Google QUIC over Satellite Links

PANRG Interim

June 3, 2020

John Border – Hughes Network Systems

John.Border@Hughes.com

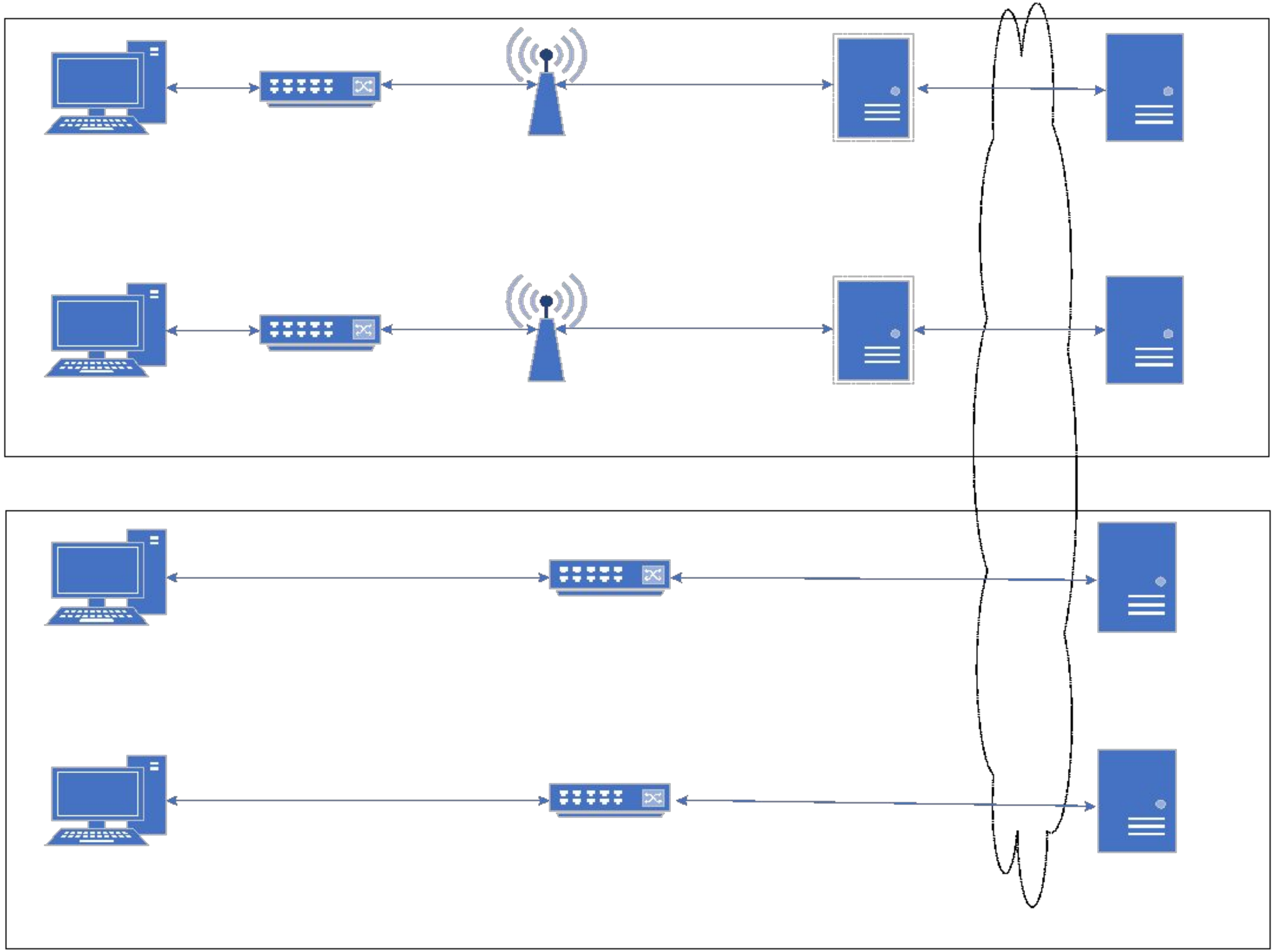
Problem Statement

- Transport layer protocols such asTCP do not perform well enough over high delay bandwidth product links like GEO satellites links without any modifications
  - Long latency impacts error recovery not just window sizes
- Hughes implements a split-TCP PEP to improve network performance over such links
- QUIC is a new transport layer protocol, originally developed by Google and now being standardized by the IETF
- QUIC can’t be split in the same way as TCP and hence may suffer from relatively poor performance (when compared to PEP-ed TCP) on high latency links
- Testing being done to see how big the performance disparity is

