Multipath TCP: Breaking the Single-Path Limit

Why settle for one path when you can use all of them?

SECTION 1

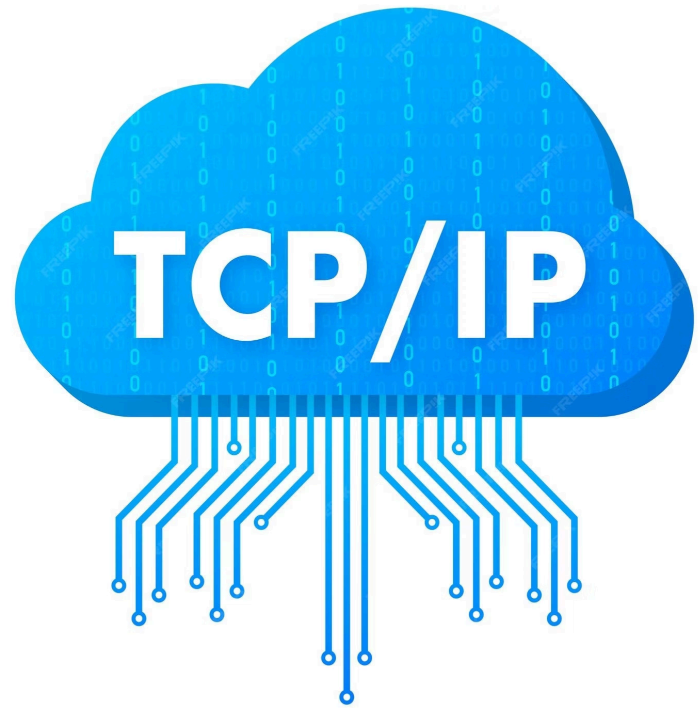
The Problem with Standard TCP

In a world of constant connectivity, standard TCP's limitations create bottlenecks and unreliable experiences, especially for mobile users.

The "One Connection, One Path" Limitation

Standard TCP connections are **rigidly defined** by a 5-tuple: Source IP, Source Port, Dest IP, Dest Port, and Protocol.

- If *any* of these parameters change (e.g., your IP address changes as you move), **the connection breaks**.
- Imagine a single-lane road: if that road is blocked, all traffic stops. Standard TCP operates on this principle, building a **single "pipe"** between two devices.



Why This Matters for Modern Devices

Wasted Bandwidth

Modern devices often have multiple network interfaces simultaneously active (4G/5G and WiFi). Standard TCP can only utilize one at a time, leaving valuable bandwidth unused.

Poor Reliability

When the active link (e.g., a weak WiFi signal) degrades, the connection either hangs or drops entirely, instead of seamlessly switching to a stronger available network.

