Improving Video Performance in VNC Under High Latency Conditions

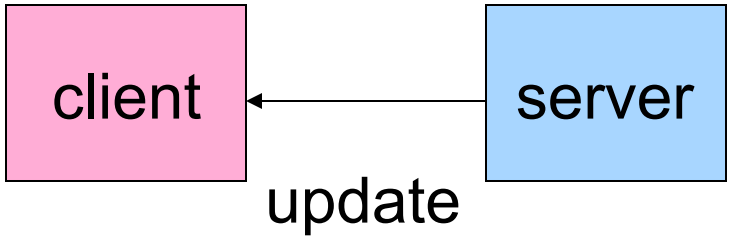
Cynthia Taylor, Joe Pasquale
University of California, San Diego

- Introduction
  - What Is Thin Client Computing?
  - Thinner Clients
  - Collaboration
  - Desktop versus Thin Client
  - The Problem with Supporting Video
  - Server Push
  - Client Pull
  - Virtual Network Computing
  - Defining Performance
- Adding a Message Accelerator
- Experimental Design & Results
- Conclusion

The Problem with Supporting Video

- Video is hard for Thin Client Systems
  - Frequent updates
  - Many pixel changes per update
  - All server generated
  - Becomes drastically worse over high latency

Server Push

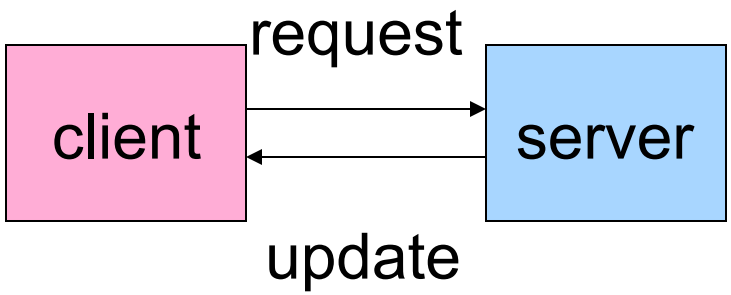


Server Push

- X-Windows is a server push system

Robert W. Scheifler and Jim Gettys. The x window system. ACM Trans. Graph., 5(2):79-109, 1986.

Client-Pull



Client Pull

- VNC is a client-pull system.

T. Richardson, Q. Stafford-Fraser, K.R. Wood, and A Hopper. Virtual network computing. Internet Computing, 2(1):33-38, 1998.

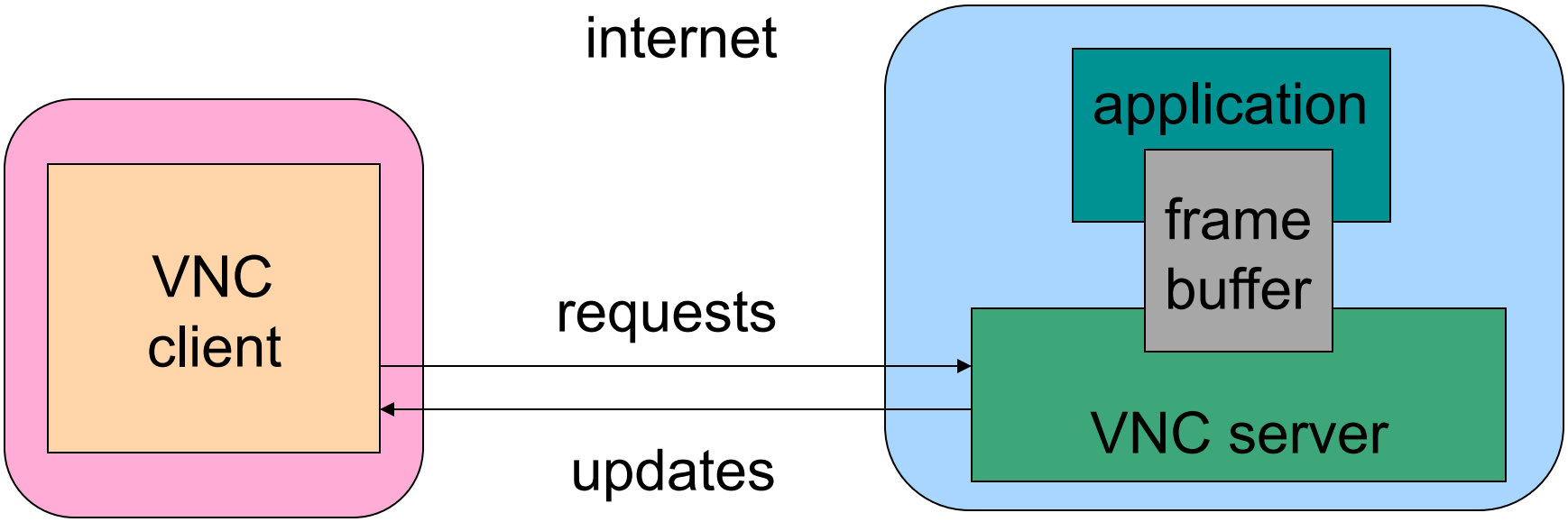
Virtual Network Computing

- 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.

