RTP Payload format for Application and Desktop Sharing

Omer Boyaci & Henning Schulzrinne
November 18, 2008
Application Sharing

- Sharing an application with multiple users
- There is only one copy of the application
- Participants do not need application itself
- Briefly, participants
  - receive screen updates
  - send keyboard and mouse events
Terminology

- Application Host (AH)
- Participant
- Remoting protocol
- Human Interface Protocol (HIP)
- Different modes of remote access
  - Remote Desktop Connection
  - (Full or partial)Screen Sharing
  - Application Sharing
Screenshot (2)
Architecture

Remoting protocol
Architecture

Participant → Application Host (AH) → Participant

Human Interface Protocol (HIP)
Architecture

Binary Floor Control Protocol (BFCP) manages the ownership of AH side human interface devices.
Multimedia Support (Movies)

- Composite image comparing JPEG and PNG: notice artifacts in JPEG versus solid PNG background.
Comparison of Sharing Systems in terms of web page visiting performance

![Bar chart showing comparison of VNC-ZRLE, BASS, and RDP systems in terms of total megabytes transferred. VNC-ZRLE has 25, BASS has 35, and RDP has 38.]

Boyaci & Schulzrinne, draft-boyaci-avt-app-sharing-00 10
Multimedia Support (Movies)
Comparison of Sharing Systems in terms of multimedia performance

Unlimited Bandwidth
Comparison of Sharing Systems in terms of multimedia performance

![Comparison Graph]

3Mb/s Bandwidth
Remoting Messages

• Application host (AH) to participants
  – WindowStateInfo
  – RegionUpdate
  – MoveRectangle
  – MousePointerInfo

• Participants to AH (NACK messages)
  – PLI (Picture Loss Indication)
  – NACK request
HIP Messages

• MousePressed
• MouseReleased
• MouseMoved
• MouseWheelMoved
• KeyPressed
• KeyReleased
• KeyTyped
VNC problems

- Client-pull based
- No multicast support
- Same encoding for all updates
- CPU usage increases for each new user
TeleTeachingTool (VNC based)

Figure 2: TeleTeachingTool environment overview

**Problems**
Modified VNC protocol and clients (TTT server is pushing updates)
No compression (only hextile encoding) because packets can get lost
Open Issues

• Transport Protocol
  – RTP
  – MSRP
  – Custom Made
Open Issues

• How to do retransmission requests for UDP clients

• **Current proposal**
  – NACK-based solution
  – NACK suppression to prevent floods
  – RTP-library level retransmissions
Current Internet Drafts

draft-boyaci-avt-app-sharing-00
draft-boyaci-avt-png-00
Backup Slides
Protocol Message Structures
Common remoting/hip Header
# A Window Record

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>WindowID</td>
<td>GroupID</td>
<td>Reserved</td>
</tr>
<tr>
<td>Left</td>
<td>Top</td>
<td>Width</td>
</tr>
<tr>
<td>Height</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
A Region Update Message

| Msg Type = 2 | PT | WindowID = 1 |
| Left |
| Top |

Payload
The MoveRectangle Message

Source Left
Source Top
Width
Height
Destination Left
Destination Top
The MousePressed Message

```plaintext
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-------------------------------------------------------------------------
|   Left
+-------------------------------------------------------------------------
|   Top
+-------------------------------------------------------------------------
```
The KeyPressed Message
The KeyTyped Message
Comparison of Sharing Systems
Screenshot (Overlapped Windows)
Screenshot (Multicast App. Tool)
Screenshot (Ultra VNC)
## Comparison of Sharing Systems

<table>
<thead>
<tr>
<th>System</th>
<th>Pull/Push</th>
<th>Technique</th>
<th>Multicast</th>
<th>App Sharing</th>
<th>Movies</th>
</tr>
</thead>
<tbody>
<tr>
<td>VNC</td>
<td>Pull</td>
<td>Mirror Driver</td>
<td></td>
<td>Region</td>
<td></td>
</tr>
<tr>
<td>TTT: TeleTeachingTool(VNC)</td>
<td>Pull+Push</td>
<td>Mirror Driver</td>
<td></td>
<td>Region</td>
<td></td>
</tr>
<tr>
<td>SharedAppVNC</td>
<td>Pull</td>
<td>Polling</td>
<td></td>
<td>Region</td>
<td></td>
</tr>
<tr>
<td>MAST</td>
<td>Push</td>
<td>Polling</td>
<td>Region</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RDP</td>
<td>Push</td>
<td>Mirror Driver</td>
<td></td>
<td></td>
<td>High B/W</td>
</tr>
<tr>
<td>THINC</td>
<td>Push</td>
<td>Mirror Driver</td>
<td></td>
<td></td>
<td>High B/W</td>
</tr>
<tr>
<td>Distributed Workspace</td>
<td>Push</td>
<td>Mirror Driver</td>
<td></td>
<td></td>
<td>High B/W</td>
</tr>
</tbody>
</table>
VNC problems