Handover Latency

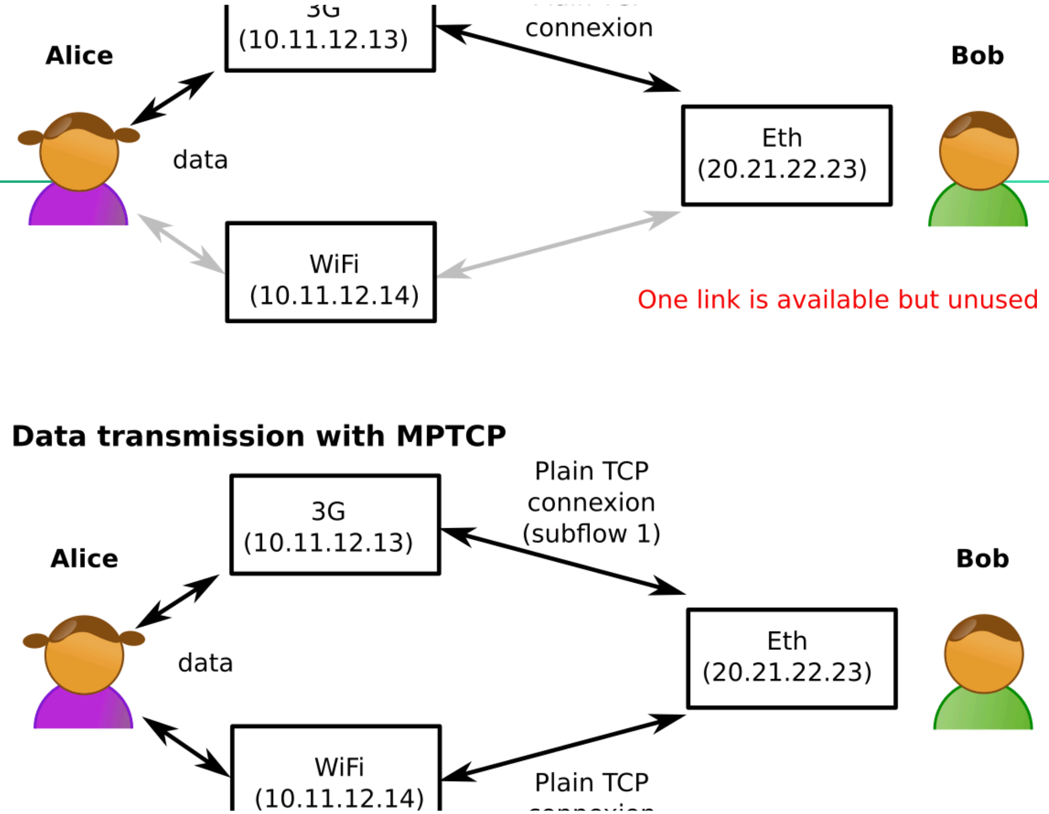
Switching between networks (e.g., from WiFi to cellular) typically requires terminating the existing TCP connection and establishing a brand new one, causing noticeable delays and service interruptions.

SECTION 2

What is Multipath TCP?

Multipath TCP (MPTCP) is an extension to TCP that allows a single connection to use multiple paths simultaneously, enhancing performance and reliability.

Where MPTCP Fits in the OSI Model



Standard TCP

Application → TCP → IP (single path)

MPTCP Stack

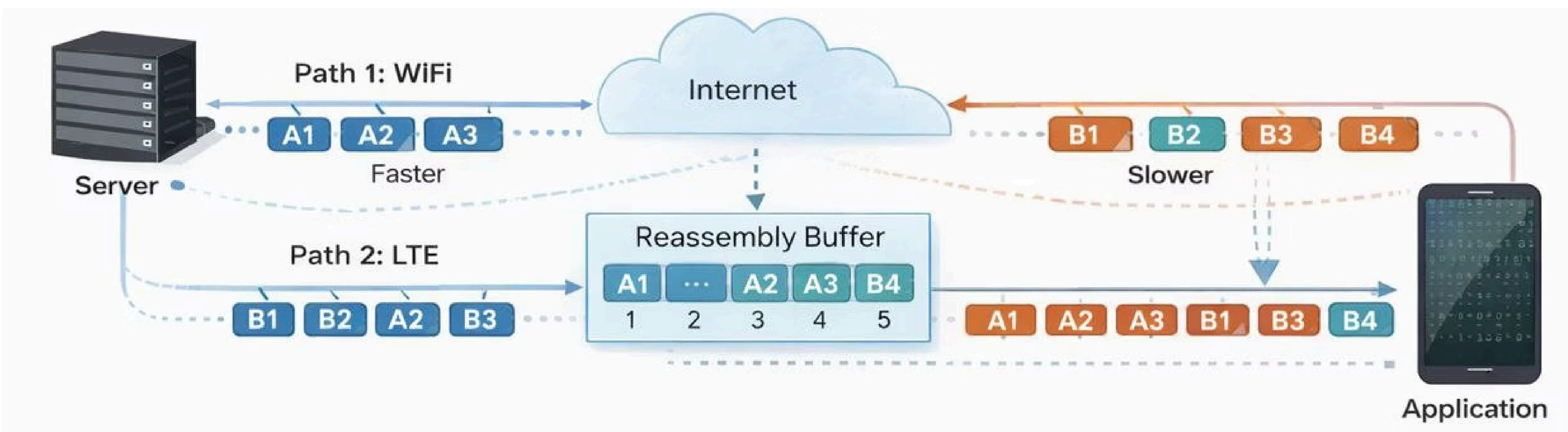
Application → MPTCP → TCP subflows → IP (multiple paths)

