and application session systems, it is the delay between when the presenter initiates an activity, such as mouse click or cursor move on the screen and when a participants or student observes the same action on their computer.

One way of testing this would be by observation as both desktop computers are on the same table. One can perhaps click on one desktop and watch how quickly the action responds on the other desktop.

To have amore accurate measure, screen recording software can be used. The shared application can be run on the same screen together with Wireshark to see the packets leave of the application sharing software. This would allow measuring the delay between when a mouse click on a menu of the shared application and the moment the IP packet is send onto the network. This measurement can be to milliseconds.

4.2. Smoothness

A smooth experience is one that flows easily with a minimum of choppiness or flicker. In screen and application sharing, what underlies a smooth experience is often a high frame rate.

This can be tested by watching a video clip in the shared screen or application and observer the smoothness of the video playback. The video should be encoded at different frame rates such as:

- 25 frames per second used in standard video and
- 5 frames per second being used in CCTV security cameras

An ideal performance would be to see a smooth picture on the viewers screen encoded at 25 frames per sec or higher.

It should be checked and noted whether the software allows the user to change the frame rate. The frame rate change would directly affect the bandwidth use.

The frame rate when used with compression may have a negative correlation where the higher the frequency of update, the less efficient it is as the endpoint may not be able to compress and decompress the updates fast enough.

4.3. Fidelity

Fidelity denotes how accurate a copy is to its source. When screen sharing, presentation with perfect fidelity looks identical to both the presenter and attendee.

This can be tested by observation on both desktop computers. One can make observations with different settings of screen resolution and background inclusion or exclusion. The resolution changes would however directly affect the bandwidth use.

As the fidelity can be changed by changing the screen resolution, it should be noted whether the software allows user to change resolution.

4.4. Partial Screen Updates

Some software use techniques to minimize the amount of traffic to be send over the network for updating the screen and would try to only send parts of the screen that have changed. This is also known as Run Length Encoding (RLE).

The effectiveness of this would change as the proportion of screen has to change. The test here is to see when having windows on the screen needing frequent updates, such as a video clip, what is the optimum size in terms of proportion considering resolution one can have. It can be measured in terms of $\frac{1}{2}$, $\frac{1}{4}$ or so.

4.5. Compression

Many of the software use compression to further reduce bandwidth usage and improve application performance.

This test is to see the affect of compression in improving performance of the application sharing solution. Compression rate on images of different formats may be higher then other, but this test is to see the significance.

5. Network

This part is to test the issues concerning the bandwidth with assumption that the software is ideal and has no problems.

5.1. Protocols

This would be to see the protocols used by each system.

5.2. Physical International Connection

In the case of web based application, this test would check if one physical connection is used. It is also to see if there is a difference in performance of the network when connecting at different times.

5.3. Bandwidth

In this test, the bandwidth used by each system and protocol would be measured. The bandwidth would also be manipulated to test system performance at different speeds.

The bandwidth usages of the application would be directly affected by some settings in the software such as screen resolutions, frame update rate and compression enabling.

The solutions can be tested at different speeds such as:

- 10 mbps, 5 mbps, 1 mbps, 512 kbps, 128 kbps, 56 kbps

In addition to the bandwidth, network latency would be added such as 500 ms, 300 ms, 100 ms and 10 ms. Added to this would be random packet loss and random connection loss to emulate unreliable networks. Combination of these various attributes one may have a test situation such as 5 mbps bandwidth, 300 ms latency, 5 seconds connection loss and 5 % packet loss.

Tests should be done from the UNAM connection to have the situation of the connection for more relevant testing.

The Wide Area Network Emulator (WANem) provides this functionality.

6. Testing Candidates

The Application Screen Sharing Solutions to be tested can be categorized as follows:

- Display Protocol
 - NX
 - FreeNX
 - VNC/RFB
 - TightVNC, RealVNC, iTALC
 - RDP
 - Windows Remote Desktop
 - HTTP
 - FreeNX, TeamViewer, Vyew, LogMeIn, Adobe Acrobat CONNECTNOW, Mikogo, Skyfex, Shareitnow, Yuuguu
- Transport Protocol
 - UDP
 - TeamViewer
 -
- Architecture
 - Replicated
 - Flexible JAMM
- Platform
 - Windows
 - Windows Remote Desktop, TightVNC, TeamViewer
 - Linux
 - FreeNX, TightVNC, RealVNC, iTALC
- Offered
 - Free
 - FreeNX, TightVNC, RealVNC, iTALC, TeamViewer, Mikogo
 - Limited
 - CrossLoops
 - Trail
 - GoToMeeting, Glance, eBLVD, BeamYourScreen, Gatherplace, SLLight, AT&T Connect, Anyplace Control, Assemb'Live, Elluminate, Fuze Meeting, Hear Me, Toktumi Desktop Sharing
-

7. Remote Desktop Full Screen Update Tests

Full screen update tests would check the maximum time it would takes the solution to update the screen. **The test would also see the compression efficiency.** For this test the following would be measured:

- ✓ Data size transferred
- ✓ Time it takes to do completed data transfer
- ✓ Average speed in mbps
- ✓ Average packets send per second
- ✓ Average packets size
- ✓ Time it takes to complete screen update

The test would first be done with default settings and later with solution tuned for best performance.

Benchmark: Bandwidth usage for resolution, window size and frame rate combination.

Below is an example of a table for testing the bandwidth usage for different solution with attributes below:

- 1024x768 pixels
- Windows sizes taken as proportion of screen such as full, ½ or ¼ of screen
- Video encoded ranging from 25 to 5 frames per second
- Network latency for example 10 ms, 100 ms, 300 ms and 500 ms
- Some solution allow different compression and encoding such as RLE, ZLIB, Hextile

Solution	: [Product Name]
Resolution	: [1024x768, 1280x1024]
Color Dept	: [8 bit, 16 bit, 32 bit]
Display Protocol	: [VNC, HTTP, AIP, X, NX, etc]
Transport Protocol	: [UDP, TCP]
Display Encoding	: [Low-level Graphics, High-level Graphics, 2D draw Primitives]
Screen Update	: [Lazy, Eager, Server Push, Client Pull]
Encoding	: [RLE, ZLIB, etc]
Network Latency	: [10 ms, 100 ms, 300 ms, 500 ms]

Notes:

- ✓ When doing a testing, the screen recording has to be done at the same time as capturing the network packets.
- ✓ The IP packets have to be captured closed to the client thereby needing to connect both client and main pc to a hub acting device. Also to make sure the network emulation is completed on packets captured.
- ✓ The network stats have to be compiled immediately after capture to account dropped packets and other network errors.
- ✓ Screen recording software should capture at a higher frame rate.
- ✓ Ping test done to measure network quality by using the flood option wit packet size 1440 bytes. The packet size is chosen as initial screen sharing test done show the largest packet size where 1448 bytes. The following are the results of ping test for two servers in Namibia:
 - Command eneterd was `sudo ping <server> -s 1440 -f`
 - elc.nolnet.edu.na Hosted by Gijima:

```
PING elc.nolnet.edu.na (96.44.128.207) 1440(1468) bytes of data.  
.....  
3674 packets transmitted, 3507 received, 4% packet loss,  
time 50904ms  
rtt min/avg/max/mdev = 57.637/59.393/93.349/1.263 ms, pipe  
7, ipg/ewma 13.859/60.527 ms
```
 - wwwscan.unam.na which is the mail server at UNAM campus:

```
PING unammail.unam.na (196.20.25.200) 1440(1468) bytes of data.  
.....  
3237 packets transmitted, 3059 received, 5% packet loss,  
time 46564ms  
rtt min/avg/max/mdev = 112.225/125.039/197.240/10.748 ms,  
pipe 15, ipg/ewma 14.389/120.921 ms
```
- ✓ Determine for each solution whether only the part of the screen that changes is updated.