Test Setup

- Google Drive Server
- Google Chrome Client (v75.0.3770)
- Puppeteer library used to automate testing
    - Node.js script browses to Google Drive and downloads the specified file
    - Watcher setup for changes to the download directory by using fs.watch API
        - Listens for eventType= change for the file being downloaded and keeps checking that file’s size
        - Watcher initiated when the file is clicked for download and closed when downloaded file’s size is equal to the expected file size
    - Start and end time captured by the script when the watcher starts and closes
- Multiple (typically 100) runs for each test

Setup

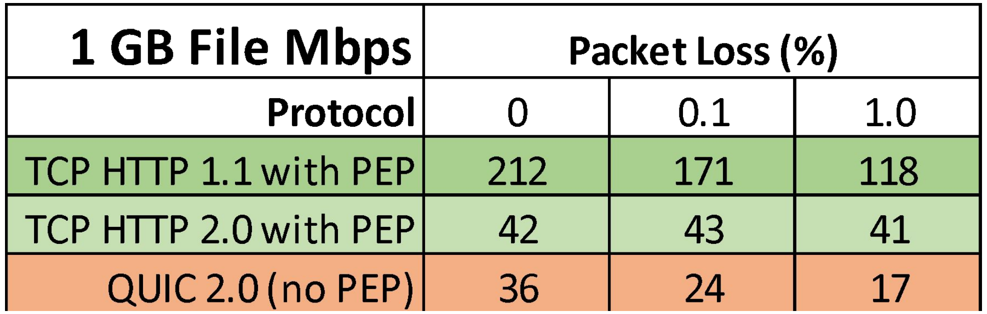


Testing Variants

- Protocols
  - HTTP/1.1 over TCP (--disable-http2 flag)
  - HTTP/2 over TCP (--disable-quic flag)
  - HTTP/2 over QUIC (--enable-quic flag)
- Testbed
  - 1 Gbps connection to the Internet
  - Delay box simulating satellite delay
  - Two variants
    - Going through Hughes' terminal and gateway
      - Spoofed TCP and QUIC
    - Bypassing Hughes equipment
      - Unspoofed TCP and QUIC
- Files Sizes – 0.5 GB, 1.0 GB and 1.5 GB
- Packet Loss Rates – 0%, 0.1% and 1%
  - At the Delay Box
  - We tried 10% but the results were both bad and inconsistent

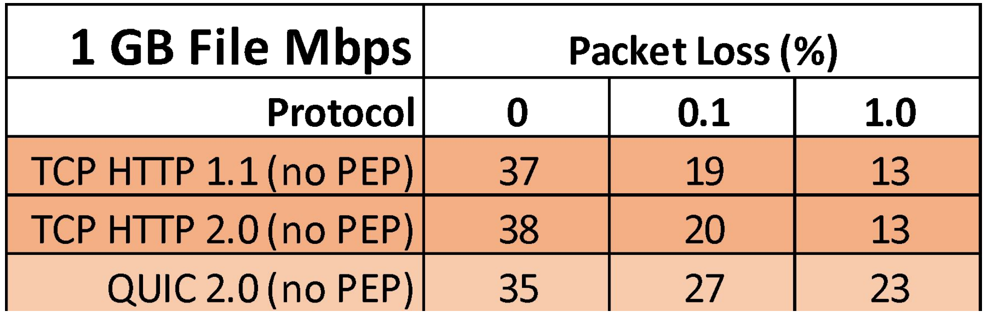
Results

- Through Hughes equipment (with TCP PEP)
- 600 ms Latency



1 GB File Mbps	Packet Loss (%)		
Protocol	0	0.1	1.0
TCP HTTP 1.1 with PEP	212	171	118
TCP HTTP 2.0 with PEP	42	43	41
QUIC 2.0 (no PEP)	36	24	17

