Multicast Video-Streaming on Embedded Linux Environment

Daichi Fukui, Toshiba Corporation
Japan Technical Jamboree 63
Dec 1st, 2017
Background

- **Multicast networking**
  - Use single IP address to send packets to multiple clients
  - 224.0.0.0 to 239.255.255.255 for IPv4

- **Video streaming**
  - Play video via networking
  - No need to save a video on disks

- **Multicast streaming on Linux aimed for IoT**
  - Multicast networking + Video streaming
  - Useful for playing a video simultaneously on multiple clients
  - Demand for multicast-streaming videos in a closed network
Purpose

• Research how to multicast-stream video on Linux
  – Find software for multicast-streaming in an embedded board
  – Give appropriate parameters to streaming software

• Build video-streaming system using Raspberry pi
  – Multicast streaming of videos using Raspberry pi
  – Low-cost embedded board to stream/play videos
  – Reduced hardware development cost

• Evaluate capability to stream/play video
  – Video-streaming load on a server according to # of clients
  – Video-playing load on clients
    • Comparison of between PC/embedded board
# Video streaming applications

<table>
<thead>
<tr>
<th>S/W name</th>
<th>License</th>
<th>Architecture</th>
<th>Available on boards?</th>
<th>debian package</th>
<th>Language</th>
<th>Supported media format</th>
<th>User Interface (server)</th>
<th>User Interface (client)</th>
</tr>
</thead>
<tbody>
<tr>
<td>icecast</td>
<td>GPLv2</td>
<td>x86, ARM</td>
<td>○</td>
<td>○</td>
<td>C, JS</td>
<td>(audio only)</td>
<td>Command line</td>
<td>Web browser</td>
</tr>
<tr>
<td>gstreamer</td>
<td>LGPL</td>
<td>x86, ARM</td>
<td>○</td>
<td>○</td>
<td>C</td>
<td>FLV, MP4, ...</td>
<td>Command line</td>
<td>X server, Command line</td>
</tr>
<tr>
<td>VLC</td>
<td>LGPLv2.1</td>
<td>x86, ARM</td>
<td>○</td>
<td>○</td>
<td>C/C++</td>
<td>FLV, MP4, ...</td>
<td>Command line, Qt</td>
<td>Qt</td>
</tr>
<tr>
<td>FFmpeg</td>
<td>LGPL2.1 + GPLv2+</td>
<td>x86, ARM</td>
<td>○</td>
<td>△</td>
<td>C</td>
<td>MP4, ...</td>
<td>Command line</td>
<td>X server, command line</td>
</tr>
<tr>
<td>Mist Server</td>
<td>aGPLv3</td>
<td>x86, ARM, MIPS</td>
<td>○</td>
<td>×</td>
<td>C, JS</td>
<td>FLV, MP3, OGG</td>
<td>Web browser</td>
<td>Web browser</td>
</tr>
<tr>
<td>red5</td>
<td>Apache</td>
<td>x86, ARM</td>
<td>×</td>
<td>×</td>
<td>Java</td>
<td>FLV, MP4, ...</td>
<td>Web browser, Flash</td>
<td>Web browser, Flash</td>
</tr>
</tbody>
</table>
# Video streaming applications