7.1. Methodology

7.1.1. Software

- NetEm Network Emulator: Based on the iproute Linux kernel. Control network latency and transfer rate.
- WireShark. Capture traffic and analyze
- MS Excel. Used to calculate network data sums for each picture update.

- Presentation Programs such as MS PowerPoint and Open Office Presentation. To display pictures using full screen.

7.1.2. Steps or Procedure

7.1.2.1. Network

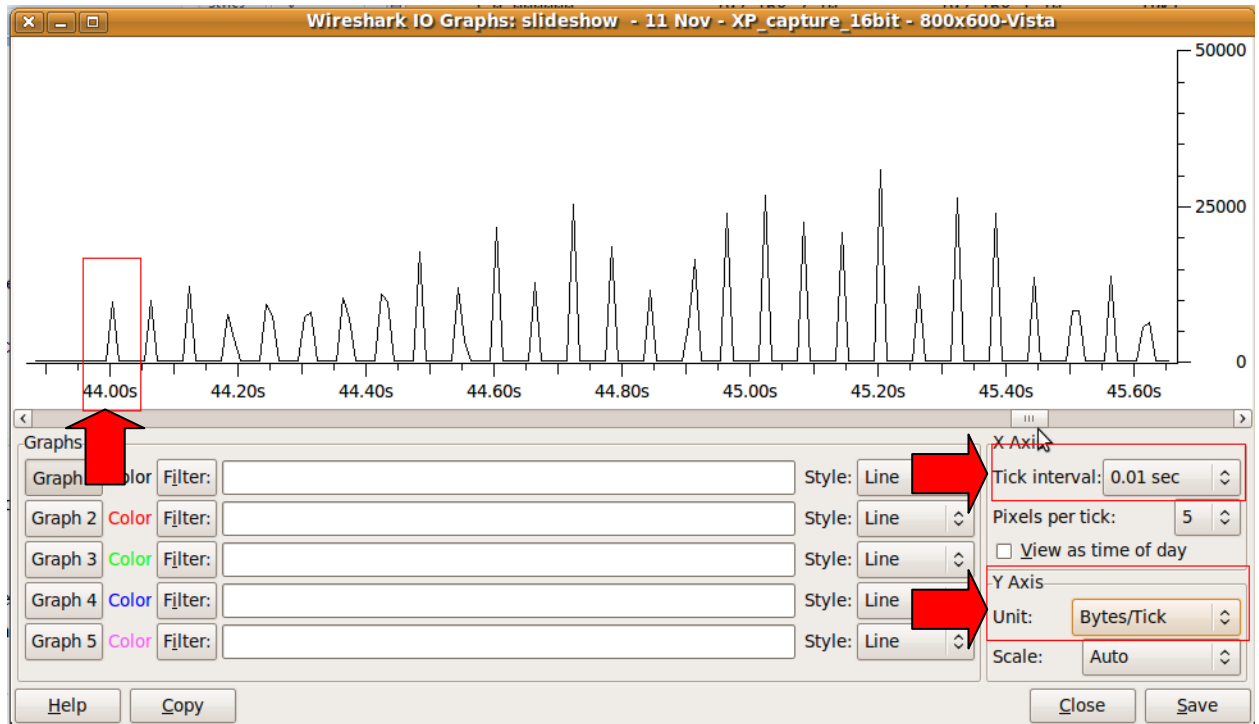
The steps or procedures for the full screen update tests are outlined below.

- ✓ First the solution being tested is started on the pc to act as the display server.
- ✓ The screen resolution is set at the necessary places.
- ✓ The latency is set at the Main PC using the Traffic Control command as shown here:

```
sudo tc qdisc add dev eth1 root netem delay 60ms 5ms
```

 - Above eth1 would be the interface to which the display server is connected, so it can also be eth2 or eth0
 - netem is the Queuing Discipline used in this case as to the iproute Linux kernel module
 - For this test 60ms with variation of 5 ms is set for network latency as this is what the latency is more or less to Namibia. Servers such as www.telecom.na, www.mweb.com.na and elc.nolnet.edu.na were pinged. Results were sometimes about 70ms and 50ms.
- ✓ The client to be used for the remote session is connected to the display server.
- ✓ A presentation is started on the display server with full screen images. MS PowerPoint is used for windows and Open Office Presentation for Ubuntu. Same images are used in same order.
- ✓ WireShark is started to capture packets to and from the display server.

- ✓ After all slideshow is done and captured packets are saved, the I/O Graph is opened from menu Statistics->IO Graph. Example shown below:



- ✓ The tick interval is changed to 0.01 sec and the Y units to Bytes/Tick. This allows for an accurate reading of the start and end times of screen updated traffic on the graph.
- ✓ The start and end times for the screen update transferred is then used to located the start and end packets numbers. Example shown below:

The screenshot shows the Wireshark interface with the packet list and details pane. The packet list shows a sequence of packets, with packet 5070 highlighted. The details pane shows the structure of packet 5070, including Ethernet II, Internet Protocol, and Transmission Control Protocol. A red arrow points to the 'Len: 1460' field in the TCP details.

No.	Time	Source	Destination	Protocol	Info
5067	43.990183	192.168.2.10	192.168.1.10	TPKT	Continuation
5068	43.990303	192.168.1.10	192.168.2.10	TCP	ms-wbt-server > 51247 [ACK] Seq
5069	44.006974	192.168.1.10	192.168.2.10	TPKT	Continuation
5070	44.007408	192.168.1.10	192.168.2.10	TPKT	Continuation
5071	44.007443	192.168.1.10	192.168.2.10	TPKT	Continuation
5072	44.007693	192.168.1.10	192.168.2.10	TPKT	Continuation
5073	44.007743	192.168.1.10	192.168.2.10	TPKT	Continuation


Details for Frame 5070 (1514 bytes on wire, 1514 bytes captured):

- Ethernet II, Src: HonHaiPr_6e:21:aa (00:22:68:6e:21:aa), Dst: D-Link_92:af:b1 (00:21:91:92:af:b1)
- Internet Protocol, Src: 192.168.1.10 (192.168.1.10), Dst: 192.168.2.10 (192.168.2.10)
- Transmission Control Protocol, Src Port: ms-wbt-server (3389), Dst Port: 51247 (51247), Seq: 4204205, A
- TPKT

- ✓ As shown above the start time from the I/O Graph is about 44.0 sec. In the screenshot above the first packet with a big payload would correspond to the start of the traffic peak which corresponds to packet number 5070. Using same steps the end packet number of the traffic peak corresponding to screen update should be determined.