- Direct path (with no PEP)
- 600 ms Latency

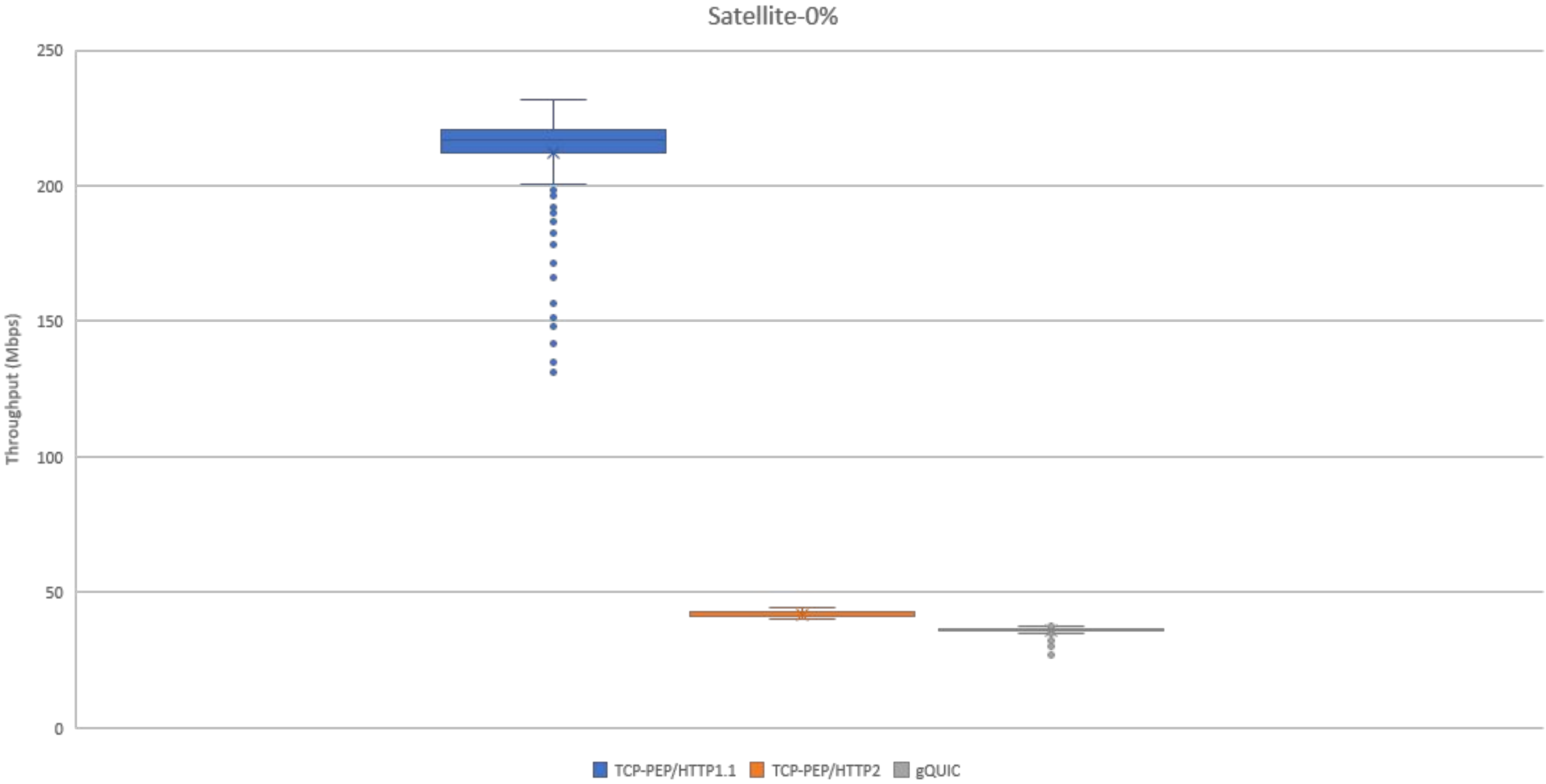


1 GB File Mbps	Packet Loss (%)		
Protocol	0	0.1	1.0
TCP HTTP 1.1 (no PEP)	37	19	13
TCP HTTP 2.0 (no PEP)	38	20	13
QUIC 2.0 (no PEP)	35	27	23