<table>
<thead>
<tr>
<th>S/W name</th>
<th>License</th>
<th>Architectures</th>
<th>Available on boards?</th>
<th>debian package</th>
<th>Language</th>
<th>Supported media format</th>
<th>User Interface (server)</th>
<th>User Interface (client)</th>
</tr>
</thead>
<tbody>
<tr>
<td>icecast</td>
<td>GPLv2</td>
<td>x86, ARM</td>
<td>○</td>
<td>○</td>
<td>C, JS</td>
<td>(audio only)</td>
<td>Command line</td>
<td>Web browser</td>
</tr>
<tr>
<td>gstreamer</td>
<td>LGPL</td>
<td>x86, ARM</td>
<td>○</td>
<td>○</td>
<td>C</td>
<td>FLV, MP4, ...</td>
<td>Command line</td>
<td>X server, Command line</td>
</tr>
<tr>
<td>VLC</td>
<td>LGPLv2.1</td>
<td>x86, ARM</td>
<td>○</td>
<td>○</td>
<td>C/C++</td>
<td>FLV, MP4, ...</td>
<td>Command line, Qt</td>
<td>Qt</td>
</tr>
<tr>
<td>FFmpeg</td>
<td>LGPL2.1+</td>
<td>x86, ARM</td>
<td>○</td>
<td>△</td>
<td>C</td>
<td>MP4, ...</td>
<td>Command line</td>
<td>X server, command line</td>
</tr>
<tr>
<td>Mist Server</td>
<td>aGPLv3</td>
<td>x86, ARM, MIPS</td>
<td>×</td>
<td>×</td>
<td>C, JS</td>
<td>FLV, MP3, OGG</td>
<td>Web browser</td>
<td>Web browser</td>
</tr>
<tr>
<td>red5</td>
<td>Apache</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>FLV, MP4, ...</td>
<td>Web browser, Flash</td>
<td>Web browser, Flash</td>
</tr>
</tbody>
</table>

**FFmpeg/VLC are suitable because of …**
- many projects using FFmpeg/VLC,
- adequate documentation,
- low-memory requirement,
- easier parameter setting
## Evaluation environment

<table>
<thead>
<tr>
<th>Purpose</th>
<th>Name</th>
<th>CPU</th>
<th>NIC</th>
<th>Mem.</th>
<th>Monitor</th>
<th>OS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Client</td>
<td>PC1</td>
<td>1.4GHz x2</td>
<td>1Gbps</td>
<td>4GB</td>
<td>1280x800</td>
<td>Ubuntu 16.04</td>
</tr>
<tr>
<td></td>
<td>PC2</td>
<td>2.7GHz x2(4)</td>
<td>1Gbps</td>
<td>8GB</td>
<td>1366x768</td>
<td>Ubuntu 16.04</td>
</tr>
<tr>
<td></td>
<td>PC3</td>
<td>1.2GHz x2</td>
<td>1Gbps</td>
<td>3GB</td>
<td>1280x800</td>
<td>Ubuntu 16.04</td>
</tr>
<tr>
<td></td>
<td>PC4</td>
<td>2.7GHz x2(4)</td>
<td>1Gbps</td>
<td>4GB</td>
<td>1600x900</td>
<td>Ubuntu 14.04</td>
</tr>
<tr>
<td></td>
<td>RaspberryPi V1 Type B+ (*)</td>
<td>0.7GHz x1</td>
<td>100Mbps</td>
<td>400MB</td>
<td>1280x800</td>
<td>Raspbian 8.0</td>
</tr>
<tr>
<td>Server</td>
<td>PC5</td>
<td>1.7GHz x2(4)</td>
<td>1Gbps</td>
<td>4GB</td>
<td>1366x768</td>
<td>Ubuntu 16.04</td>
</tr>
</tbody>
</table>

* Low-price and small single board computer produced by Raspberry Pi Foundation
  Processor: ARMv6 (ARM1176JZF-S)
  Price: 40USD
Multicast networking

- 224.0.0.0/4 as IPv4 multicast address
- Streaming software
  - Server (send streaming): VLC, FFmpeg
  - Clients (receive streaming): VLC, omxplayer (RPi only)
- Network

# of clients varies from 1 to 5
omxplayer

- Hardware-accelerated video/audio player
- Other players are not HW-accelerated by default
  - So it is hard to play videos on RPi without omxplayer
  - Need to rebuild the players to exploit GPU power on RPi
- Available as raspbian package
Evaluation of network bandwidth

- **Details of evaluation environment**
  - **S/W:** iperf
  - **Sender:** send UDP packets to receiver
    - Bitrate increased gradually starting from 10Mbps
    - 10, 20, 30, ..., 100, 200, 400, 800, 1000 Mbps
  - **Receiver:** receive UDP packets sent from sender
  - **Value:** Network bandwidth of PC/RPi