- ✓ Having the start and end packet numbers for each instance of screen update, filter is applied to only view the packets captured for each particular screen update separately. The filter would be `frame.number >= 5070 and frame.number <= 6801`
- ✓ Now the traffic statistic is easily obtained by using menu Statistics->Summary. Example summary below:

Traffic	Captured	Displayed	Marked
Packets	49326	1581	0
Between first and last packet	167.942 sec	3.069 sec	
Avg. packets/sec	293.708	515.148	
Avg. packet size	857.379 bytes	835.975 bytes	
Bytes	42291095	1321676	
Avg. bytes/sec	251819.172	430650.989	
Avg. MBit/sec	2.015	3.445	



7.1.2.2. Software

- ✓ To measure the time it takes for the solution to paint the screen is measured using screen recording software. The recording is then scanned frame by frame through a video editing tool to get time difference from start of screen update to end in milliseconds.

7.2. Solution 1: Windows Remote Desktop

[About Windows Remote Desktop]

Settings

- Date : 9 November
- Application : Windows Remote Desktop on XP SP3
- Client : Terminal Server Client on Ubuntu
- Resolution : 1280x960, 16bit color dept
- Latency : 60ms ± 5ms
- Display Server : Nanoware PC
- Experience : High Speed Broadband (*Visual styles, Persistent bitmap caching*)
- Transport Protocol : TCP

Notes:

Uses interlacing so the screen gets painted twice. Half of the download already has the screen updated and second half only makes it clearer. Picture quality is hard to differentiate. The resolution above is shown when viewing the screen settings during remote session. It seems the

remote desktop uses the resolution setting of the connecting client computer.

Results

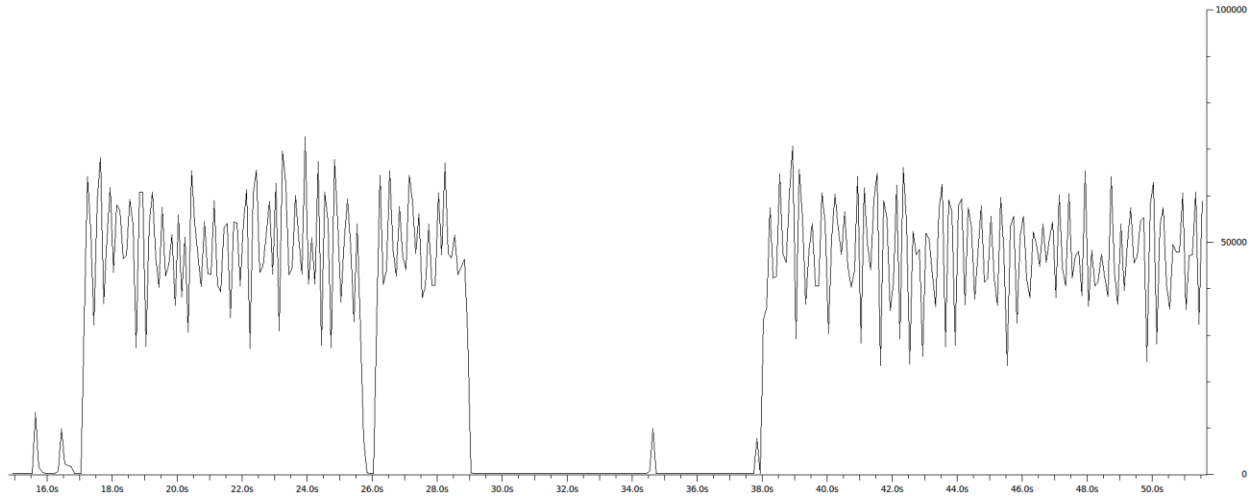


Figure 2: I/O Graph. Y=bytes

No.	Data (bytes)	Time	Avg. mbps	Avg. Packets/s	Avg. Packets Size
1	5651443 (5.39 MB)	11.8 sec	3.827	558.9	855.9 bytes
2	6753188 (6.44 MB)	14.2 sec	3.808	529.7	898.6 bytes
3	8245131 (7.86 MB)	16.8 sec	3.925	593.2	827.0 bytes
4	7590665 (7.23 MB)	15.8 sec	3.839	540.9	997.1 bytes

- Data is the amount of bytes send for the screen update for a single picture
- Time is the duration of updating a single picture
- The average kbps is the average transfer rate during the data transferred to completed update of single picture

Settings

- Date : 11 November
- Application : Windows Remote Desktop on XP SP3
- Client : Remote Desktop Client on Windows 7
- Resolution : 800x600, 16bit color dept
- Latency : 60ms ± 5ms
- Display Server : Nanoware PC
- Experience : High Speed Broadband (*Desktop composition, Visual styles*)
- Transport Protocol : TCP

- Notes:

Results

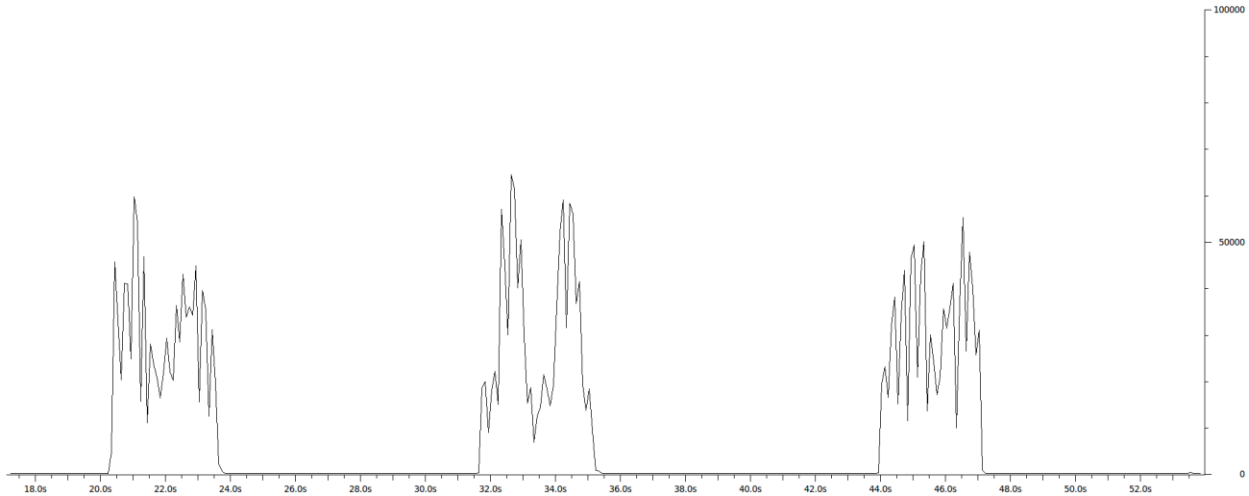


Figure 3: I/O Graph. Y= bytes

In above graph the interlacing affect is clearer. The first screen paint finishes at the point where the graph takes a dip as can be seen above.