- Client-pull based
- No multicast support
- Same encoding for all updates
- CPU usage increases for each new user
TeleTeachingTool (VNC based)

Figure 2: TeleTeachingTool environment overview
Information about BASS
## Supported Platforms/OS

<table>
<thead>
<tr>
<th></th>
<th>Server</th>
<th>Client*</th>
</tr>
</thead>
<tbody>
<tr>
<td>Windows</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>*nix</td>
<td>-+</td>
<td>+</td>
</tr>
<tr>
<td>Mac OS X</td>
<td>-+</td>
<td>+</td>
</tr>
</tbody>
</table>

*Client is Java based.*
Client (Viewer) Architecture

• Client can
  – Connect to server
  – Wait for incoming connections

• Client supports
  – TCP
  – UDP (+Multicast)
BASS Windows Server Architecture

User space

Kernel space

Server Main Thread

Mirror Driver

Window coordinates
BASS Windows Server Architecture

Server Main Thread

User space

Kernel space

Mirror Driver

Window coordinates
BASS Windows Server Architecture

Kernel space

User space

Server Main Thread

Mirror Driver

Window coordinates
BASS Windows Server Architecture

Kernel space

User space

Server Main Thread

Client Thread1

Client Thread2

Client Thread3

Window coordinates

X: 320, Y: 320
X: 340, Y: 450
X: 420, Y: 200
X: 430, Y: 400
X: 330, Y: 300
X: 340, Y: 550
X: 300, Y: 500
X: 340, Y: 500
Challenges

• Different client bandwidths/speeds
• Late Joiner
• The effects of packet loss
• Reliable multicast
Multimedia Support (Movies)
Multimedia Support (Movies)

- Our system uses PNG to compress and transmit the region updates.
- PNG is lossless and effective for computer-generated images but ineffective for real-world captures like pictures or movies.
- JPG is more suitable for photographic images.
- However, JPG is lossy and not effective for computer-generated images (text, line, shapes, ...).
- Our system should use both.
Multimedia Support (Movies)

- Composite image comparing JPEG and PNG: notice artifacts in JPEG versus solid PNG background.
Multimedia Support (PNG vs JPG)

Image Size

Size x
360x150
162K

Size 4x
720x300
648K
Multimedia Support (PNG vs JPG)

Ethernet (60Mb/s)

Boyaci & Schulzrinne, draft-boyaci-avt-app-sharing-00
Multimedia Support (PNG vs JPG)

**FPS**

<table>
<thead>
<tr>
<th>Size x</th>
<th>FPS PNG</th>
<th>FPS JPG</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>22.5</td>
<td>7.5</td>
</tr>
<tr>
<td>4</td>
<td>20</td>
<td>10</td>
</tr>
</tbody>
</table>

**KB/s**

<table>
<thead>
<tr>
<th>Size x</th>
<th>KB/s PNG</th>
<th>KB/s JPG</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1100</td>
<td>1000</td>
</tr>
<tr>
<td>4</td>
<td>900</td>
<td>800</td>
</tr>
</tbody>
</table>

**Wireless (4Mb/s)**
PNG/JPG Detection Algorithm

Region > 40,000px?
- YES
- NO

New Region?
- YES
- NO

Continue Checking

Detected?
- NO
- YES

Use Detected Format

Create a record & Start Checking

Region record
- coordinates
- PNG Size
- counter
- Time Stamp
Sharing a Movie (Media Player)
Sharing a Movie File

• Capturing from the Frame Buffer is expensive.
• Instead
  – Transcode the movie to Theora beforehand
• Then stream the theora directly to participants
  – Java client supports theora playback
• Negligible CPU usage during playback on the server
  –
Sharing a Movie (Our Method)
Sharing a Movie File

Movie File (wmv, mpg, divx)

FFMpeg2Theora

Theora Movie

Java Streaming Server

UDP/Multicast

Java Client

Java Client
Challenges

- Different client bandwidths/speeds
- Late Joiner
- The effects of packet loss
- Reliable multicast
Different Client Bandwidths/Speeds
Different Client Bandwidths/Speeds

- Possible Solutions
  - Slowest one
  - Average speed
  - Fastest one
Different Client Bandwidths/Speeds

