VNC in High-Latency Environments and Techniques for Improvement

Taurin Tan-atichat - ttanatic@cs.ucsd.edu
Joseph Pasquale - pasquale@cs.ucsd.edu

Dept. of Computer Science and Engineering
University of California, San Diego (UCSD)

IEEE GLOBECOM 2010
Miami, FL

December 7, 2010
Outline

- Introduction
- Goals and Non-Goals
- Design
- Experimental Results
- Conclusion
Introduction

- VNC (Virtual Network Computing) is a popular cross-platform thin-client system
- High-latency networks growing in popularity
- VNC client repeatedly requests framebuffer updates
- Performance suffers in high latency
  - Capped at 1/RTT FPS
Introduction - VNC System Overview
Goals and Non-Goals

- Goals of VNC-HL (High Latency)
  - Improve frame rate performance in high latency
  - Backwards compatible with VNC

- Non-Goals
  - Improve user input latency
Design

- Emulate a server-push system
  - Framebuffer update requests should arrive at the server quickly after anytime the framebuffer has changed
- Pre-request framebuffer updates
Design - VNC-HL System Overview

Framebuffer Update Request → T ms → Framebuffer Update Request → T ms → Framebuffer Update Request → ... → High-Latency Network

Client

Server
Design - VNC-HL Pre-Request Diagram

1. Start
2. Process Blocks
3. Set Timer for T ms
4. Send Framebuffer Update Request
5. Received Framebuffer Update Response
6. Timer Expires
Experimental Results

- Benchmarked video FPS (frames per second)
- Varied both latency and PRP (pre-request period)
Experimental Results

- Generally, a faster PRP (pre-request period) yields better FPS (frames per second)
- PRP of 50 ms yielded ~15 FPS in latencies of up to 500 ms
Experimental Results
Experimental Results

<table>
<thead>
<tr>
<th>Latency (ms)</th>
<th>0</th>
<th>50</th>
<th>500</th>
<th>1000</th>
</tr>
</thead>
<tbody>
<tr>
<td>VNC-HL 1 ms PRP</td>
<td>217.35 PRPS 21.0 FPS</td>
<td>212.74 PRPS 22.2 FPS</td>
<td>218.16 PRPS 17.9 FPS</td>
<td>197.86 PRPS 7.6 FPS</td>
</tr>
<tr>
<td>VNC-HL 50 ms PRP</td>
<td>4.91 PRPS 16.1 FPS</td>
<td>9.78 PRPS 14.8 FPS</td>
<td>7.44 PRPS 14.1 FPS</td>
<td>7.51 PRPS 9.6 FPS</td>
</tr>
<tr>
<td>VNC</td>
<td>- -</td>
<td>- -</td>
<td>- -</td>
<td>- -</td>
</tr>
<tr>
<td></td>
<td>10.8 FPS</td>
<td>3.0 FPS</td>
<td>0.4 FPS</td>
<td>0.2 FPS</td>
</tr>
</tbody>
</table>

- Anomaly to have a faster PRP perform much worse than a slower PRP
Experimental Results

- A fast PRP does not guarantee high FPS
- Under load, VNC server ignores requests if framebuffer has not changed since last request
- A PRP that is too fast can overload the server by filling up the TCP window
Experimental Results

When a PRP is very fast, it degrades FPS.

Over time, the flood of requests spread out (burstiness reduced) and FPS improves.
Experimental Results

- Best PRP in our system was 40-50 ms
- Equivalent to ~20-25 requests per second
- Highest performance our system could attain was < 25 FPS so there isn't any benefit to use a PRP faster than 40 ms
Conclusion

- VNC performs poorly in high latency
  - 3 FPS in 50 ms of latency
  - 0.4 FPS in 500 ms of latency
- VNC-HL performs much better
  - 14.8 FPS in 50 ms of latency with a 50 ms PRP
  - 14.1 FPS in 500 ms of latency with a 50 ms PRP
- Use a PRP as fast as the highest FPS the system supports but no faster

Note: FPS = frames per second  
     PRP = pre-request period
Q & A
Backup

- Generally, a faster PRP yields better FPS.
- PRP of 30 ms had a degraded performance in a latency of 1000 ms.