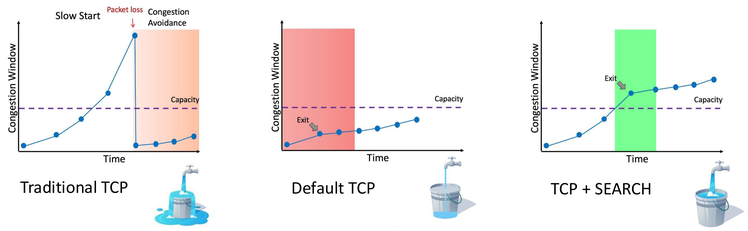
Implementation of TCP SEARCH in FreeBSD and Evaluation on a Satellite Network

Maryam Ataei Kachooei, Samuel Ollari, Benjamin Skarnes, Jae Chung, Amber Cronin, Feng Li, Benjamin Peters, Mark Claypool

Introduction



➢ SEARCH implemented in:

Linux and QUIC.

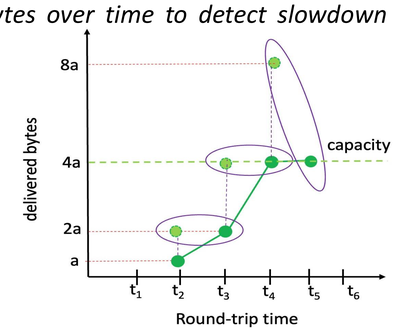
First ported to FreeBSD in this work.

SEARCH Algorithm Overview

SEARCH Link: https://search-ss.wpi.edu/

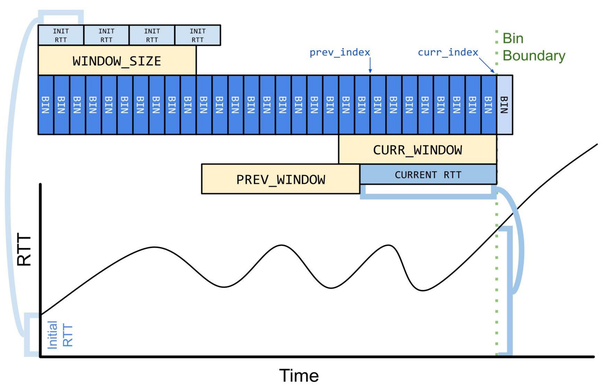
SEARCH compares delivered bytes over time to detect slowdown before congestion.

- Tracks delivery rate across RTT-based bins.
- Detects when delivery rate stops growing.
- Exits slow start when delivery slows near capacity.



SEARCH maintains a sliding window of delivery bins across multiple RTTs.

- Maintains two sliding windows, each **3.5 × initial RTT**.
- Total of **25 bins**, each storing **ACKed bytes** for a fixed interval.
- **curr_window** and **prev_window** identified by bin indices.
- **prev_window** offset by **1 RTT** from **curr_window**.
- Computes normalized difference:
  norm = (2 × prev_window − curr_window) / (2 × prev_window).
- Exits slow start when: **norm ≥ 0.35**.



Implementation in FreeBSD

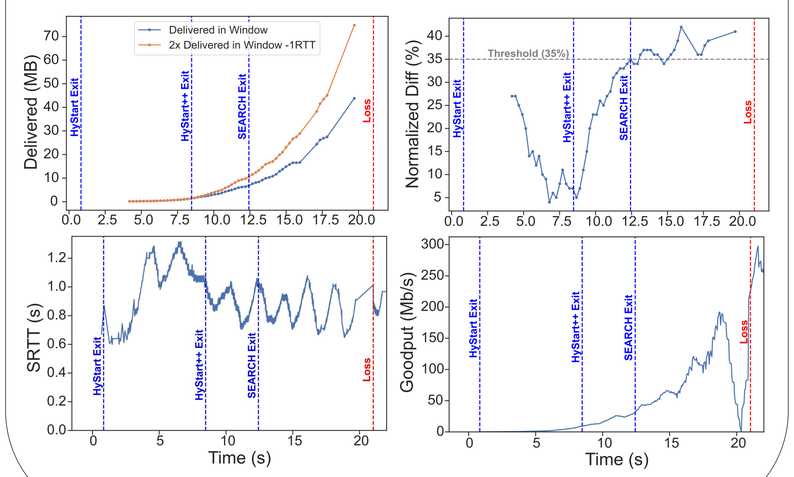
- Implemented as a **modular congestion control algorithm** in **FreeBSD 14**, extending **NewReno**.
- Integrated into the **`ack_received`** callback to track delivered bytes on each ACK.
- Uses the **initial RTT** (via `t_srtt`) to configure the algorithm:
  - **Bin size** = (3.5 × initial RTT) / 10
  - **25 bins** used for tracking current and previous windows.

