Passive Aggressive Measurement with MGRP

Pavlos Papageorge$^{1,2}$, Justin McCann$^2$, Michael Hicks$^2$

Google$^1$

University of Maryland, College Park$^2$
Choice 1: Passive Measurement

Observing existing traffic

**Efficient** but **inadequate**

Cannot detect when network conditions improve
Choice 2: Active Measurement

Video conference

Standalone measurement tools

**Inefficient**: Bandwidth intensive

**Intrusive**: Probes can interfere with application data
Choice 3: Custom Active Measurement

Video conference

Shape application data for measurement

Efficient but not Modular

Not Reusable: Cannot interchange algorithms
MGRP
The Measurement Manager Protocol

MGRP piggybacks application data inside active probes
MGRP Properties

Enables aggressive probing with passive-like overhead

• End-to-end measurement architecture
  – Schedules probes for transmission
  – Piggybacks application data on probes

• Transparent to applications

• Independent of measurement algorithms

• Easy to adapt existing measurement tools

• Can piggyback data across applications

• $\frac{1}{10}$ sec precision for probe gap generation
Motivation

Why do we need MGRP?

Micro-Benchmarks

Piggybacking is feasible and improves network performance

MGRP

Architecture
Implementation
Step-by-Step Examples

Case Study: MediaNet
MGRP in the Network Stack

- Layer 4 transport protocol
- Implemented in the Linux kernel
MGRP: Step by Step Example

Packets enter a Payload Buffer where they are delayed by a fixed amount while waiting to be piggybacked.

1. Contribute TCP packets

MGRP Sender piggybacks payload on probes
MGRP: Step by Step Example

MGRP Sender piggybacks payload on probes
MGRP: Step by Step Example

1. Contribute TCP packets
2. Send probes using the Probe API
3. Fragment and piggyback TCP payload on probes

Packets enter a Payload Buffer where they are delayed by a fixed amount while waiting to be piggybacked.

Probes are not delayed. They are sent immediately.

For every probe we look into the payload buffer for TCP payload to fill the empty padding.

MGRP Sender piggybacks payload on probes
MGRP: Step by Step Example

Send MGRP packets

(i) Packet train with 2 empty probes and one partially reused

(ii) Packet train that is fully piggybacked

(iii) The final fragment of the last piggybacked TCP packet

Receive MGRP packets

MGRP packets traverse the network
MGRP: Step by Step Example

MGRP Receiver reconstitutes probes and payload
MGRP: Step by Step Example

MGRP Receiver reconstitutes probes and payload
MGRP: Step by Step Example

1. Send probes using the Probe API
2. Contribute TCP packets
3. Fragment and piggyback TCP payload on probes
4. Send MGRP packets

(i) Packet train with 2 empty probes and one partially reused
(ii) Packet train that is fully piggybacked
(iii) The final fragment of the last piggybacked TCP packet
Outline

Motivation
Why do we need MGRP?

Micro-Benchmarks
Piggybacking is feasible and improves network performance

MGRP
Architecture
Implementation
Step-by-Step Examples

Case Study: MediaNet
Available Bandwidth Tool

• **Pathload**: An active measurement tool
  • Measures end-to-end available bandwidth
  • By Jain & Dovrolis at Georgia Tech

• Good candidate for our evaluation
  – Available bandwidth is a very useful network property
  – Quite accurate (even for GigE speeds, PAM05)
  – Non-trivial overhead (we can test probe reuse)
Bandwidth Timeseries with Pathload

Graph showing bandwidth over time with different traffic types such as cross traffic, probe (not piggybacked), and TCP. The graph compares bandwidth over a sequence of seconds.
Effective Probe Overhead is Minimal

STEP: pathload pFAST
Pathload Completes Faster

CDF of pathload completion times: pFAST

CDF of pathload completion times: pSLOW

STEP: pathload
Pathload Completes More Often

CDF of pathload completion times: pFAST

CDF of pathload completion times: pSLOW

STEP: pathload
Benefits of MGRP

• Saves bandwidth
  – Reduces measurement overhead
  – Fewer probes compete with application data

• Allows measurement tools to:
  – Send more probes
  – Send probes continuously
  – Complete faster and be more accurate
Lessons Learned

• Measurement tools need to be adjusted
  – Must account for piggybacked traffic

• Blind piggybacking can be harmful
  – Pigybacked packets share fate of probes
  – Some probes have high loss risk

• Long MGRP data buffers may affect TCP
  – Need to keep latency small fraction of RTT
Outline

Motivation
Why do we need MGRP?