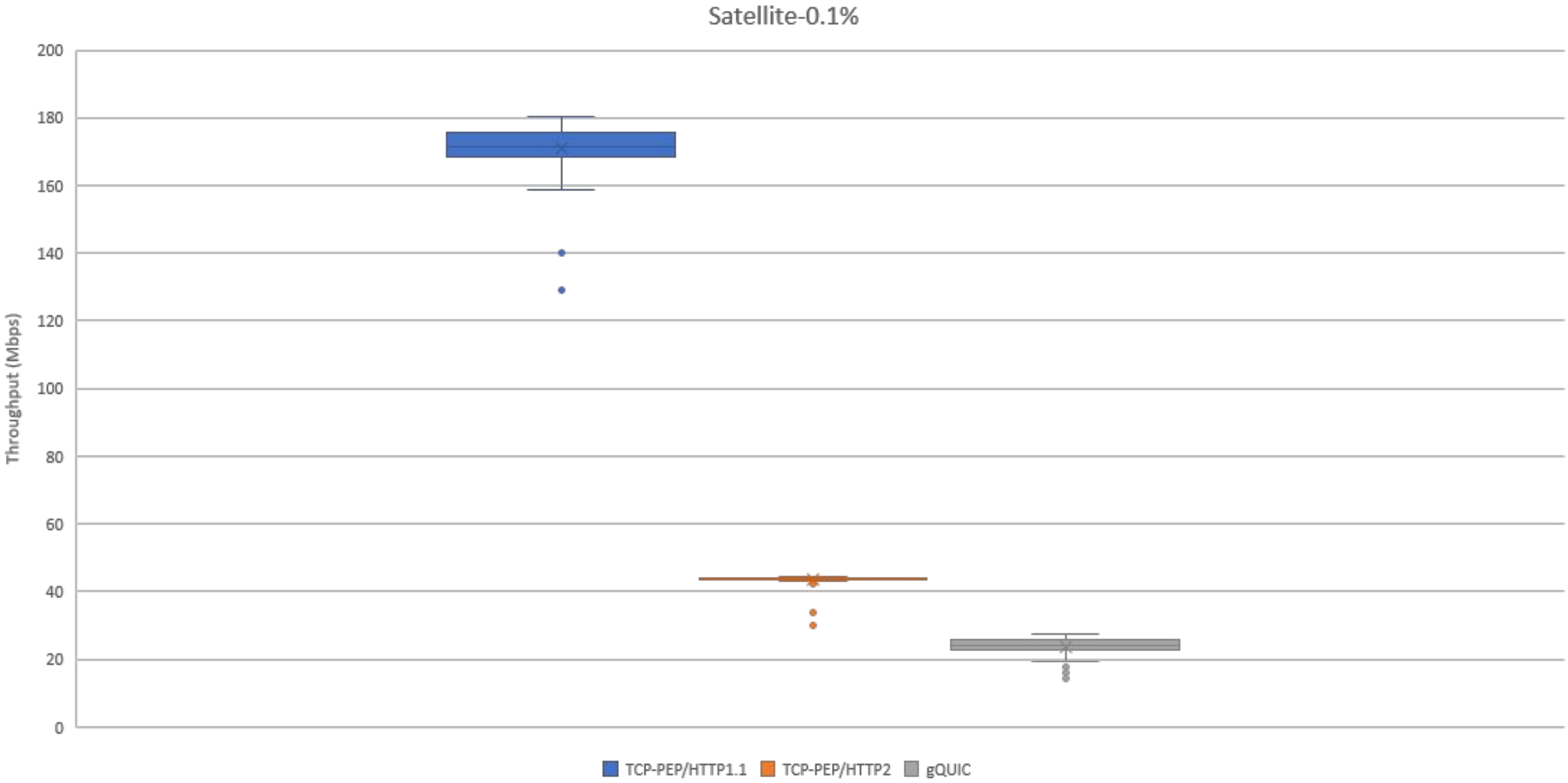
Bonus Material

Results in Graph Form

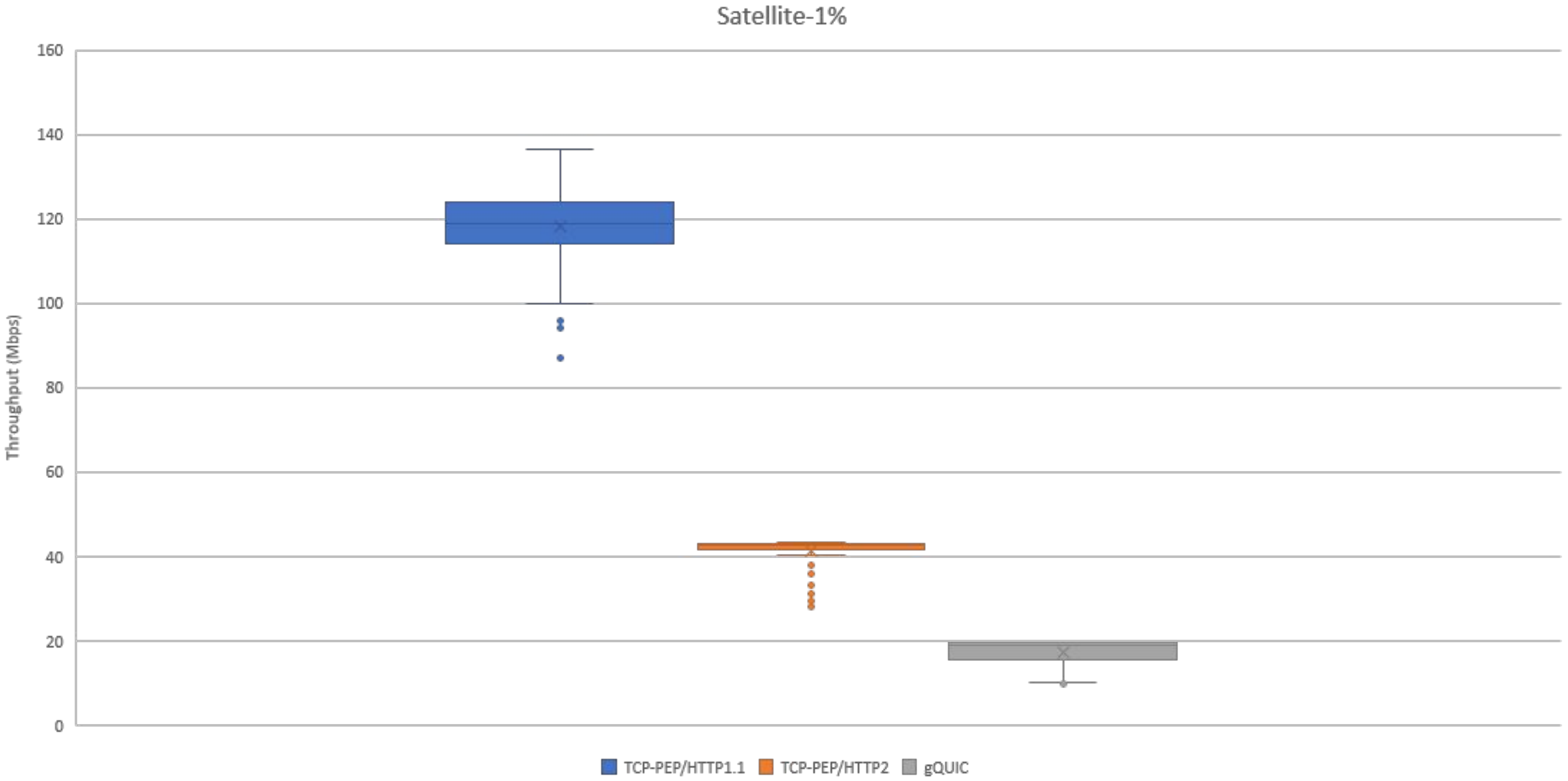
1.0 GB File Results Sample with Hughes PEP – No Packet Loss