T. Richardson, Q. Stafford-Fraser, K.R. Wood, and A Hopper. Virtual network computing. Internet Computing, 2(1):33-38, 1998.

Tristan Richardson. The RFB Protocol. Technical report, RealVNC Ltd, 2007.

How VNC Works



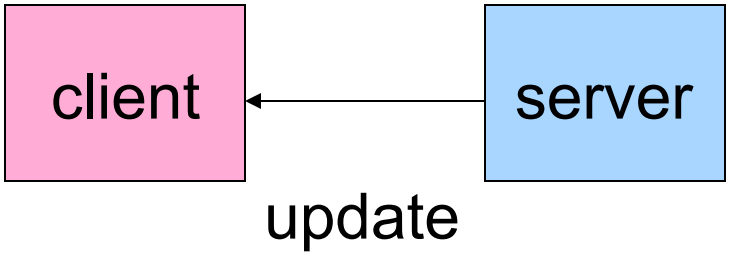
- It runs at the application layer and reads updates from the framebuffer.

Defining Performance

1. Client requests new update

2. Client waits

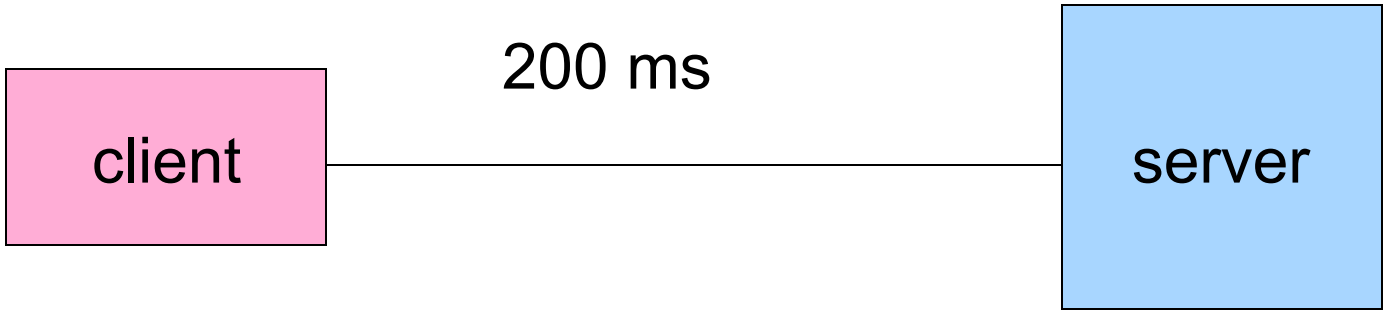
3. Server sends update



4. Client processes update

- Introduction
- Adding a Message Accelerator
  - VNC with High Network Latency
  - The Message Accelerator and VNC
  - Pipelining Updates
  - Message Accelerator with High Network Latency
- Experimental Design & Results
- Conclusion