Micro-Benchmarks
Piggybacking is feasible and improves network performance

MGRP
Architecture
Implementation
Step-by-Step Examples

Case Study: MediaNet
Case Study: MediaNet Overlay

- Streams MPEG video at different rates
- **Overlay nodes** report if they can send at desired rate
- **Pathload** continuously monitors the paths

(a) MediaNet can modify the streaming rate

<table>
<thead>
<tr>
<th>Frame Type</th>
<th>Frame Size (bytes)</th>
<th>Frequency (frames/sec)</th>
<th>High Rate (Kbps)</th>
<th>Medium Rate (Kbps)</th>
<th>Low Rate (Kbps)</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>13500</td>
<td>2</td>
<td></td>
<td>1200</td>
<td>200</td>
</tr>
<tr>
<td>P</td>
<td>7625</td>
<td>8</td>
<td>700</td>
<td></td>
<td>Dropped</td>
</tr>
<tr>
<td>B</td>
<td>2850</td>
<td>20</td>
<td></td>
<td>Dropped</td>
<td>Dropped</td>
</tr>
</tbody>
</table>
Without probes MediaNet cannot react
But probes interfere without MGRP
MGRP improves the stream quality

(a) Medianet’s View of the Network

(b) MPEG streaming frames/sec

decoded frames/sec

![Graphs showing network bandwidth and decoded frames per second over time.](image)
MGRP Improves MediaNet
The aggregate MPEG streaming rate is higher

<table>
<thead>
<tr>
<th>Experiment</th>
<th>Runs</th>
<th>AverageRun Duration (sec)</th>
<th>Aggregate Streaming Rate (Mbps)</th>
<th>Improvement over non-MGRP</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Original MediaNet</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(mgrpOFF/pOFF)</td>
<td>14</td>
<td>337</td>
<td>1.84</td>
<td></td>
</tr>
<tr>
<td><strong>Pathload</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pSLOW</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>mgrpOFF</td>
<td>22</td>
<td>336</td>
<td>1.96</td>
<td></td>
</tr>
<tr>
<td>mgrp10</td>
<td>32</td>
<td>336</td>
<td>2.05</td>
<td><strong>4.40%</strong></td>
</tr>
<tr>
<td><strong>Pathload</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pFAST</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>mgrpOFF</td>
<td>10</td>
<td>335</td>
<td>1.86</td>
<td></td>
</tr>
<tr>
<td>mgrp10</td>
<td>22</td>
<td>336</td>
<td>2.28</td>
<td><strong>22.52%</strong></td>
</tr>
</tbody>
</table>

Related to the quality of the playback
MGRP Improves MediaNet
The number of decoded MPEG frames increases

<table>
<thead>
<tr>
<th>Experiment</th>
<th>Runs</th>
<th>Average Run Duration (sec)</th>
<th>Aggregate Frames per Second</th>
<th>Improvement over non-MGRP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Original MediaNet (mgrpOFF/pOFF)</td>
<td>14</td>
<td>337</td>
<td>30.11</td>
<td></td>
</tr>
<tr>
<td>Pathload pSLOW</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>mgrpOFF</td>
<td>22</td>
<td>336</td>
<td>39.58</td>
<td></td>
</tr>
<tr>
<td>mgrp10</td>
<td>32</td>
<td>336</td>
<td>43.42</td>
<td><strong>9.69%</strong></td>
</tr>
<tr>
<td>Pathload pFAST</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>mgrpOFF</td>
<td>10</td>
<td>335</td>
<td>39.10</td>
<td></td>
</tr>
<tr>
<td>mgrp10</td>
<td>22</td>
<td>336</td>
<td>52.08</td>
<td><strong>33.19%</strong></td>
</tr>
</tbody>
</table>

Directly affects the quality of the playback
How MGRP stands out

MGRP is a new protocol that piggybacks application data inside probes.

Piggybacking reduces bandwidth wasted by probes and enables measurement tools to be more aggressive, faster and more accurate.

Any measurement algorithm can now be written as if active, but implemented as passive.

MGRP is generic and is transparent to applications
Questions?
Additional Slides
MGRP: Next Steps

- Add support for:
  - ICMP packets
  - TTL limited packets

- Automatically set piggybacking ratio

- Enable one-way probing with remote timestamp collection
MGRP Packet Format

MGRP Header

MGRP Packet

Payload of TUN 0
(PROBE | TCP | FRAG)

Payload of TUN 1
(optional, TCP | FRAG)
MGRP Piggybacking

(a) MGRP probe with no piggybacking

(b) MGRP probe fully piggybacked

(c) MGRP probe partially piggybacked
Characteristics of Active Probes

• Have varying sizes
• Need precise inter-packet gaps
• Are largely empty padding
• Usually sent in groups
• More probes: better/faster results

• Can probing be aggressive without the overhead?
Piggybacking requires that we adjust Pathload
Piggybacking reduces the probing overhead

STEP: probe train pk2
Effective Probe Overhead is Minimal

STEP: probe train pk2
Piggybacking may be too Aggressive
Too many piggybacked packets get lost
Solution: Reduce the Piggybacking Ratio

WEB: pathload pFAST
So that High Risk Probes are Avoided

WEB: pathload pFAST