• Possible Solutions
  – Slowest one
    • Problem: Penalize everybody except the slowest
  – Average speed
  – Fastest one
Different Client Bandwidths/Speeds

• Possible Solutions
  – Slowest one
    • Problem: Penalize everybody except the slowest
  – Average speed
    • Possible solution
  – Fastest one
Different Client Bandwidths/Speeds

• Possible Solutions
  – Slowest one
    • Problem: Penalize everybody except the slowest
  – Average speed
    • Possible solution (Can we do better?)
  – Fastest one
Different Client Bandwidths/Speeds

- Possible Solutions
  - Slowest one
    - Problem: Penalize everybody except the slowest
  - Average speed
    - Possible solution (Can we do better?)
  - Fastest one
    - The best solution
    - Client bandwidths are fully utilized
Different Client Bandwidths/Speeds
Different Client Bandwidths/Speeds

1 Mbps

2 Mbps

3 Mbps

Server Main Thread
Challenges

- Different client bandwidths/speeds
- Late Joiner
- The effects of packet loss
- Reliable multicast
Late Joiner

• Force server to generate full screen update
Late Joiner

• Force server to generate full screen update
  – Problems
    • Misbehaving clients can degrade performance
    • If Join/Leave rate is high, too much burden on server
  – Solution
    • Generate full screen updates if really necessary
    • Otherwise start the new client from last full screen update
Different Client Bandwidths/Speeds

Server Main Thread

1Mbps
Client Thread -1

2Mbps
Client Thread -2

3Mbps
Client Thread -3

8Mbps
Client Thread -4
Different Client Bandwidths/Speeds

- 1 Mbps
- 2 Mbps
- 3 Mbps
- 8 Mbps
Different Client Bandwidths/Speeds

1M bps
Client Thread -1

2M bps
Client Thread -2

3M bps
Client Thread -3

8M bps
Client Thread -4

Server Main Thread
Different Client Bandwidths/Speeds

1 Mbps
1

2 Mbps
2

3 Mbps
3

8 Mbps
4

Server Main Thread

Client Thread -1

Client Thread -2

Client Thread -3

Client Thread -4
Different Client Bandwidths/Speeds

1 Mbps: Client Thread -1
2 Mbps: Client Thread -2
3 Mbps: Client Thread -3
8 Mbps: Client Thread -4

Server Main Thread
Different Client Bandwidths/Speeds

1 Mbps

2 Mbps

3 Mbps

8 Mbps

Server Main Thread

Client Thread -1

Client Thread -2

Client Thread -3

Client Thread -4
Different Client Bandwidths/Speeds

1 Mbps
Client Thread -1

2 Mbps
Client Thread -2

3 Mbps
Client Thread -3

8 Mbps
Client Thread -4

2 Mbps
Client Thread -5

Server Main Thread
Different Client Bandwidths/Speeds

Server Main Thread

1 Mbps
- Client Thread -1

2 Mbps
- Client Thread -2

3 Mbps
- Client Thread -3

8 Mbps
- Client Thread -4

2 Mbps
- Client Thread -5
Challenges

- Different client bandwidths/speeds
- Late Joiner
- **The effects of packet loss**
- Reliable multicast
The effects of Packet Loss

• This problem applies to
  – Multicast
  – UDP

• The PNG images can be large
  – Regular desktop can be ~900KB
  – ~600 Ethernet packets
  – One packet loss wastes all PNG image
The effects of Packet Loss

• Solution
  – Small PNG images
    • Around ~1500 bytes
    • Consist of a few scanlines
  – Disadvantages
    • Increased CPU usage (client&server)
    • Lower compression ratio (%20 lower)
  – Advantages
    • One packet loss = no update for a few scanlines
The effects of Packet Loss
Challenges

- Different client bandwidths/speeds
- Late Joiner
- The effects of packet loss
- Reliable multicast
Reliable Multicast

- RTP Library stores last N rtp packets
- Clients send NACK for lost packets
- RTP Library resend the requested packets
The effects of Packet Loss
Overview

• Introduction
• Demo
• Architecture
• Challenges
• Features
• Conclusion
Recording

- Clients can record the whole/part session
- Anybody can play these files locally
- These files can be streamed to receivers via streaming server
- Streaming server supports multiple receivers
  - Also late joiners
Listening Client

- Client waits for incoming connections
- It can display windows from multiple users
- Can be used for RGB cable replacement
Conclusion

• Application sharing allows users to share a single application with multiple participants.

• Participants don't need the application.

• It is not specific to a single application.

• Extra features like recording is added.