Improving Thin Client Performance Using the Smart Proxy Architecture

Cynthia Taylor, Joe Pasquale
University of California, San Diego
● Introduction
  ● Sample Device
  ● Sample Application
● Definitions
● The Smart Proxy Architecture
● Improving VNC
● Results & Conclusion
Devices

Zypad Wearable Computer

128 MB Ram
GPS, Accelerometer

Video Glasses
Application

- Displaying contextual information about the user’s location

Virtual Worlds
Maps
Historic Information
• Introduction
• Definitions
  • What is Thin Client Computing?
  • Why Thin Clients?
  • Latency and Performance
• The Smart Proxy Architecture
• Improving VNC
• Results & Conclusion
What is Thin Client Computing?
What is Thin Client Computing?
What is Thin Client Computing?

User input → internet → Screen updates → Server
Why Thin Clients?
Lightweight Devices
Why Thin Clients? Intensive Applications

- Machine Learning/Vision
  - Object recognition
  - Speech recognition
- Graphics
  - Rendering
- Data Storage
  - Video
Why Thin Clients?
Security & Data Loss

- A lost laptop doesn’t mean lost data
- Helps companies stay compliant with privacy laws such as HIPAA
Latency and Performance
● Introduction
● Definitions
● The Smart Proxy Architecture
  ● Resource Assumptions
  ● The Smart Proxy Architecture
  ● Uses of the Smart Proxy
● Improving VNC
● Results & Conclusion
Resource Assumptions: Active Wireless Spaces
Uses of the Smart Proxy

- Buffering updates
- Compress or Decompress Updates
  - Scalable Video Coding
- Video Processing
- Encryption
Introduction
Definitions
The Smart Proxy Architecture
Improving VNC
  - What is VNC?
  - Defining Performance
  - The Proxy and VNC
  - Example
  - Implementation Details
Results & Conclusion
What Is VNC

- VNC is a widely-used thin client system with several available open-source implementations.
Defining Performance

1. Client requests new update
2. Client waits
3. Server sends update
4. Client processes update
The Proxy and VNC

- The Smart Proxy sends requests to the server at the rate the client is processing them, without waiting for an update from the server.
- This lets the Smart Proxy adjust for time delays between the client and server.
The Proxy and VNC

- The client sees the proxy as the server, and the server sees the proxy as the client.
- As long as the proxy sends and receives messages in the RFB protocol, the VNC client and server applications require no modifications.
Example

- Client sends request - 200 ms
- Server processes - 5 ms
- Server sends update - 200 ms
- Client processes - 5 ms

Total time = 410 ms
Example

- Proxy processes - 5 ms
- Proxy sends update to Client - 15 ms
- Client processes - 5 ms
- Client sends request - 15 ms

Total time = 40 ms
Example

The proxy sends requests to the server at the same rate the client is processing them, without waiting for a response from the server.
Implementation Details

If the proxy sends faster than the server can process updates, requests can accumulate at the server.

This causes more work for the server, resulting in a slower response.
Implementation Details

- Occasionally the server sends very large updates, which may be larger than the TCP window size.
- We are working at the application level, and the underlying protocols still require waiting for acknowledgements, which are affected by network delays
Results
Future Work

- Have the proxy automatically configure itself to send at the optimal rate.
- Is it possible to have the proxy perform as well as the client-server system with no delay?
Conclusion

- We can improve VNC performance by having a Smart Proxy mediate the update rate over network delays.
- Faster thin clients can help us integrate powerful computing into our mobile lives.