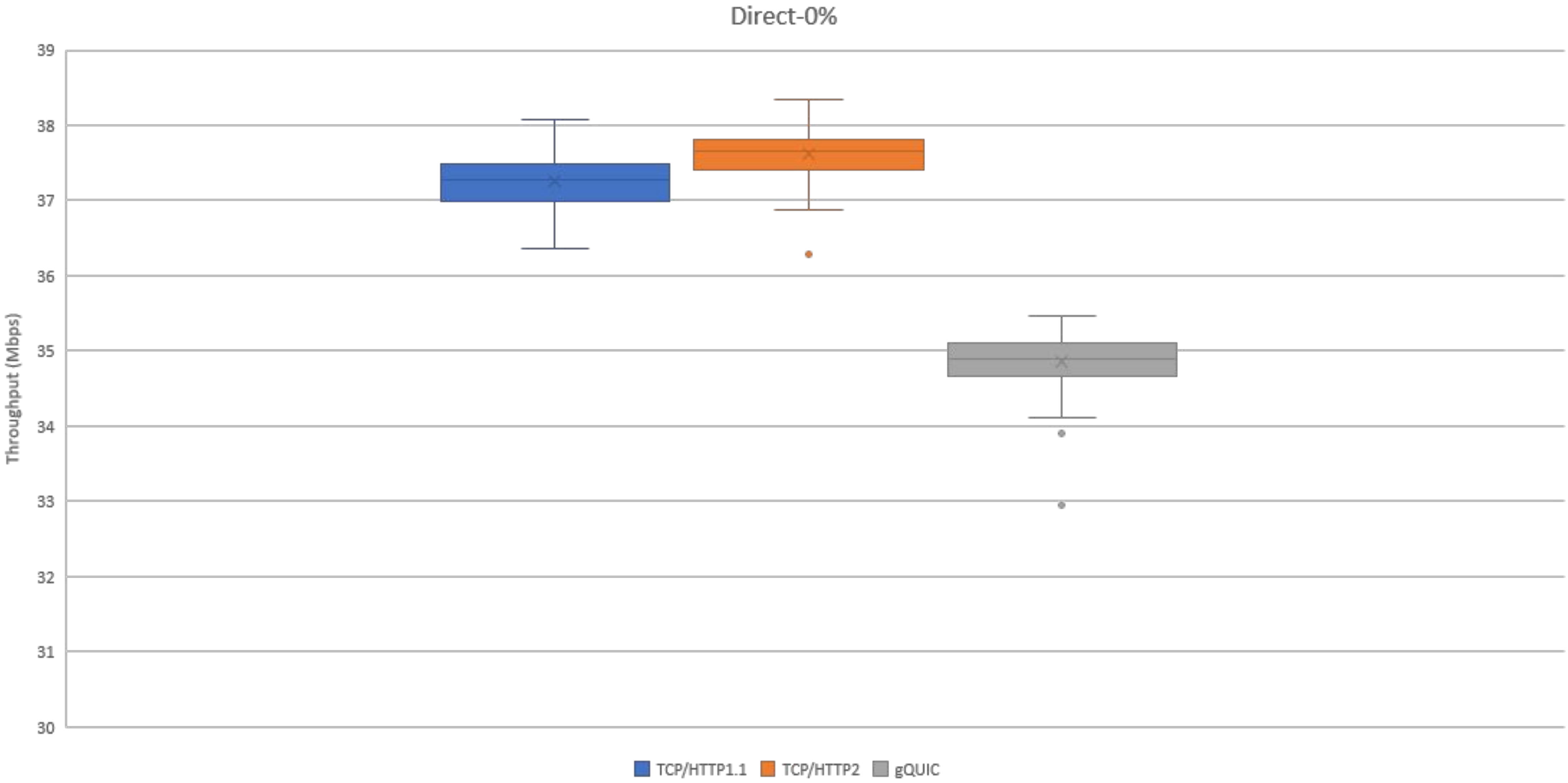
1.0 GB File Results Sample with Hughes PEP – 0.1% Packet Loss