No.	Data (bytes)	Time	Avg. mbps	Avg. Packets/s	Avg. Packets Size
1	819075 (799.9 KB)	3.4 sec	1.906	280.4	849.7 bytes
2	1612301 (1.54 MB)	3.6 sec	3.605	485.4	928.2 bytes
3	991145 (967.9 KB)	3.3 sec	2.432	352.3	862.6 bytes
4	1053450 (1.00 MB)	3.7 sec	2.295	324.3	884.5 bytes
5	965567 (942.9 KB)	3.2 sec	2.436	341.9	890.7 bytes

Settings

- Date : 11 November
- Application : Windows Remote Desktop on XP SP3
- Client : Remote Desktop Client on XP Virtual Machine on Main PC
- Resolution : 800x600, 16bit color dept
- Latency : 60ms ± 5ms
- Display Server : Nanoware PC
- Experience : High Speed Broadband (*Desktop composition, Visual styles*)
- Transport Protocol : TCP

- Notes:

Screen time is taken at different time then when network traffic was captured.

Results

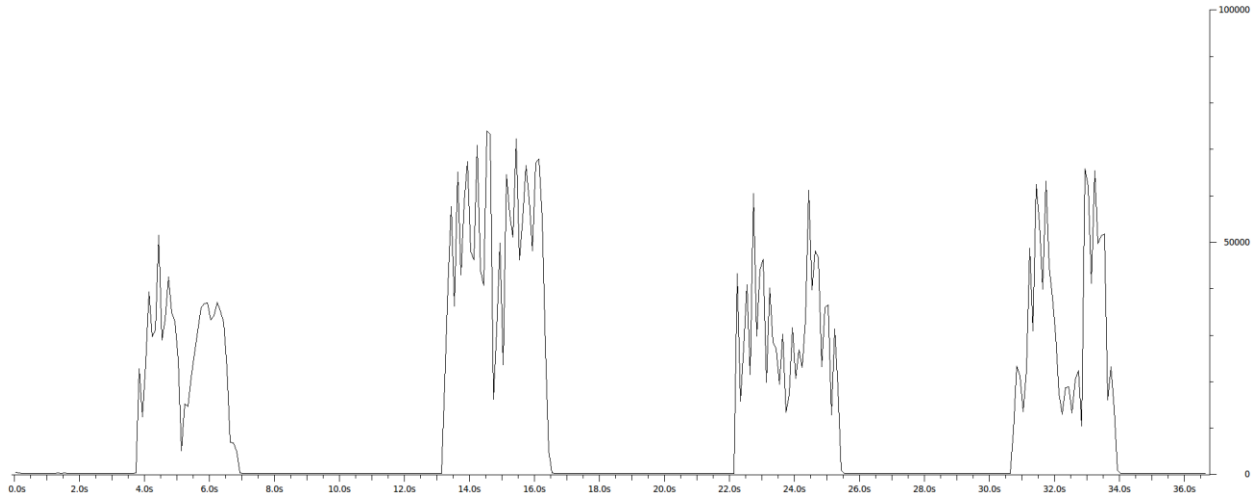


Figure 4: I/O Graph. Y=bytes

No.	Data (bytes)	Time N	Avg. mbps	Avg. Packets/s	Avg. Packets Size	Time S*
1	840466 (820.8 KB)	3.1 sec	2.199	344.3	798.2 bytes	0.801
2	1643723 (1.57 MB)	3.3 sec	4.006	577.9	866.5 bytes	0.799
3	1011359 (987.7 KB)	3.3 sec	2.473	377.7	818.3 bytes	1.0
4	1072916 (1.02 MB)	3.2 sec	2.650	394.9	838.9 bytes	0.8
5	985775 (962.7 KB)	2.9 sec	2.758	419.0	822.9 bytes	1.0

- *Time is the time it takes for first screen paint. Does not include second interlaced paint.*

7.3. Solution 2: FreeNX

FreeNX Server is the Free and GPL'd NX server implementation by Fabian Franz, based on NoMachine.com's NX technology. NX compresses the X11 data to minimize the amount of data transmitted and uses the SSH protocol to send its data. Although designed primarily to optimize X11 sessions, NX server can be configured as a proxy server to tunnel Remote Desktop Protocol (for Windows Remote Desktop Services sessions) and remote Virtual Network Computing sessions (most modern general-purpose operating system platforms), giving them some of the same speed improvements. Same screen can be shared by shadowing.

18 November 2010

Settings

- Date : 10 November 2010
- Application : FreeNX Server on Ubuntu 9.10 Desktop Version
- Client : NX Client for Linux by NoMachine
- Resolution : 1024x768
- Latency : 60ms ± 5ms
- Display Server : Nanoware PC
- Transport Protocol : TCP
- Notes:
NX Client cache cleared.

Results

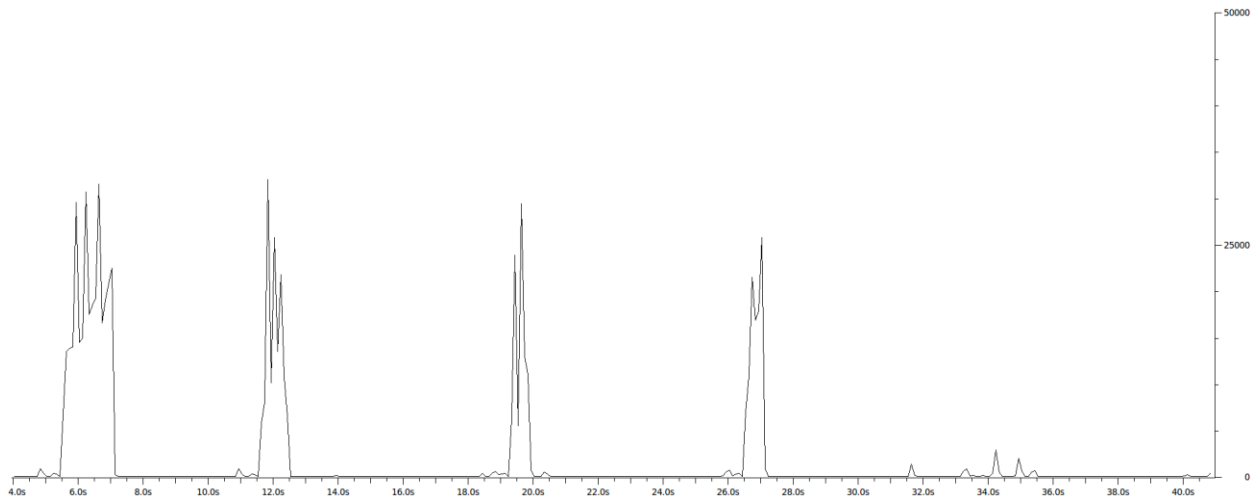


Figure 5: I/O Graph. Y = bytes

No.	Data (bytes)	Time	Avg. mbps	Avg. Packets/s	Avg. Packets Size
1	303444 (296.3 KB)	1.6 sec	1.514	215.8	877.0 bytes
2	134230 (131.1 KB)	0.8 sec	1.333	217.2	767.0 bytes
3	89 036 (86.9 KB)	0.5 sec	1.453	208.1	872.1 bytes
4	99 778 (97.4 KB)	0.5 sec	1.647	233.1	883.0 bytes

NX Manager Session log:

NXPROXY - Version 3.4.0

Copyright (C) 2001, 2010 NoMachine.

See <http://www.nomachine.com/> for more information.

Info: Proxy running in client mode with pid '7042'.

