HTTP-like Streams for 9P

John Floren

Rochester Institute of Technology

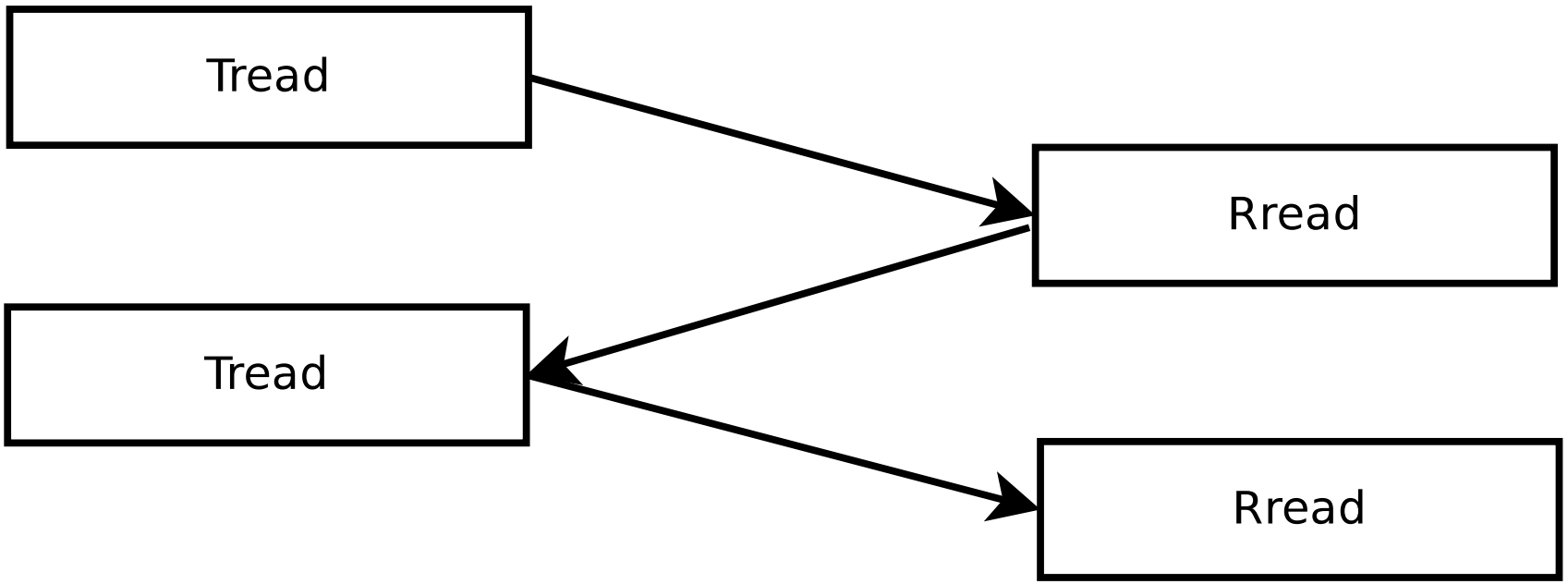
October 5, 2010

With Ron Minnich and Andrey Mirtchovski

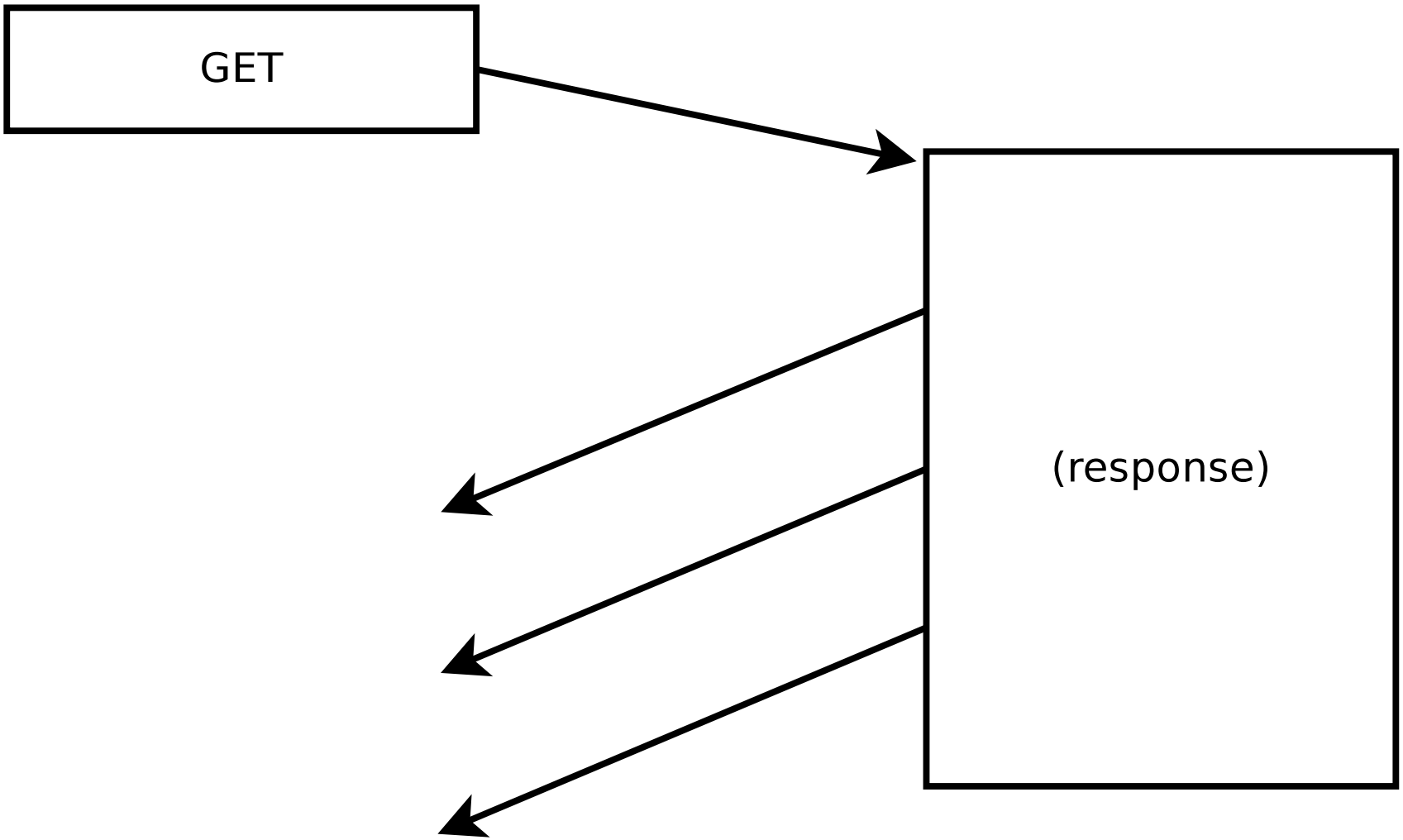
Introduction

- Copying files via 9P is slow.
- Why? 9P waits for a single response for every message sent.
- Over high-latency links, the waiting becomes problematic.
  - Send a `Tread` and start waiting
  - 50 ms later, `Tread` arrives at server, which responds
  - Another 50 ms later, the `Rread` arrives
- We spend 100 ms waiting for each chunk of data read!
- `fcp` is not the answer; you shouldn't have to do threading just to read a file.
- We frequently read files sequentially, but 9P doesn't care. Each time we have to specifically ask for the next chunk of data and suffer the associated latency.

