Improving Network Latency Effects in VNC

Cynthia Taylor, Joe Pasquale
University of California, San Diego
• Introduction
  • Thinner Clients
  • Desktop versus Thin Client
  • The Problem with Supporting Video
  • Server Push
  • Client Pull
  • Virtual Network Computing
  • Defining Performance
• Adding a Smart Proxy
• Experimental Design & Results
• Conclusion
Thinner Clients: Light Weight Devices
Desktop versus Thin Client
The Problem with Supporting Video

- Video is hard for Thin Client Systems
  - Frequent updates
  - Many pixel changes per update
  - All server generated
Server Push

- X-Windows is a server push system

Client-Pull

- VNC is a client-pull system.

Virtual Network Computing

- VNC is a widely-used thin client system.
- It is cross-platform and has several available open-source implementations.
- It was developed by Tristan Richardson at the Olivetti Research Lab.


How VNC Works

- It runs at the application layer and reads updates from the framebuffer.
Defining Performance

1. Client requests new update
   
   \[\text{client} \rightarrow \text{server}\]

2. Client waits
   
   \[\text{client} \rightarrow \text{server}\]

3. Server sends update
   
   \[\text{client} \leftarrow \text{server}\]

4. Client processes update
   
   \[\text{client} \leftarrow \text{server}\]
- Introduction
- Adding a Smart Proxy
  - VNC with High Network Latency
  - The Proxy and VNC
  - Pipelining Updates
  - Smart Proxy with High Network Latency
- Experimental Design & Results
- Conclusion
VNC with High Network Latency

- Client sends request - 200 ms
- Server sends update - 200 ms

Update Rate = 2.5 updates/second
The Proxy and VNC

- The Smart Proxy sends requests to the server at the rate the client is processing them, and quickly receives updates from the server.
- This lets the Smart Proxy adjust for latency between the client and server.
Pipelining Updates

- The proxy sends requests to the client at the rate the client is processing, without waiting for a request.
Smart Proxy with High Network Latency

- Client reads pipelined update from proxy - 75 ms

Update Rate = 13 updates/sec
• Introduction
• Adding a Smart Proxy
• Experimental Design & Results
• Conclusion
Experimental Design

- We use NetEm to add network delays to both client and server to simulate network latency
Results: Smart Proxy Outperforms Unmodified System
Conclusion

- We can improve VNC performance by having a Smart Proxy mediate the update rate over network latency.
- By using the Smart Proxy, we do not have to modify an existing code, avoiding issues of parallel code maintenance and source code availability.
Future Work

- Add different functionality to the Smart Proxy
  - Down sample or reduce dimensions of video
  - Add Machine Vision functionality such as face detection