MPTCP introduces an abstraction layer above standard TCP, managing multiple "subflows" to provide a unified, resilient connection to applications.

How MPTCP Maintains Data Order: Data Sequence Number (DSN)

While Multipath TCP utilizes multiple TCP subflows, each with its own internal sequence numbering, it introduces an overarching Data Sequence Number (DSN) at the MPTCP layer to ensure proper reordering and delivery of data across all paths.

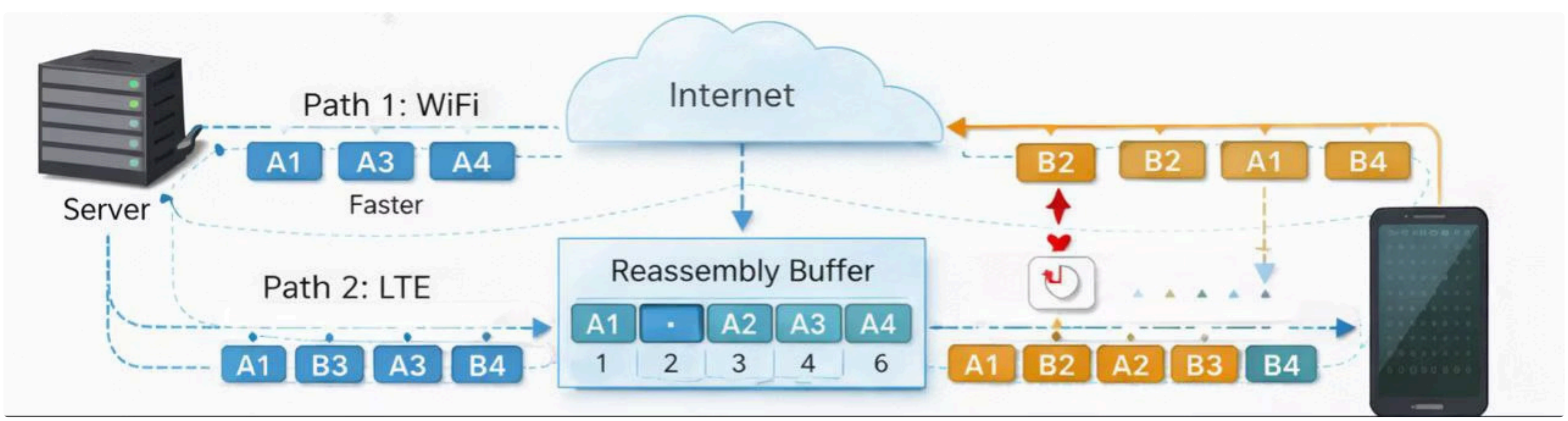
- Each underlying TCP subflow operates with its own independent TCP sequence numbers.
- MPTCP assigns a global DSN to the entire data stream, independent of the subflows.
- The MPTCP receiver uses the DSN to meticulously reorder incoming packets from all subflows.
- This ensures that the application receives data in the correct sequence, even if individual packets arrive out of order or via different paths with varying latencies.



Challenge in MPTCP: Head-of-Line (HoL) Blocking

Even with multiple paths, the fundamental requirement to deliver data in order can lead to delays if one path is significantly slower, causing a backlog at the receiver's reassembly buffer.