- **Commands to obtain data**
  - **sender**
    ```
    $ iperf -c ${ipaddr} -y C -u -f m -b ${size}M -i 1 -t 11
    ```
  - **receiver**
    ```
    $ iperf -s -y C -u -f m -i 1 > logfile
    ```
Network bandwidth

- **PC (receiver)** – Packet loss rate is almost zero
  - Max bitrate: 766Mbps

- **RPi (receiver)**
  - Max bitrate: 57Mbps
Evaluation of video multicast-streaming

• Details of evaluation environment
  – Measuring duration: 120 seconds
  – Server: VLC/FFmpeg used to multicast-stream video
  – Client: VLC, omxplayer (RPI only) to play video
  – Value: CPU usage, network load
    • Calculate median for each data

• Commands to obtain data
  – CPU usage
    $ top -d1 -n120 -b > logfile
  – Network load
    $ timeout 120 dstat --noheader --net-packet --output logfile
Details of sample video

• Underwater video
  – Duration: 30 seconds
  – File size: 57.6 MB
  – Video bitrate: 15377 kbps
  – Resolution: 1920x1080 (played as full screen)
  – Frame rate: 30fps
  – Video codec: H264
  – Audio: None
  – Media format: MPEG4
  – Source: orangeHD.com
Parameter setting for streaming

• VLC
  – Bitrate limited to 2M bps due to low VLC capability of streaming

  vlc -v underwater-video.mp4
  --sout '#transcode{vcodec=h264,venc=x264}
  '{scenecut=20,bframes=0},vb=2048,scale=1.0}'
  ':rtp{access=udp,mux=ts,dst=224.0.0.1,port=1234,'
  'sap,name="water"}'} ¥
  :sout-all

• FFmpeg
  – Bitrate limited to 2M/8M bps due to low RPi capability of playing

  ffmpeg -i "underwater-video.mp4" ¥
  -preset ultrafast ¥
  -vcodec libx264 ¥
  -tune zerolatency ¥
  -b 2048k (or 8192k)¥
  -f mpegts udp://224.0.0.1:1234
Load on server/VLC (bitrate=2048k)

- Constant load regardless of # of clients

CPU usage on server PC5

<table>
<thead>
<tr>
<th># of clients</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU usage (%)</td>
<td>29.5</td>
<td>28.8</td>
<td>31.0</td>
<td>29.0</td>
<td>29.7</td>
</tr>
</tbody>
</table>

Packet sending load on server PC5

<table>
<thead>
<tr>
<th># of clients</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet sending load (Mbps)</td>
<td>2.12</td>
<td>2.13</td>
<td>2.20</td>
<td>2.09</td>
<td>2.01</td>
</tr>
</tbody>
</table>
Load on clients (bitrate=2048k)

**PC1 (VLC to play streaming)**

CPU usage on client PC1

<table>
<thead>
<tr>
<th># of clients</th>
<th>CPU usage (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>24.5</td>
</tr>
<tr>
<td>2</td>
<td>25.4</td>
</tr>
<tr>
<td>3</td>
<td>24.2</td>
</tr>
<tr>
<td>4</td>
<td>27.9</td>
</tr>
<tr>
<td>5</td>
<td>25.5</td>
</tr>
</tbody>
</table>

Packet receiving load on client PC1

<table>
<thead>
<tr>
<th># of clients</th>
<th>Packet receiving load (Mbps)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2.07</td>
</tr>
<tr>
<td>2</td>
<td>2.21</td>
</tr>
<tr>
<td>3</td>
<td>2.09</td>
</tr>
<tr>
<td>4</td>
<td>2.17</td>
</tr>
<tr>
<td>5</td>
<td>2.05</td>
</tr>
</tbody>
</table>

**Rasp.Pi (omxplayer to play streaming)**

CPU usage on client RPi

<table>
<thead>
<tr>
<th># of clients</th>
<th>CPU usage (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>8.4</td>
</tr>
<tr>
<td>2</td>
<td>6.3</td>
</tr>
<tr>
<td>3</td>
<td>8.4</td>
</tr>
<tr>
<td>4</td>
<td>7.8</td>
</tr>
<tr>
<td>5</td>
<td>4.4</td>
</tr>
</tbody>
</table>