Session: Starting session at 'Wed Nov 10 19:08:05 2010'.

18 November 2010

Warning: Connected to remote version 3.3.0 with local version 3.4.0.

Info: Connection with remote proxy completed.

Info: Using ADSL link parameters 512/24/1/0.

Info: Using cache parameters 4/4096KB/16384KB/16384KB.

Info: Using pack method 'adaptive-7' with session 'gnome'.

Info: Using ZLIB data compression 1/1/32.

Info: Using ZLIB stream compression 4/4.

Info: No suitable cache file found.

Info: Forwarding X11 connections to display ':0.0'.

Info: Listening to font server connections on port '11000'.

Session: Session started at 'Wed Nov 10 19:08:05 2010'.

Info: Established X server connection.

Info: Using shared memory parameters 1/4096K.

7.4. Solution 3: TightVNC

TightVNC is an RFB protocol solution for remote screen access and sharing. It uses so-called "tight encoding" of areas, which is effectively a combination of JPEG compression and other types of encoding. Other encodings are ZRLE, RRE, CoRRE, Zlib, Hextile and CopyRect. It can be used on variety of platforms. Custom compression level of encoding used and of JPEG can be set by user.

Settings

- Date : 12 November 2010
- Application : TightVNC on Windows 7
- Client : VNC Viewer on XP
- Resolution : 800x600, 24 bit color dept
- Latency : 60ms ± 5ms
- Display Server : Nanoware PC
- Transport Protocol : TCP

- Notes:

Default setting used. Tight Encoding. Screen shows a small blue dot fro mouse click in replaying screen recording. The screen recording is taken a different time for this reading.

Results

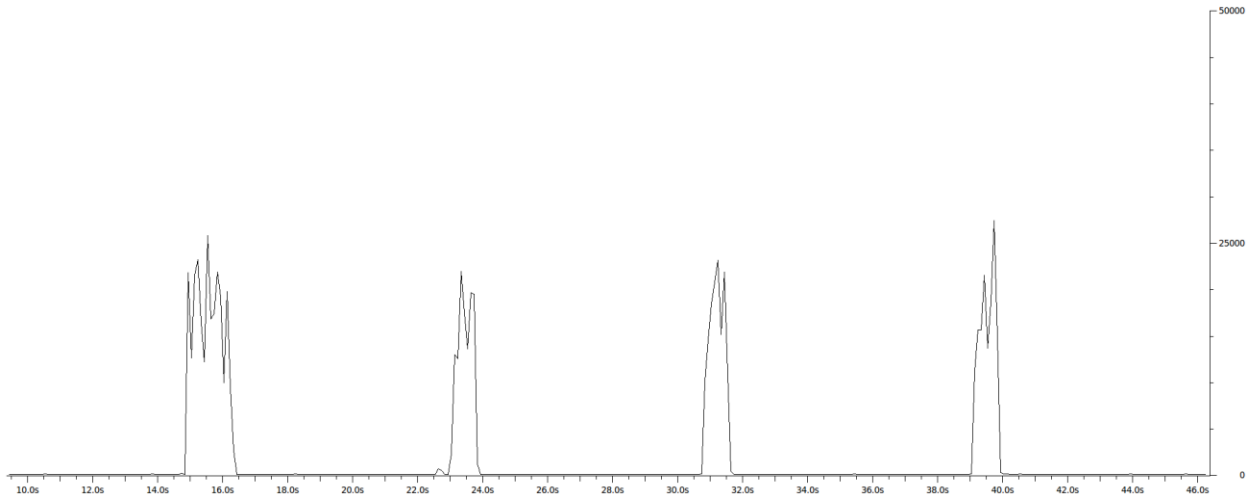


Figure 6: I/O Graph. Y = bytes

No.	Data (bytes)	Time N	Avg. mbps	Avg. Packets/s	Avg. Packets Size	Time S
1	115560 (112.9 KB)	0.9 sec	1.024	186.1	687.9 bytes	1.203
2	250505 (244.6 KB)	1.6 sec	1.219	237.8	640.7 bytes	2.406
3	120947 (118.1 KB)	1.1 sec	0.849	193.9	547.3 bytes	1.203
4	138839 (135.6 KB)	1.2 sec	0.954	194.1	614.3 bytes	1.203

7.5. Solution 4: TeamViewer

TeamViewer is a web based screen sharing solution that allows one to share your computer screen over the Internet. It supports the use of UDP to communicate screen update where possible to improve performance. It uses TCP and HTTP relayed through a central web service which would allow connections behind firewalls. It allows choosing the session to be optimized for quality or bandwidth. You can also allow remote person to control screen or not. It can be used on variety of platforms that is Windows, Linux and MAC OS.

Settings

- Date : 12 November 2010
- Application : TeamViwer on Windows 7
- Client : TeamViewer on XP
- Resolution : 800x600, 32bpp
- Latency : 60ms ± 5ms
- Display Server : Nanoware PC
- Transport Protocol : UDP

- Notes:
Quality option is set to optimize for quality.

Results

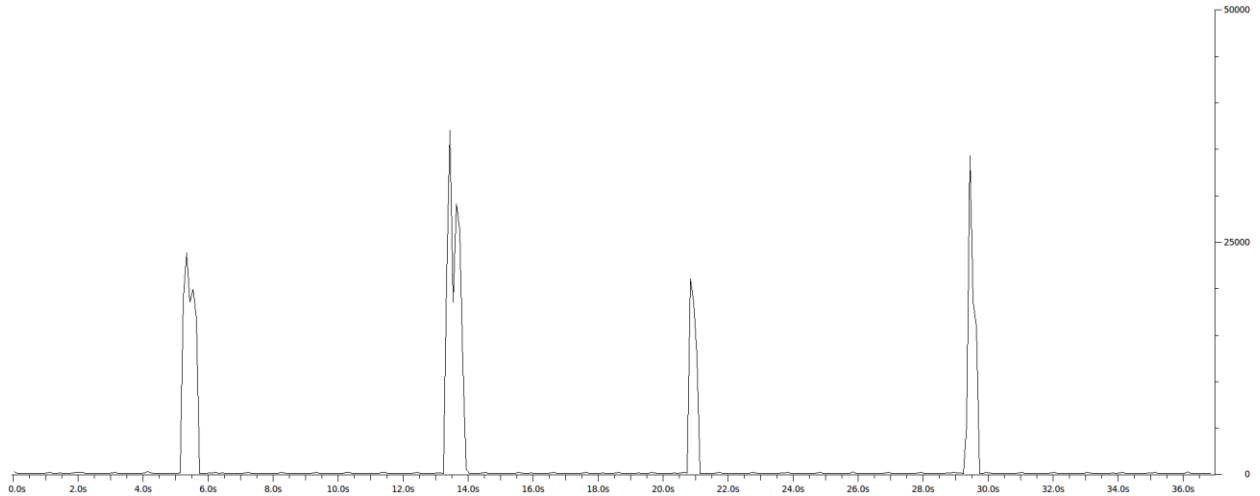


Figure 7: I/O Graph. Y=bytes.