The 9P Model

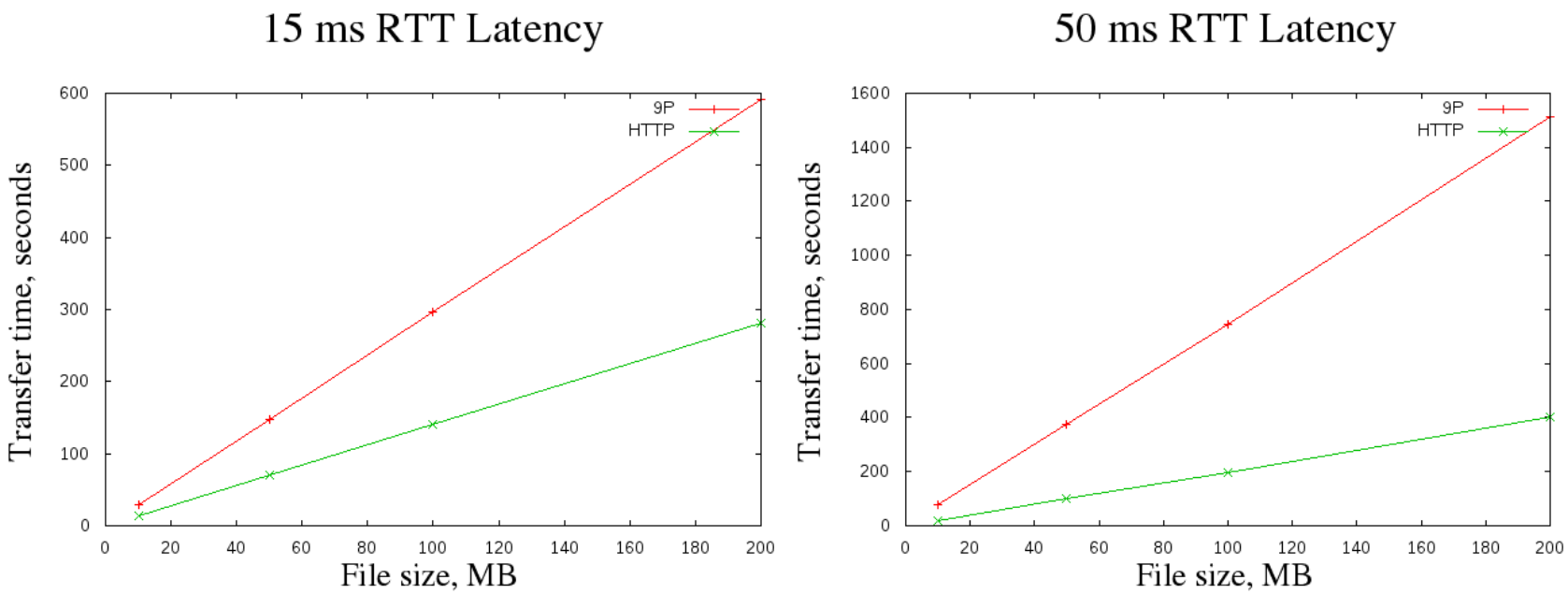


The HTTP Model



Motivation

We compared HTTP and 9P for transferring files over high-latency links.