Packet receiving load on client RPi

<table>
<thead>
<tr>
<th># of clients</th>
<th>Packet receiving load (Mbps)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2.21</td>
</tr>
<tr>
<td>2</td>
<td>2.21</td>
</tr>
<tr>
<td>3</td>
<td>2.00</td>
</tr>
<tr>
<td>4</td>
<td>2.11</td>
</tr>
<tr>
<td>5</td>
<td>2.00</td>
</tr>
</tbody>
</table>
Load on clients (bitrate=2048k)

- **PC1 (VLC to play streaming)**

  CPU usage on client PC1

  - # of clients: 1, 2, 3, 4, 5
  - CPU usage (%): 24.5, 25.4, 24.2, 27.9, 25.5

  Packet receiving load on client PC1

  - # of clients: 1, 2, 3, 4, 5
  - Packet receiving load (Mbps): 2.07, 2.21, 2.09, 2.17, 2.05

- **Rasp.Pi (omxplayer to play streaming)**

  CPU usage on client RPi

  - # of clients: 1, 2, 3, 4, 5
  - CPU usage (%): 2.07, 2.21, 2.00, 2.11, 2.00

  Packet receiving load on client RPi

  - # of clients: 1, 2, 3, 4, 5
  - Packet receiving load (Mbps): 2.21, 2.21, 2.00, 2.11, 2.00

  **Constant load regardless of # of clients**
Load on server/FFmpeg (bitrate=2048k)

Constant load regardless of # of clients

CPU usage on server

<table>
<thead>
<tr>
<th># of clients</th>
<th>CPU usage (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>88.5</td>
</tr>
<tr>
<td>2</td>
<td>87</td>
</tr>
<tr>
<td>3</td>
<td>87.3</td>
</tr>
<tr>
<td>4</td>
<td>88.5</td>
</tr>
<tr>
<td>5</td>
<td>84.2</td>
</tr>
</tbody>
</table>

Packet sending load on server

<table>
<thead>
<tr>
<th># of clients</th>
<th>Packet sending load (Mbps)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3.5</td>
</tr>
<tr>
<td>2</td>
<td>3.6</td>
</tr>
<tr>
<td>3</td>
<td>3.6</td>
</tr>
<tr>
<td>4</td>
<td>3.6</td>
</tr>
<tr>
<td>5</td>
<td>3.9</td>
</tr>
</tbody>
</table>
Load on client (bitrate=2048k)

- **PC1 (VLC to play streaming)**

  CPU usage on client PC1

  Packet receiving load on client PC1

- **Rasp.Pi (omxplayer to play streaming)**

  CPU usage on client RPi

  Packet receiving load on client RPi
Load on client (bitrate=2048k)

- **PC1 (VLC to play streaming)**
  - CPU usage on client PC1
  - Packet receiving load on client PC1

- **Rasp.Pi (omxplayer to play streaming)**
  - CPU usage on client RPi
  - Packet receiving load on client RPi

**Constant load regardless of # of clients**
Load on server/FFmpeg (bitrate=8192k)

Constant load regardless of # of clients
Load on clients (bitrate=8192k)

- **PC1** (VLC to play streaming)

  CPU usage on client PC1

<table>
<thead>
<tr>
<th># of clients</th>
<th>CPU usage (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>41.2</td>
</tr>
<tr>
<td>2</td>
<td>38.7</td>
</tr>
<tr>
<td>3</td>
<td>38.7</td>
</tr>
<tr>
<td>4</td>
<td>39.4</td>
</tr>
<tr>
<td>5</td>
<td>39.8</td>
</tr>
</tbody>
</table>

  Packet receiving load on client PC1

<table>
<thead>
<tr>
<th># of clients</th>
<th>Packet receiving load (Mbps)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>12.1</td>
</tr>
<tr>
<td>2</td>
<td>12.0</td>
</tr>
<tr>
<td>3</td>
<td>12.2</td>
</tr>
<tr>
<td>4</td>
<td>12.3</td>
</tr>
<tr>
<td>5</td>
<td>12.1</td>
</tr>
</tbody>
</table>