No.	Data (bytes)	Time	Avg. mbps	Avg. Packets/s	Avg. Packets Size
1	97632 (95.3 KB)	0.3 sec	2.333	259.9	1122.2 bytes
2	143556 (140.2 KB)	0.5 sec	2.230	238.8	1167.1 bytes
3	73618 (71.9 KB)	0.2 sec	2.714	285.8	1197.4 bytes
4	64581 (63.1 KB)	0.3 sec	1.952	234.2	1041.6 bytes

Settings

- Date : 12 November 2010
- Application : TeamViwer on Windows 7
- Client : TeamViewer on elc.nolnet.edu.na
- Resolution : 800x600, 32bpp
- Latency : about 50 – 70 ms
- Display Server : Nanoware PC
- Transport Protocol : TCP
- Notes:
Quality option is set to optimize for quality. NX connection is made to elc server from where the TeamViewer session is initiated. NX uses 800x600 resolutions with 8bit color dept.

Results

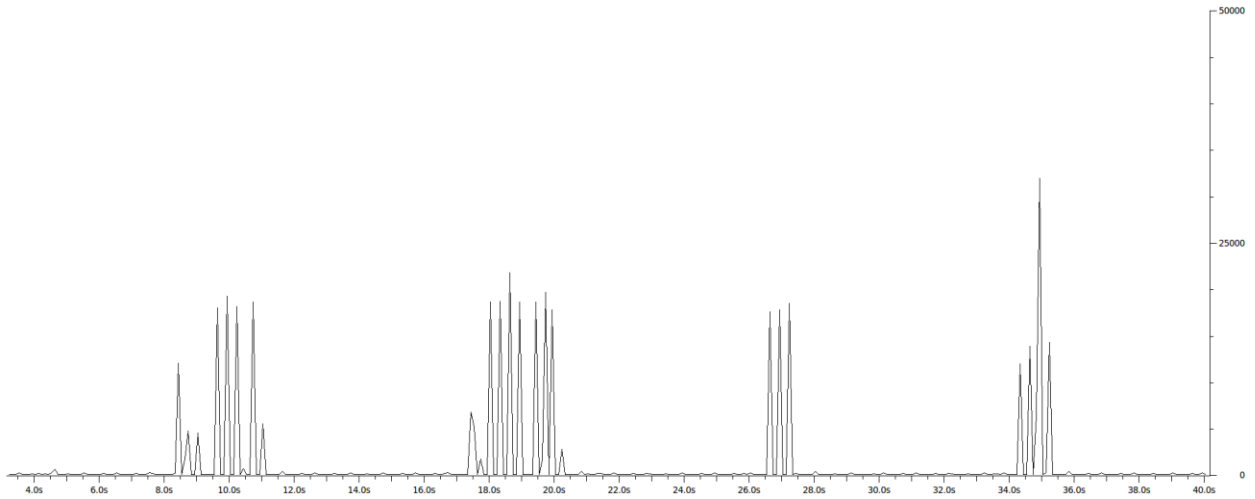


Figure 8: I/O Graph. Y=bytes.

No.	Data (bytes)	Time	Avg. mbps	Avg. Packets/s	Avg. Packets Size
1	102869 (100.5 KB)	2.6 sec	0.321	61.6	651.1 bytes
2	151749 (148.2 KB)	2.8 sec	0.439	77.1	712.4 bytes
3	53676 (52.4 KB)	0.6 sec	0.745	111.0	838.7 bytes
4	68999 (67.4 KB)	1.1 sec	0.504	105.6	105.6 bytes

8. Video / Frequent Screen Update

This section would deal with testing the handling of frequent and fast screen update by using video playback.

Notes:

- ✓ Difference between data transferred when playing back video clip through screen sharing solution and streaming with real-time solution such as MobiCents. Also to see the video quality and number of dropped frames.

Methodology

The method used here is by using a screen recorder record the video playback on the client pc. This way one step through the recorded session frame by frame using video editing software and in this case the Avidemux video editor is used. We can then and see how many frames are fully drawn as per second. The software used to record the screen here is the TipCam free screen recorder. It only supports recording at 9 frames per second without dropping to me frames itself. This would be sufficient for now as the video played back for testing is only 5 frames per second.

Wireshark is used to measure the bandwidth usage. The Wireshark capture was done on the client Nanoware PC this time unlike previous on the main PC.

Setting:

- Date : 18 November 2010
- Display Server : TightVNC Server on Windows 7, Nanoware PC
- Screen Resolution : 800x600
- Video size : ½ wide, 1/3 high
- Frame Rate : 1 frame/sec
- Client : XP Nanoware PC
- Network Latency : 60ms ± 5ms
- Notes:
This is more for determining bottom line. To see statistics for complete frames without some frames being dropped due to software not copying. How the screen sharing software deals with frame rates higher then it can paint would then be measured against these reading.

Results:

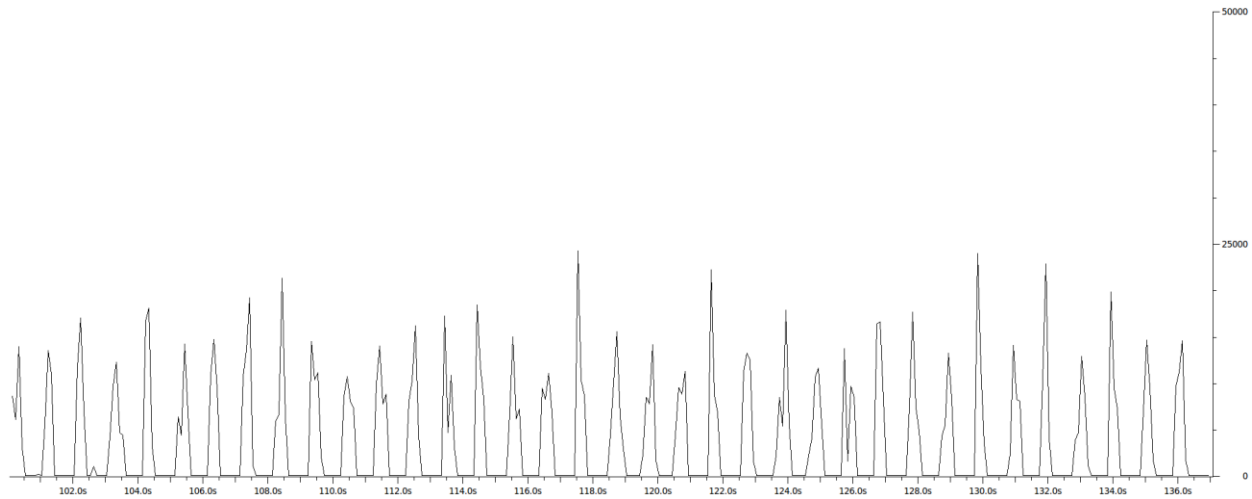


Figure 9: I/O Graph. Y=bytes

Network Traffic Summary:

- Time between first and last packet : 136.992 sec
- Bytes captured : 4504836 (4.296 MB)
- Average mbps : 0.263
- Average packets/sec : 55.95
- Average packet size : 587.71 bytes

Video Frame Summary:

- Total Time recorded : ?
- Total Complete Frames : ?
- Average Frame Rate : 1 frame/sec

Setting:

- Date : 18 November 2010
- Display Server : TightVNC Server on Windows 7, Nanoware PC
- Screen Resolution : 800x600
- Video size : ½ wide, 1/3 high
- Frame Rate : 5 frame/sec
- Client : XP Nanoware PC
- Network Latency : 60ms ± 5ms

- Notes:
???