VNC with High Network Latency



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

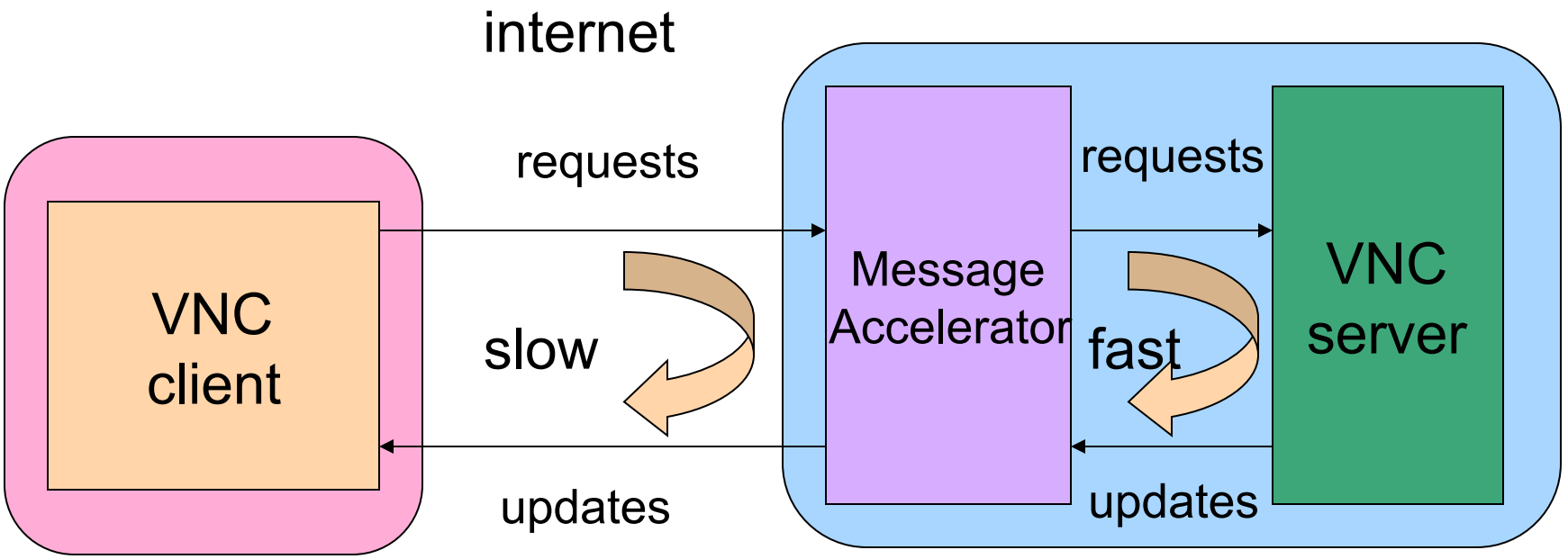
Update Rate = 2.5 updates/second

More Generally, Update Rate = 1/RTT

Two Approaches

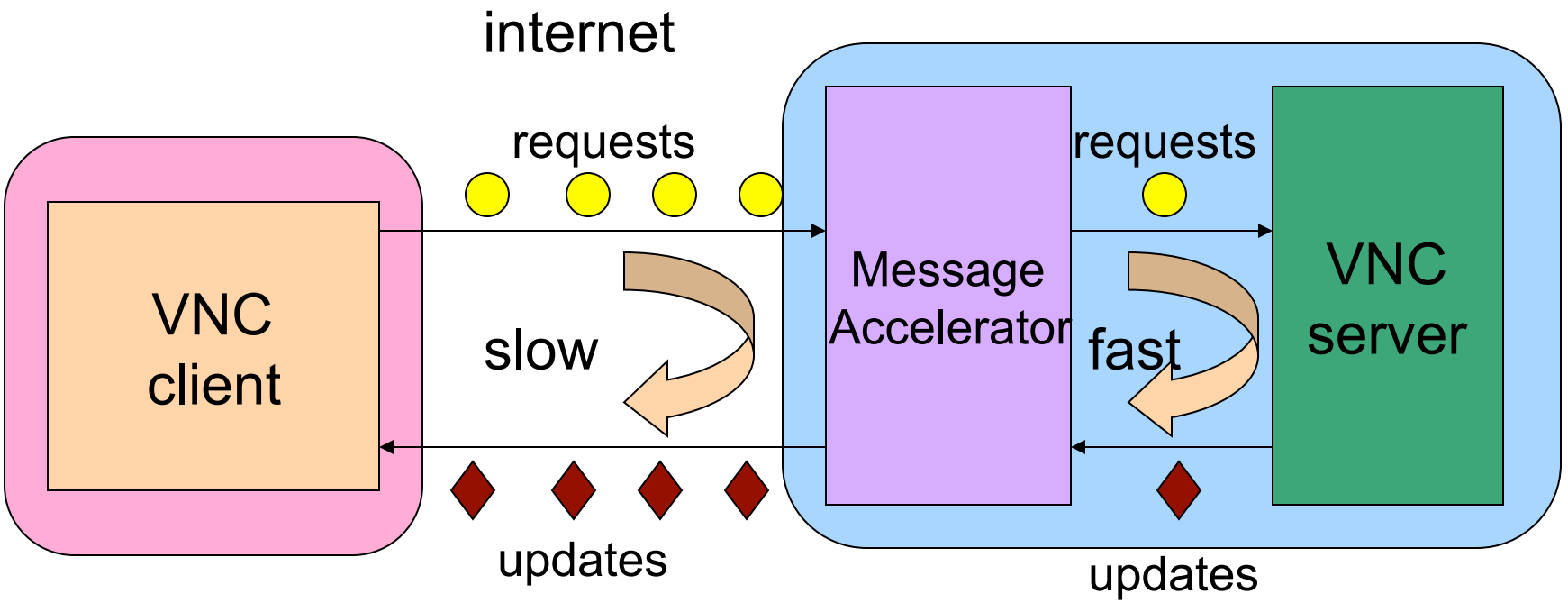
- Adding a proxy, unmodified client and server
- Modify the client