SEARCH Flow in FreeBSD

```
Incoming ACK
↓
Update delivered bytes in current bin
↓
Check if 10 bins are filled (curr_window ready)
↓
Compare with prev_window (offset by 1 RTT)
↓
Compute normalized differences:
norm = (2 × prev_window − curr_window) /
(2 × prev_window)
↓
If norm >= 0.35 → Exit slow start
```

Case Study Result

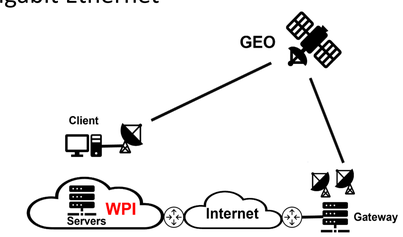
- Compare **SEARCH**, **HyStart**, and **HyStart++** during TCP slow start using real-world measurements on FreeBSD.



Evaluation

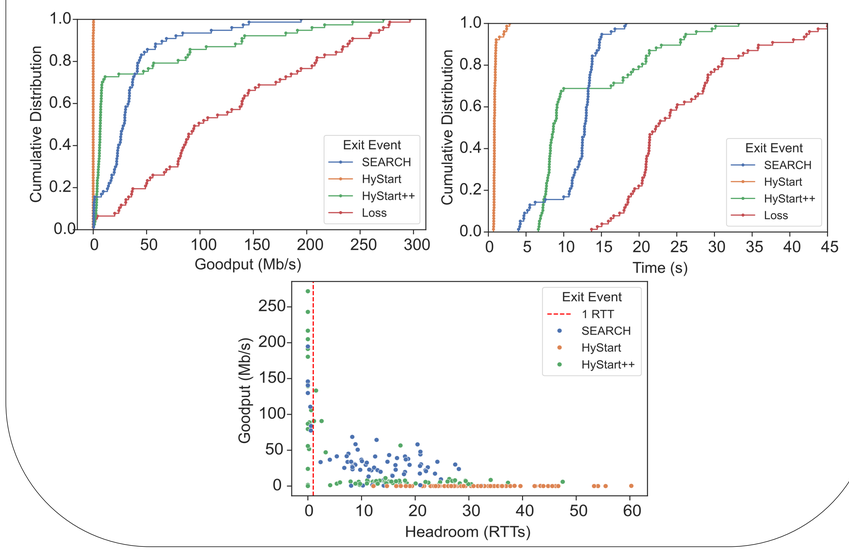
Setup:

- **Client**: Linux machine (Ubuntu 20.04) via **Viasat-2 satellite terminal**
- **Server**: FreeBSD 14 VM over Gigabit Ethernet
- **RTT**: ~600 ms
- **Capacity**: ~150 Mb/s
- **PEP disabled**



Download Details:

- 77 Iperf3 TCP Downloads.
- Duration: 45 seconds.
- Test schedule: 24 hours.



Conclusion & Future Work

- First implementation of SEARCH in FreeBSD (NewReno).
- Tested on a real GEO satellite link.
- Outperforms HyStart/HyStart++: exits at better time, higher throughput, avoids loss.

Future work:

- Test on LEO, 5G, Wi-Fi networks.
- Analyze overshoot risks.
- Deploy on CDN testbeds.
- Contribute to FreeBSD upstream.

References

- https://github.com/Project-Faster/tcp ss search
- https://github.com/Project-Faster/quicly/tree/generic-slowstart
- M. A. Kachooei, J. Chung, F. Li, B. Peters, J. Chung, and M. Claypool,"Improving TCP Slow Start Performance in Wireless Networks with SEARCH," in WoWMoM Symposium, 2024, pp. 279–288.