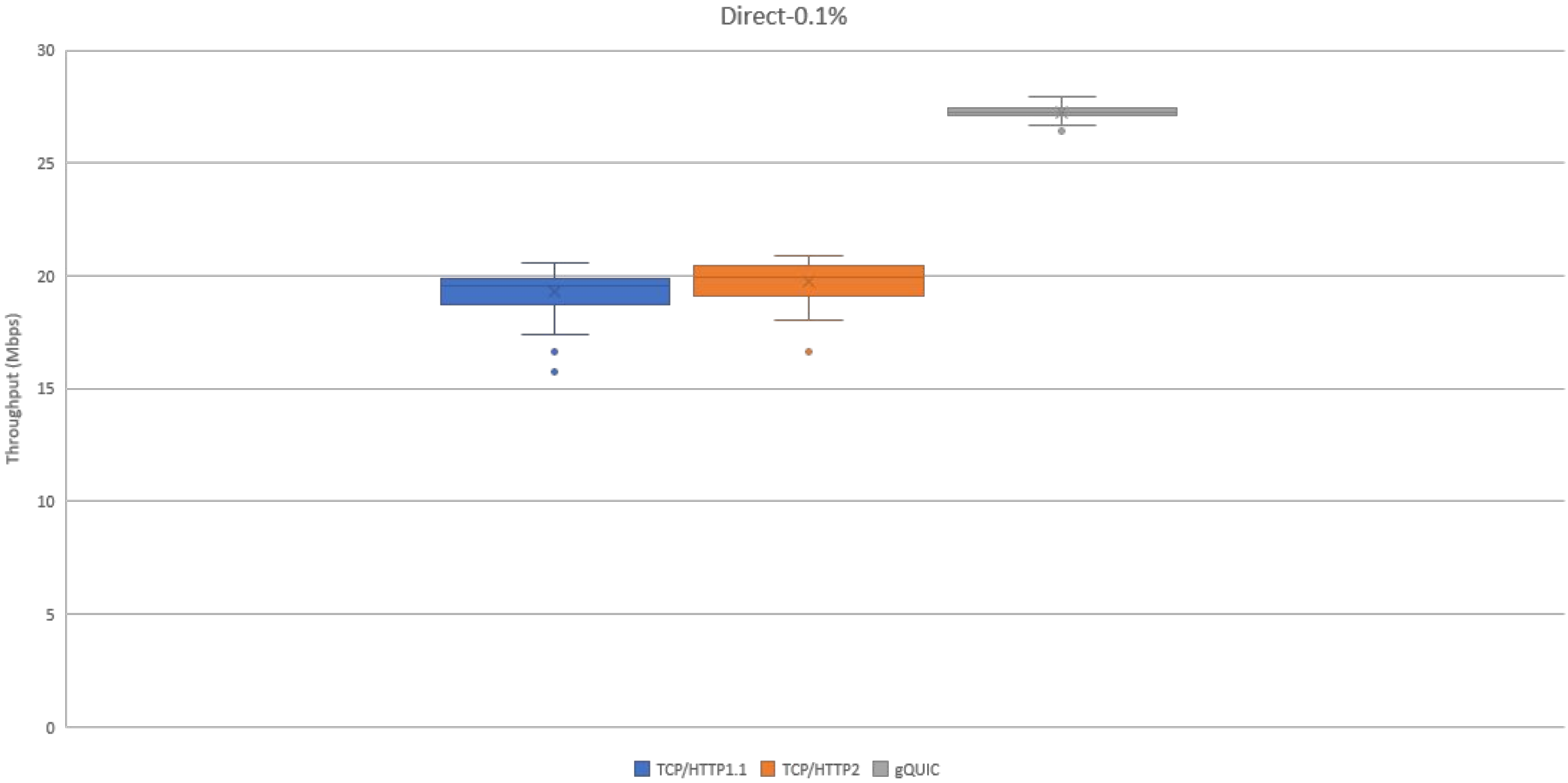
1.0 GB File Results Sample with Hughes PEP – 1% Packet Loss