The Message Accelerator and VNC



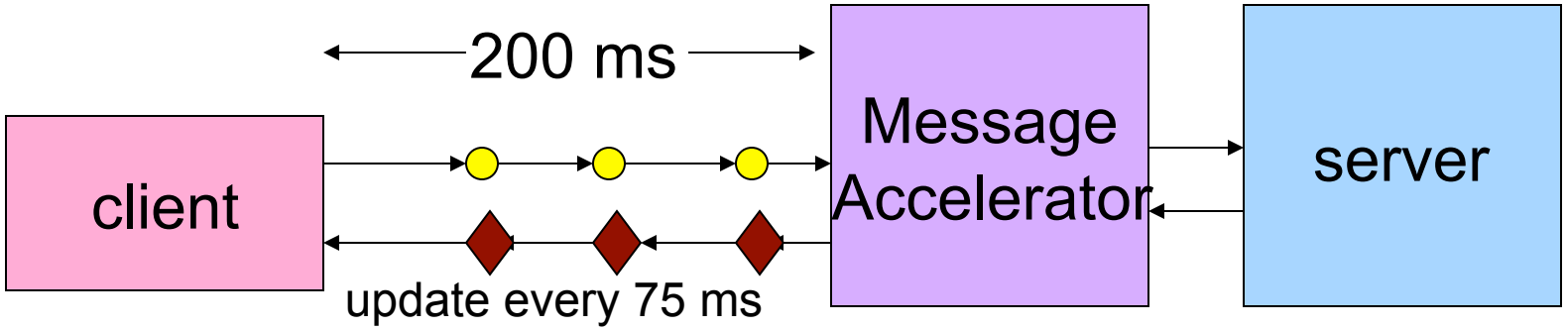
- The Message Accelerator sends requests to the server at the rate the client is processing them, and quickly receives updates from the server.
- This lets the Message Accelerator adjust for latency between the client and server

Pipelining Updates



- The proxy sends requests to the client at the rate the client is processing, without waiting for a request.