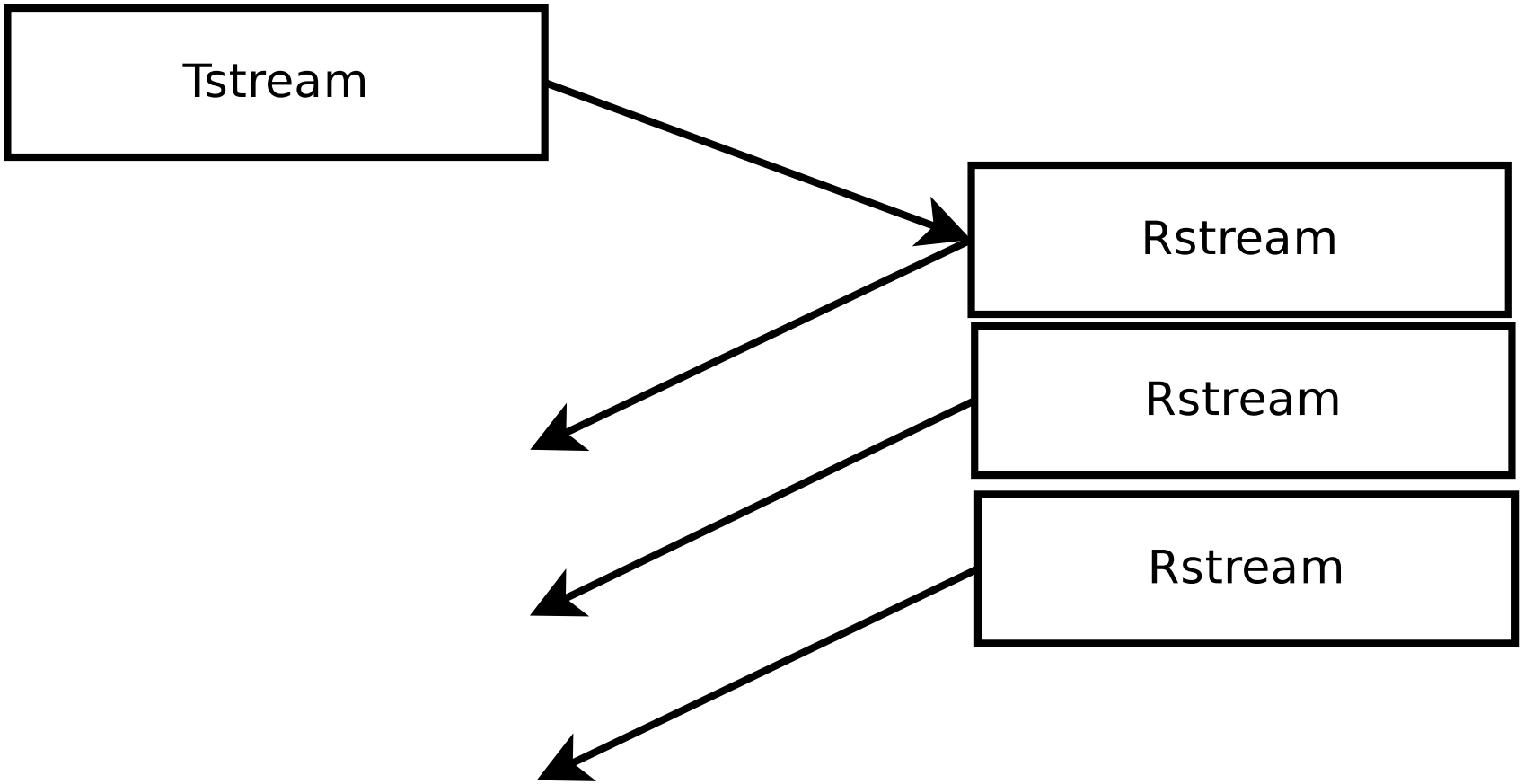


Copying a 200 MB file over 9P across a 50ms RTT link takes more than 25 minutes. The same operation requires less than 7 minutes with HTTP.

Streams

- Why not give programmers the option of having HTTP-like behavior?
- Streams aim to do just that
- Throw out the `Tmessage/Rmessage/Tmessage/Rmessage` paradigm
- Instead, send one `Tstream`, get back a lot of `Rstreams`
  - With TCP, we already have in-order, guaranteed delivery with flow control.
  - Incoming `Rstream` messages just wait in the queue to be read

The Streams Model



Implementation

- So far, there are two new system calls, `stream` and `sread`
  - `stream(fd, offset)` causes a `Tstream` to be sent out.
  - `sread(fd, buffer, size)` reads data from `Rstream` messages into a buffer.
- `devmnt` is sending `Tstream` messages.
- `exportfs` is replying with `Rstream` messages.
- The kernel is crashing when `sread` tries to read `Rstream` messages.

Conclusions

The Streams system is on its way to completion. We hope it may allow programmers to perform sequential reading (and eventually writing) without interfering with the behavior or use of the traditional I/O system.