- MPTCP utilizes multiple subflows, each potentially having different Round-Trip Times (RTTs) and performance characteristics (e.g., WiFi vs. LTE).
- Consequently, data packets belonging to the same MPTCP connection may arrive at the receiver significantly out of their original sequence.
- The MPTCP layer at the receiver employs a reassembly buffer to hold out-of-order packets and resequence them based on their Data Sequence Numbers (DSNs).
- If a packet sent via a slower or congested path is delayed, the reassembly buffer must wait for its arrival before it can release subsequent, correctly ordered packets to the application.
- This phenomenon is known as Head-of-Line (HoL) blocking, where the delivery of an entire data stream is stalled by a single missing or delayed packet.
- HoL blocking can lead to increased latency, even when the aggregate throughput of the combined paths is high, undermining the benefits of multipath transmission.



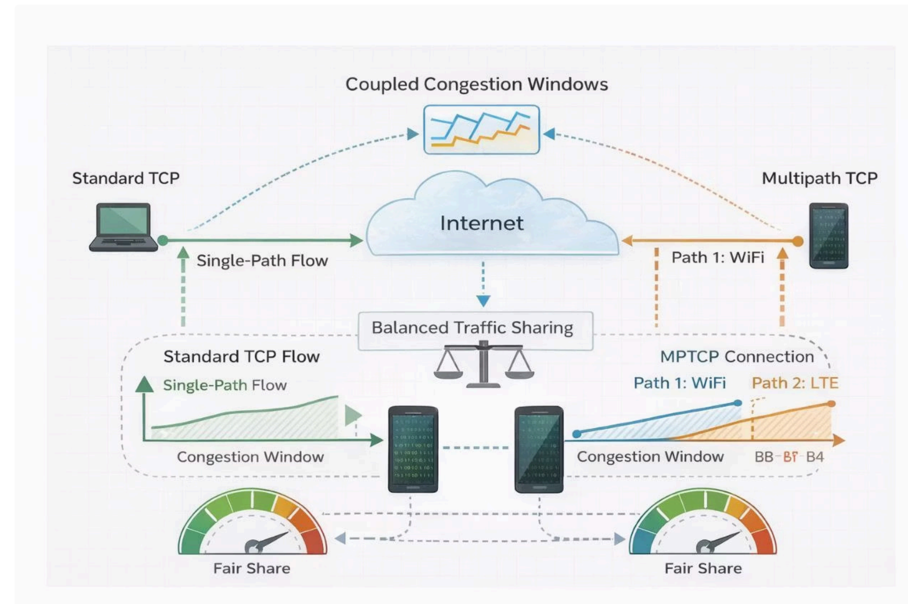
This diagram visually demonstrates how a single delayed packet (Packet 2 via LTE) can hold up the delivery of subsequent packets (3 and 4) at the reassembly buffer, illustrating Head-of-Line blocking in action.