Message Accelerator - High Network Latency

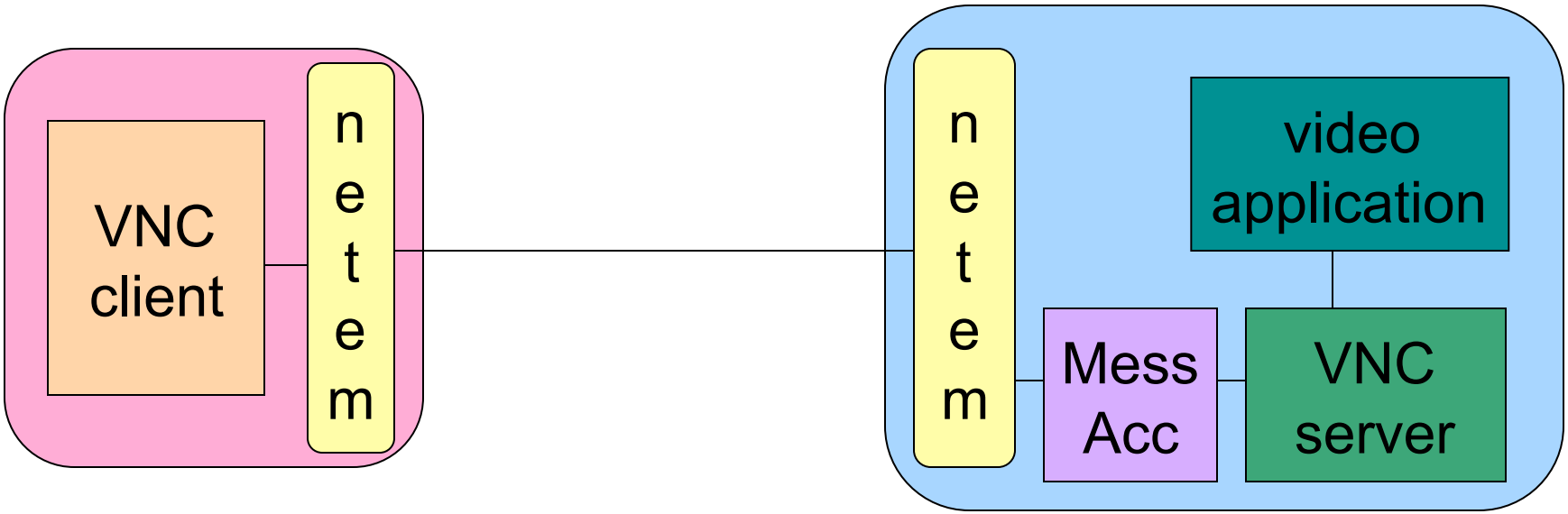


- Client reads pipelined update from proxy - 75 ms

Update Rate = 13 updates/sec

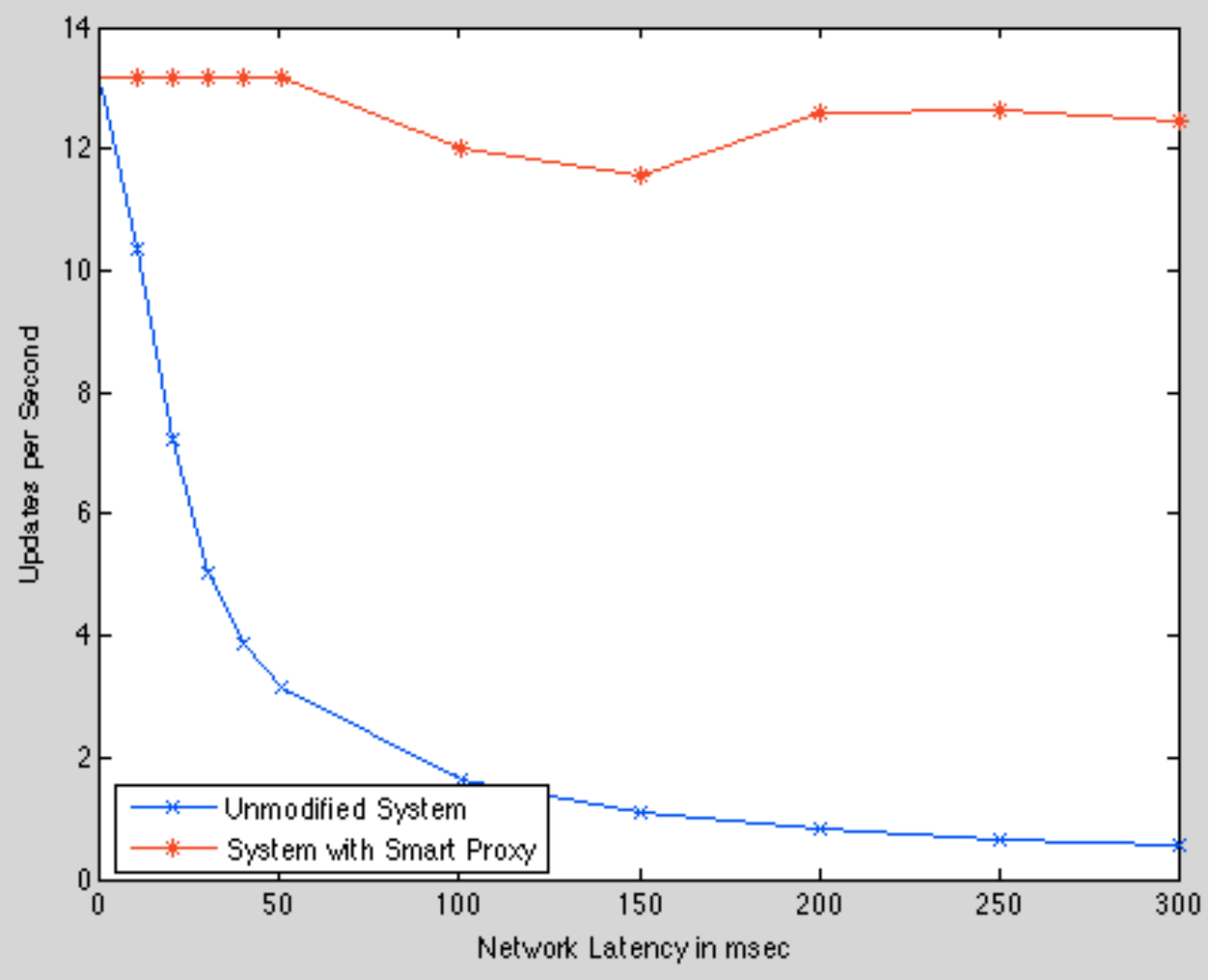
- Introduction
- Adding a Message Accelerator
- Experimental Design & Results
- Conclusion

Experimental Design



- We use NetEm to add network delays to both client and server to simulate network latency

Results: Message Accelerator Outperforms Unmodified System



Modify the Client

Conclusion

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