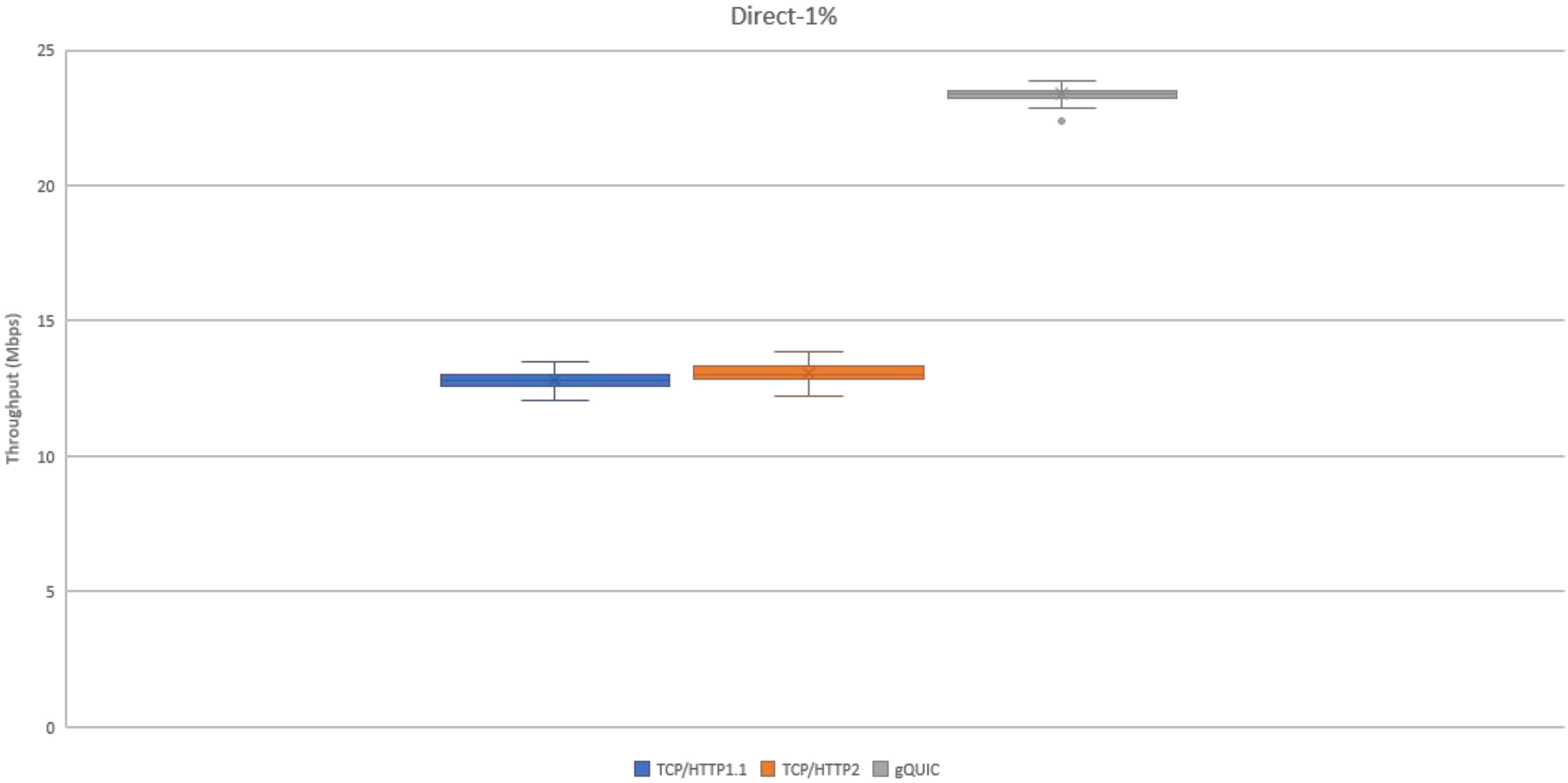
1.0 GB File Results Sample Direct – No Packet Loss