- **Rasp.Pi** (omxplayer to play streaming)

  CPU usage on client PC1

<table>
<thead>
<tr>
<th># of clients</th>
<th>CPU usage (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>15.9</td>
</tr>
<tr>
<td>2</td>
<td>14.0</td>
</tr>
<tr>
<td>3</td>
<td>14.1</td>
</tr>
<tr>
<td>4</td>
<td>14.5</td>
</tr>
<tr>
<td>5</td>
<td>13.6</td>
</tr>
</tbody>
</table>

  Packet receiving load on client PC1

<table>
<thead>
<tr>
<th># of clients</th>
<th>Packet receiving load (Mbps)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>12.0</td>
</tr>
<tr>
<td>2</td>
<td>11.8</td>
</tr>
<tr>
<td>3</td>
<td>12.1</td>
</tr>
<tr>
<td>4</td>
<td>12.1</td>
</tr>
<tr>
<td>5</td>
<td>12.5</td>
</tr>
</tbody>
</table>
Duplicating packets on multicast network

Data size is constant regardless of # of clients

Server (PC5 or Rasp.Pi) pkt

Switching Hub

Duplicated packets according to # of clients

pkt1 pkt2 pkt3 pkt4 pkt5

Client (PC1) Client (PC2) Client (PC3) Client (PC4)

Client (Rasp.Pi or PC5)

Duplication of packets confirmed on server/client using tcpdump

No additional load for multicast thanks to packet duplication
Max streaming bitrate in different streaming S/W

- **VLC**
  - Skipping/lagging detected if the bitrate is more than 2048 kbps
  - Quality of playing video is quite low; the image is coarse

- **FFmpeg**
  - PC1: No skipping/lagging even when bitrate = 15.4Mbps
  - RPi: Lagging detected if bitrate \( \geq 8192 \text{ kbps} \)
  - Playing stops suddenly after few seconds if bitrate = 15.4Mbps

Max bitrate for streaming in different conditions (kbps)

<table>
<thead>
<tr>
<th>S/W on server</th>
<th>Bitrate on PC</th>
<th>Bitrate on RPi</th>
</tr>
</thead>
<tbody>
<tr>
<td>VLC</td>
<td>2048</td>
<td>2048</td>
</tr>
<tr>
<td>FFmpeg</td>
<td>15377</td>
<td>8192</td>
</tr>
</tbody>
</table>
Why skipping/lagging?

• Narrow network bandwidth?
  – Bandwidth of Raspberry pi = 57Mbps
  – Enough for streaming the video (bitrate = 15.4Mbps)

• Small OS UDP buffer?
  – Increased UDP buffer, but showed no improvement

• PC/RPi’s too low CPU/GPU power?
  – The video played successfully on PC/RPi as a local file
  – “vlc -v underwater-video.mp4”

• Difference between UDP and RTP?
  – Changed btw. UDP and RTP, but showed no improvement

• Missing parameters for VLC/FFmpeg on server?

• Missing parameters for VLC/omxplayer on client?
  – Players on clients need more buffer
Conclusion

• Studied how to multicast-stream on Linux
  – FFmpeg or VLC to stream a video

• Built Video-streaming system using Raspberry pi
  – Multicast network for streaming
  – FFmpeg/VLC to stream a video
  – Parameters for multicast-streaming
  – Raspberry pi with omxplayer to play streaming

• Evaluated capability to stream/play video
  – Constant server/client load for streaming regardless of # of clients
  – Poor receiving capability in Raspberry pi
    • Lagging detected with FFmpeg if bitrate $\geq 8192$ kbps
  – Poor streaming capability of VLC on server
    • Skipping/Lagging detected with VLC if bitrate $> 2048$ kbps
Observation of multicast packets (Appx)

