MPTCP Inactivity Time Option and Subflow Rate Limit Option

Viet-Hoang Tran, Olivier Bonaventure
UC Louvain
MPTCP Inactivity Time Option
MPTCP Inactivity Timeout (ITO)

Client: 3-way handshake (MP_CAPABLE)

Server: TCP-RST

Close subflow

Keep the MPTCP session until when?
MPTCP Inactivity Timeout (ITO)

RFC6824(-bis) does not specify how long MPTCP hosts should maintain idle sessions.

Inactivity time: duration that an MPTCP session has no established subflow.

Keep the MPTCP session until when?
Recommend a Default ITO?

TCP does not recommend a default value for idle connection, but:

RFC1122: TCP KeepAlive >= 2 hours

RFC5382: NAT timeout >= 2 hours + 4 minutes
Use cases

1. Hosts want to keep the session alive through transient failures → Request its peer for an enough ITO.
   
   For TCP, this does not work due to NAT timeout
   
   For MPTCP, NAT is not a problem

2. Highly-loaded servers quickly terminate unused MPTCP sessions by setting a small local ITO.
   → May signal its clients that idle sessions will be closed shortly.
ITO Option Format

<table>
<thead>
<tr>
<th>Kind</th>
<th>Length = 5</th>
<th>Subtype</th>
<th>(rsv)</th>
<th>Inactivity Timeout (16 bit)</th>
</tr>
</thead>
</table>

Timeout Range:
- Min = 0: remove session immediately when there is no active subflow
- Max = $2^{16}-1$ seconds ~ 18 hours

ITO option is **indicative**: Local policy could override this request

ITO option is exchanged **unreliably**

To improve the delivery:
- May send X times per second/RTT/lifetime?
- Or attach it to a Sequence Number
Subflow Rate Limit Option
Motivation

Mobile users usually have limited cellular data quota

They want to use cellular networks, but still need to limit the monetary cost, or reserve the data quota.

But: traffic are mostly downstreamed, which clients cannot control.

→ Client could request the server a max sending rate on a subflow.
Option Format

Requested Rate (32 bits) is specified in IEEE-754 floating-point format

Range: from $1.2 \times 10^{-38}$ to $3.4 \times 10^{38}$

Unit: Kbps

SRL option is indicative and unreliable
Linux Implementation Prototype

Used eBPF to quickly testing
new MPTCP options

Based on TCP-BPF
(in mainline Linux)
Experiment: Capping on second subflow
Request rate-limit of Zero?

Allow peers to disable a subflow temporarily
Open Questions

**Improve reliability**
- May send X times per second/RTT/lifetime?
- Should the server respond to the request?

**Duration of rate-limit policy**
- until the end of connection?
- or allow clients to specify?

**Combine with other use cases?**
- backup when latency/bw satisfied
- traffic ratio among subflows
- cap max amount of data
SRL Option: Security Considerations

Attacker could throttle the rate on a subflow.

But, it could instead drop packets or inject TCP-RST or MP-FASTCLOSE.

Inserting option is one-off, while dropping packets needs continuity.
   For specialized hardware, which one is easier?

Countermeasures

- Use HMAC? cannot protect initial path, but make it harder
- Receivers cap the values in a safe range
ITO Option: Security Consideration

Implementations should define a safe range of values, restricting:

- Local setting by applications
- Received ITO options

May restrict accepting ITO options only from trusted peers.