Made with GAMMA

Coupled Congestion Control in Multipath TCP

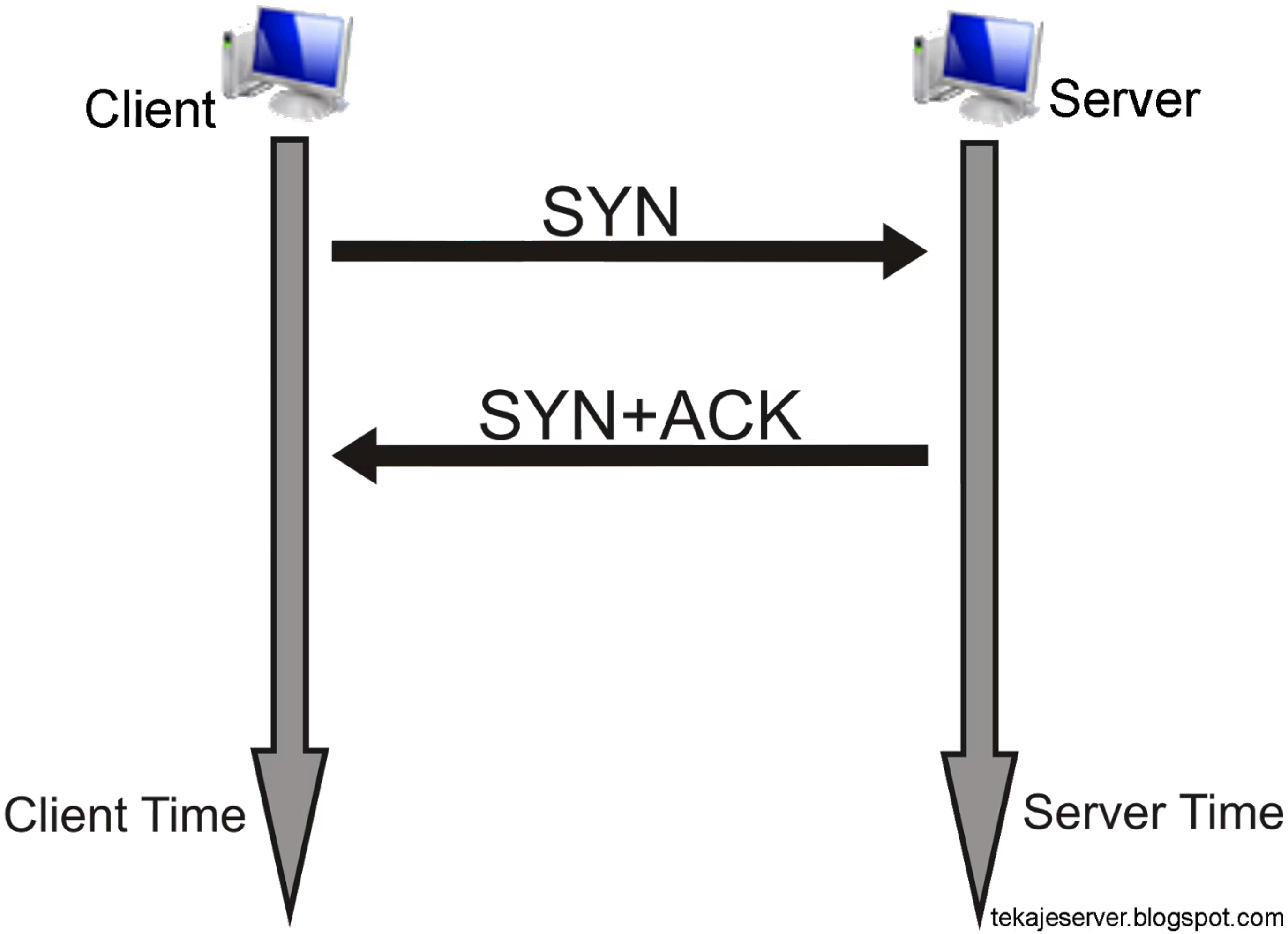
MPTCP's ability to utilize multiple paths simultaneously introduces a critical challenge: ensuring fair bandwidth sharing with traditional single-path TCP connections to maintain network stability.

- **Necessity:** Without coordinated congestion control, MPTCP's multiple subflows could aggressively consume network resources, leading to instability and unfairness.
- **The Risk:** If each MPTCP subflow behaved as an independent TCP connection, it would unfairly compete for bandwidth, potentially starving single-path TCP flows.
- **The Solution:** MPTCP employs **coupled congestion control** algorithms to coordinate the behavior of its subflows.
- **Examples:** Algorithms like LIA (Linked Increase Algorithm) and OLIA (Optimized Linked Increase Algorithm) are designed for this purpose.
- **The Goal:** To ensure that the aggregate MPTCP flow behaves like a single, well-behaved TCP connection, while intelligently balancing load and leveraging diverse paths.



This diagram visualizes how coupled congestion control allows MPTCP to share bandwidth fairly with standard TCP, preventing an unfair advantage despite using multiple paths.