- **Duplicated ICMP packets in multicasting**
  - Multicast ping from PC5(server) to each client
  - PC5(server)
    14:06:06.316506 IP 192.168.0.5 > 224.0.0.1: ICMP echo request, id 4045, seq 1, length 64
    14:06:06.316722 IP 192.168.0.1 > 192.168.0.5: ICMP echo reply, id 4045, seq 1, length 64
    14:06:06.316810 IP 192.168.0.3 > 192.168.0.5: ICMP echo reply, id 4045, seq 1, length 64
    14:06:06.316821 IP 192.168.0.4 > 192.168.0.5: ICMP echo reply, id 4045, seq 1, length 64
    14:06:06.316826 IP 192.168.0.2 > 192.168.0.5: ICMP echo reply, id 4045, seq 1, length 64
    14:06:06.317017 IP 192.168.0.6 > 192.168.0.5: ICMP echo reply, id 4045, seq 1, length 64
  - PC1(client)
    14:00:56.075160 IP 192.168.0.5 > 224.0.0.1: ICMP echo request, id 4045, seq 1, length 64
    14:00:56.075194 IP 192.168.0.1 > 192.168.0.5: ICMP echo reply, id 4045, seq 1, length 64
  - PC2(client)
    14:08:13.366629 IP 192.168.0.5 > 224.0.0.1: ICMP echo request, id 4045, seq 1, length 64
    14:08:13.366680 IP 192.168.0.2 > 192.168.0.5: ICMP echo reply, id 4045, seq 1, length 64
  - PC3(client)
    14:00:38.435625 IP 192.168.0.5 > 224.0.0.1: ICMP echo request, id 4045, seq 1, length 64
    14:00:38.435711 IP 192.168.0.3 > 192.168.0.5: ICMP echo reply, id 4045, seq 1, length 64
  - PC4(client)
    14:07:55.179511 IP 192.168.0.5 > 224.0.0.1: ICMP echo request, id 4045, seq 1, length 64
    14:07:55.179570 IP 192.168.0.4 > 192.168.0.5: ICMP echo reply, id 4045, seq 1, length 64
  - Rasp. Pi(client)
    14:30:05.724940 IP 192.168.0.5 > 224.0.0.1: ICMP echo request, id 4045, seq 1, length 64
    14:30:05.725145 IP 192.168.0.6 > 192.168.0.5: ICMP echo reply, id 4045, seq 1, length 64
Observation of multicast packets (Appx)

- **Duplicated streaming packets in multicasting**
  - Multicast streaming from PC5(server) to each client
  - Captured by tcpdump
  - **PC5(server)**
    18:46:29.993813 IP 192.168.0.5.40015 > 224.0.0.1.1234: UDP, length 1328
    18:46:29.993863 IP 192.168.0.5.40015 > 224.0.0.1.1234: UDP, length 1328
  - **PC1(client)**
    18:39:48.396436 IP 192.168.0.5.56971 > 224.0.0.1.1234: UDP, length 1328
    18:39:48.396454 IP 192.168.0.5.56971 > 224.0.0.1.1234: UDP, length 1328
  - **PC2(client)**
    18:49:02.429645 IP 192.168.0.5.40015 > 224.0.0.1.1234: UDP, length 1328
    18:49:02.429676 IP 192.168.0.5.40015 > 224.0.0.1.1234: UDP, length 1328
  - **PC3(client)**
    18:40:15.275972 IP 192.168.0.5.56971 > 224.0.0.1.1234: UDP, length 1328
    18:40:15.276036 IP 192.168.0.5.56971 > 224.0.0.1.1234: UDP, length 1328
  - **PC4(client)**
    18:47:09.161909 IP 192.168.0.5.56971 > 224.0.0.1.1234: UDP, length 1328
    18:47:09.161952 IP 192.168.0.5.56971 > 224.0.0.1.1234: UDP, length 1328
  - **Rasp. Pi(client)**
    19:10:07.008172 IP 192.168.0.5.40015 > 224.0.0.1.1234: UDP, length 1328
    19:10:07.008177 IP 192.168.0.5.40015 > 224.0.0.1.1234: UDP, length 1328