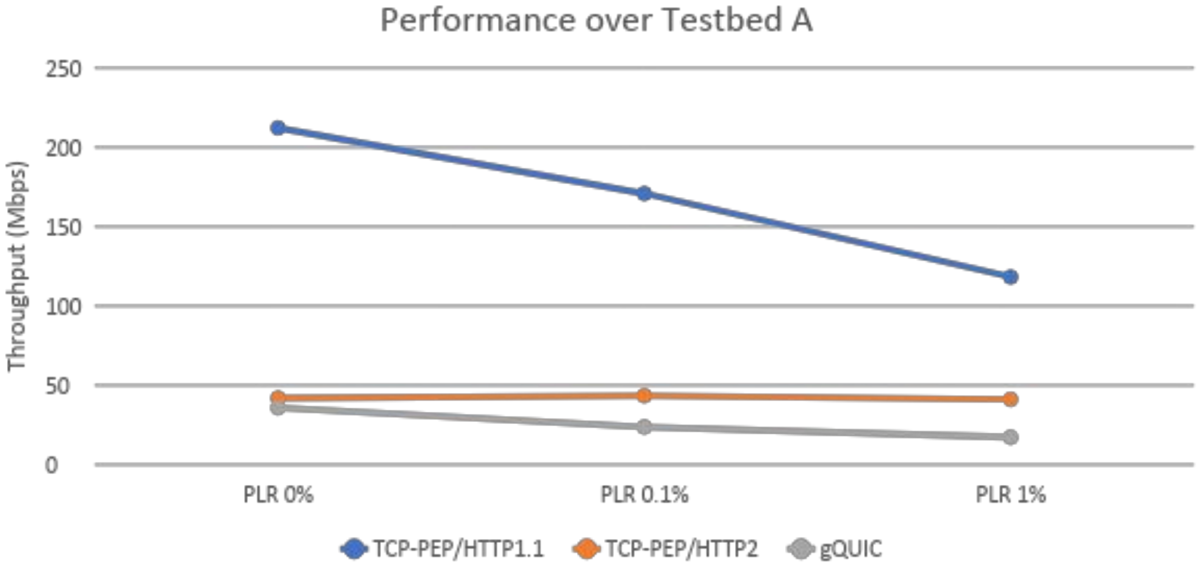
1.0 GB File Results Sample Direct – 0.1% Packet Loss

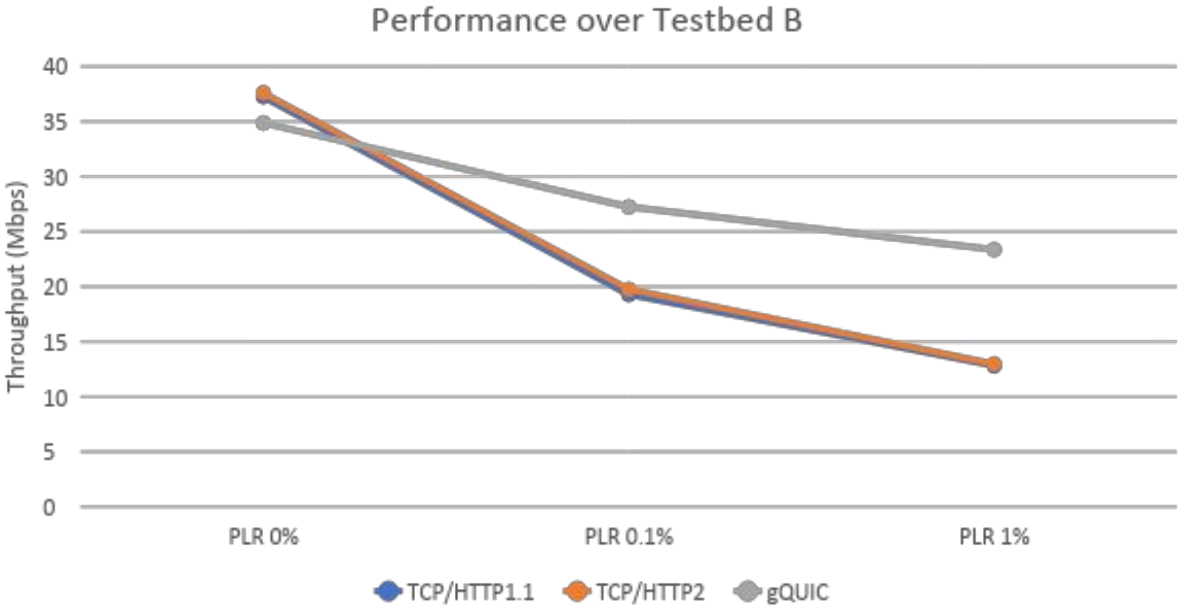


1.0 GB File Results Sample Direct – 1% Packet Loss



Performance across Testbeds


Performance over Testbed A
Throughput (Mbps)
PLR 0%
PLR 0.1%
PLR 1%
TCP-PEP/HTTP1.1
TCP-PEP/HTTP2
gQUIC


Performance over Testbed B
Throughput (Mbps)
PLR 0%
PLR 0.1%
PLR 1%
TCP/HTTP1.1
TCP/HTTP2
gQUIC