Results:

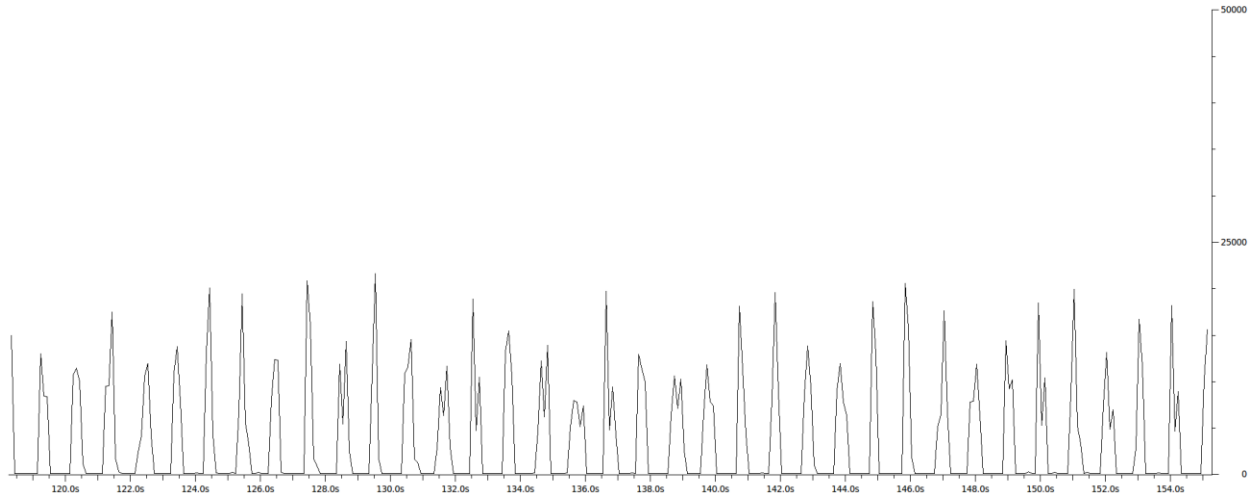


Figure 10: I/O Graph. Y=bytes

Network Traffic Summary:

- Time between first and last packet : 155.171 sec
- Bytes captured : 4852874 (4.628 MB)
- Average mbps : 0.250
- Average packets/sec : 52.774
- Average packet size : 592.609 bytes

Video Frame Summary:

- Total Time recorded : ?
- Total Complete Frames : ?
- Frame Rate : ?

18 November 2010

Setting:

- Date : 18 November 2010
- Display Server : Team Viewer on Windows 7, Nanoware PC
- Screen Resolution : 800x600, 32bpp
- Video size : ½ wide, 1/3 high
- Frame Rate : 1 frame/sec
- Client : XP Nanoware PC
- Network Latency : 60ms ± 5ms
- Transport Protocol : UDP
- Notes:
Quality is set to optimize for quality.

Results:

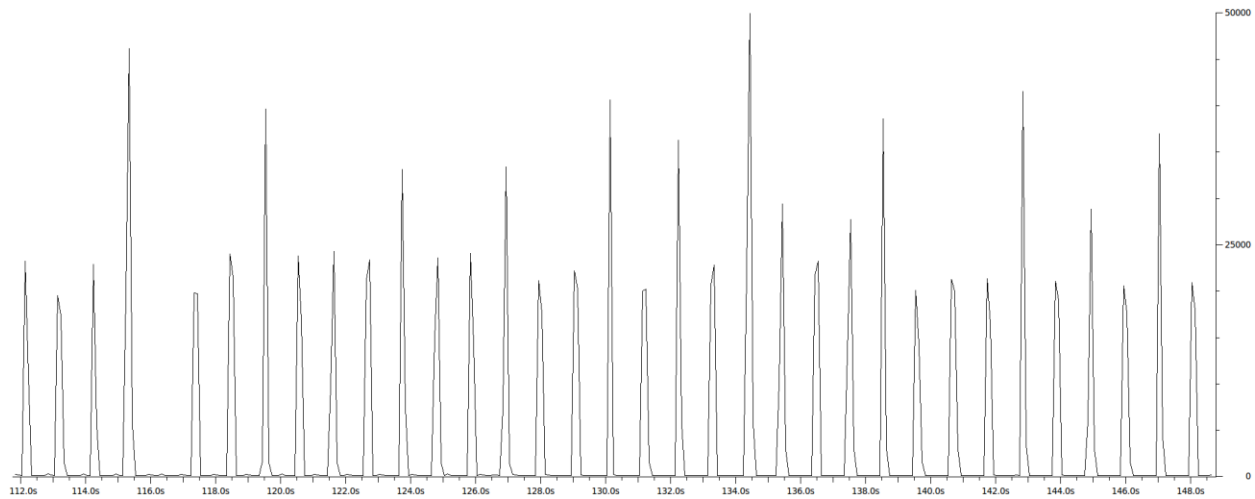


Figure 11: I/O Graph. Y=bytes

Network Traffic Summary:

- Time between first and last packet : 148.636 sec
- Bytes captured : 4822077 (4.599 MB)
- Average mbps : 0.260
- Average packets/sec : 47.922
- Average packet size : 676.973 bytes

Video Frame Summary:

- Frame Rate : 1 frame/sec

18 November 2010

Setting:

- Date : 18 November 2010
- Display Server : TeamViewer on Windows 7, Nanoware PC
- Screen Resolution : 800x600, 32bpp
- Video size : ½ wide, 1/3 high
- Frame Rate : 5 frame/sec
- Client : XP Nanoware PC
- Network Latency : 60ms ± 5ms
- Transport Protocol : UDP
- Notes:
The frames seem to be painted complete at once rather than painted line per line.

Results:

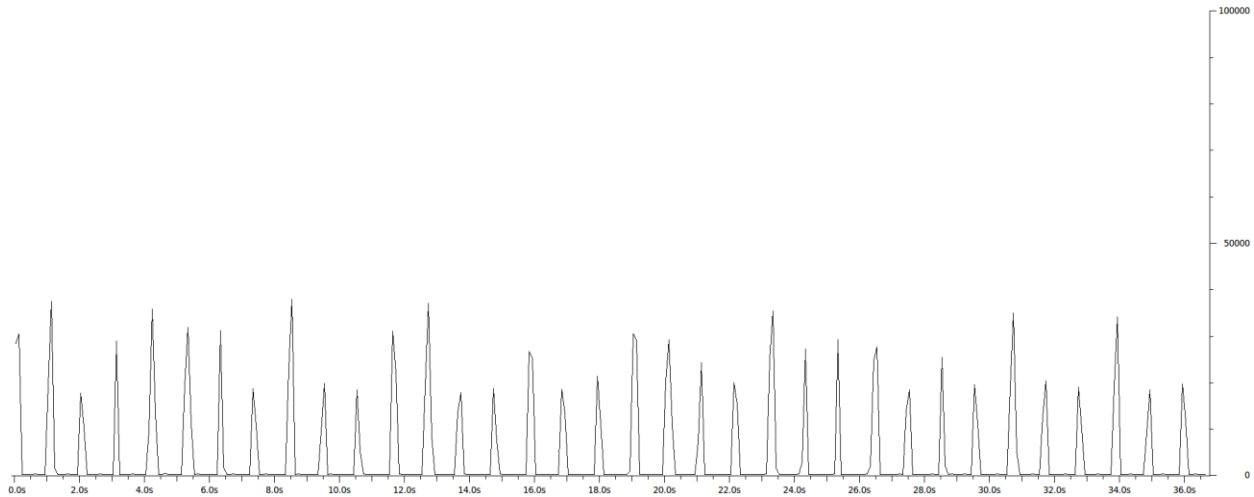


Figure 12: I/O Graph. Y=bytes

Network Traffic Summary:

- Time between first and last packet : 158.651 sec
- Bytes captured : 7143232 (6.812 MB)
- Average mbps : 0.360
- Average packets/sec : 158.651
- Average packet size : 740.845 bytes

Video Frame Summary:

- Total Time recorded : ?
- Total Complete Frames : ?
- Frame Rate : ?

Setting:

- Date : 18 November 2010
- Display Server : FreeNX on Windows 7, Nanoware PC
- Screen Resolution : 800x600
- Video size : ½ wide, 1/3 high
- Frame Rate : 1 frame/sec
- Client : XP Nanoware PC
- Network Latency : 60ms ± 5ms
- Transport Protocol : TCP

▪ Notes:

Could not do recording of desktop. The remote screen window stays still while in Wireshark the packet are still transmitted.

Results:

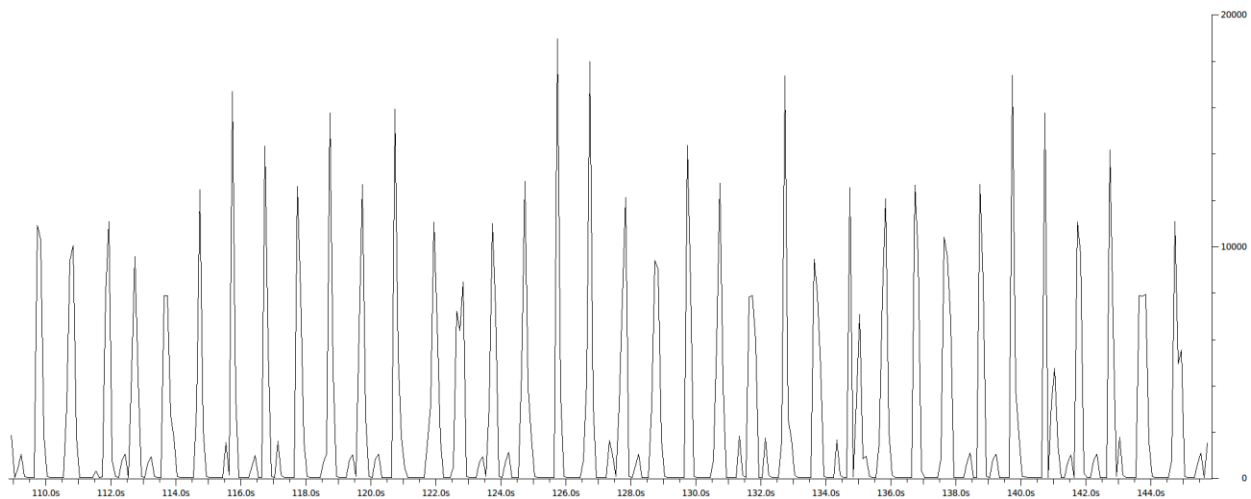


Figure 13: I/O Graph. Y=bytes

Network Traffic Summary:

- Time between first and last packet : 145.701 sec
- Bytes captured : 2653060 (2.530 MB)
- Average mbps : 0.146
- Average packets/sec : 33.754
- Average packet size : 539.249 bytes

18 November 2010

Video Frame Summary:

➤ Frame Rate : 1 frame/sec

Setting:

- Date : 18 November 2010
- Display Server : FreeNX on Windows 7, Nanoware PC
- Screen Resolution : 800x600
- Video size : ½ wide, 1/3 high
- Frame Rate : 5 frame/sec
- Client : XP Nanoware PC
- Network Latency : 60ms ± 5ms
- Transport Protocol : TCP

▪ Notes:

When viewing on screen, the frame rate seems to be 5 frame/sec at which video is played back.

Results:

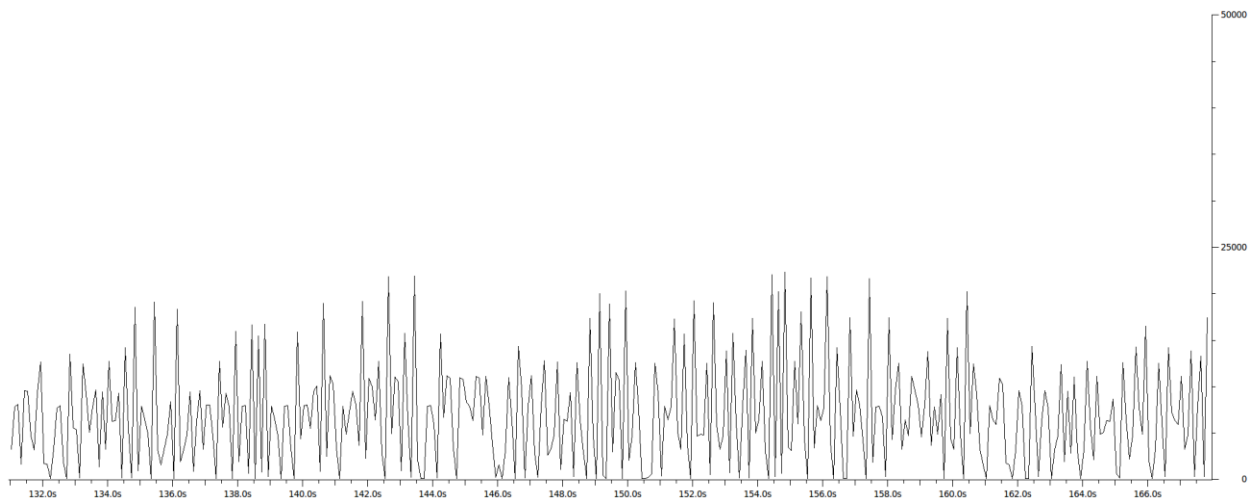


Figure 14: I/O Graph. Y=bytes

Network Traffic Summary:

- Time between first and last packet : 167.881 sec
 - Bytes captured : 7594652 (7.243 MB)
 - Average mbps : 0.362
 - Average packets/sec : 70.580
 - Average packet size : 640.953 bytes
-

Video Frame Summary:

- Frame Rate : 5 frame/sec (*observation on screen*)
-

9. User Use Cases

The following are testing scenarios of uses by users such as typing and using the mouse.

- Text
 - Type
 - Character repeat by holding key down
 - Arrow keys to move curser
 - Using shift key and CTRL key
- Mouse
 - Move cursor
 - Click and double click
 - Drag an object such as icon
 - Make are selection by drag