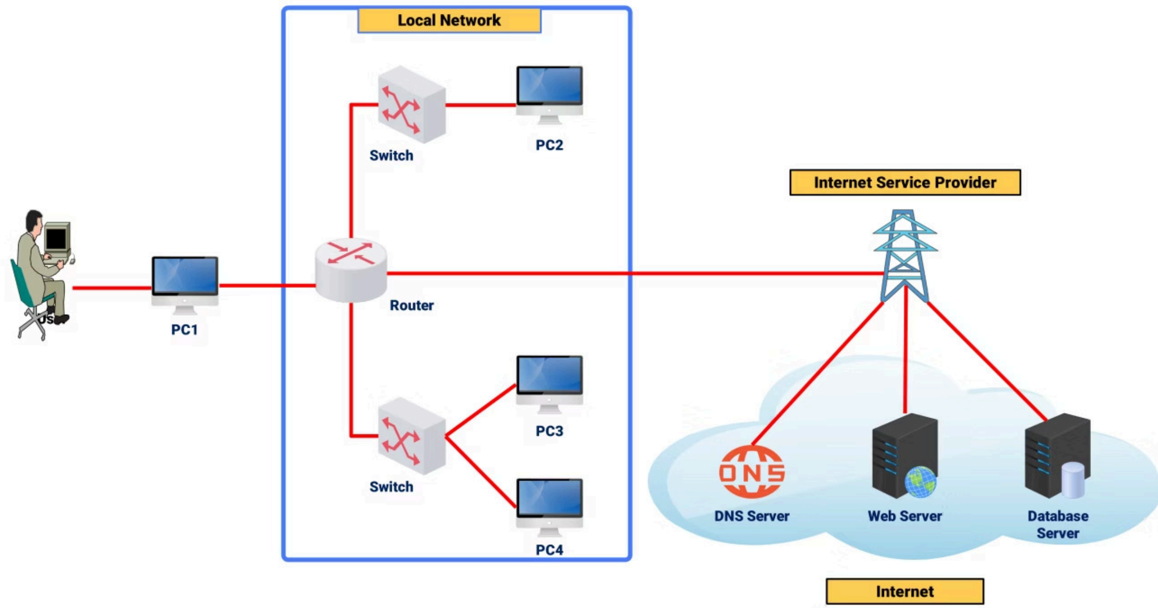
Establishing the Connection: The MP_CAPABLE Handshake



- **Step 1:** The client initiates a connection by sending a SYN packet with the MP_CAPABLE option.
- **Step 2:** If the server supports MPTCP, it responds with a SYN/ACK packet also containing the MP_CAPABLE option.
- **Result:** A standard TCP connection is established, but both endpoints are now aware that additional paths can be added later to enhance connectivity.
- **Fallback:** If the server does not support MPTCP, it simply ignores the MP_CAPABLE option, and the connection proceeds as a standard single-path TCP session.

Seamlessly Adding New Paths: MP_JOIN

- **Scenario:** A device detects a new network interface becoming available (e.g., an LTE connection activates while WiFi is in use).
- **Action:** The MPTCP client initiates a new standard TCP handshake on this newly available interface.
- **The Token:** During this handshake, the client includes the `MP_JOIN` option, carrying a unique token that identifies it as part of an *existing* MPTCP session.
- **Result:** The server associates this new connection with the ongoing MPTCP session, allowing data to flow over both paths concurrently, leveraging the aggregated bandwidth and improving resilience.



SECTION 3

Project Implementation: Application-Layer Emulation

To demonstrate MPTCP's failover capabilities, we implemented a Python-based, user-space simulation with a primary (WiFi) and backup (LTE) path for streaming data and simulating link failures.

The "Scheduler" Algorithm: Failover Logic

Primary Attempt

The system defaults to Path A for initial data transmission, simulating optimal conditions (e.g., WiFi).

Failure Detection

If a connection drop or timeout is detected (simulated after a specific number of packets, e.g., Packet #10), an exception is raised by the network layer.

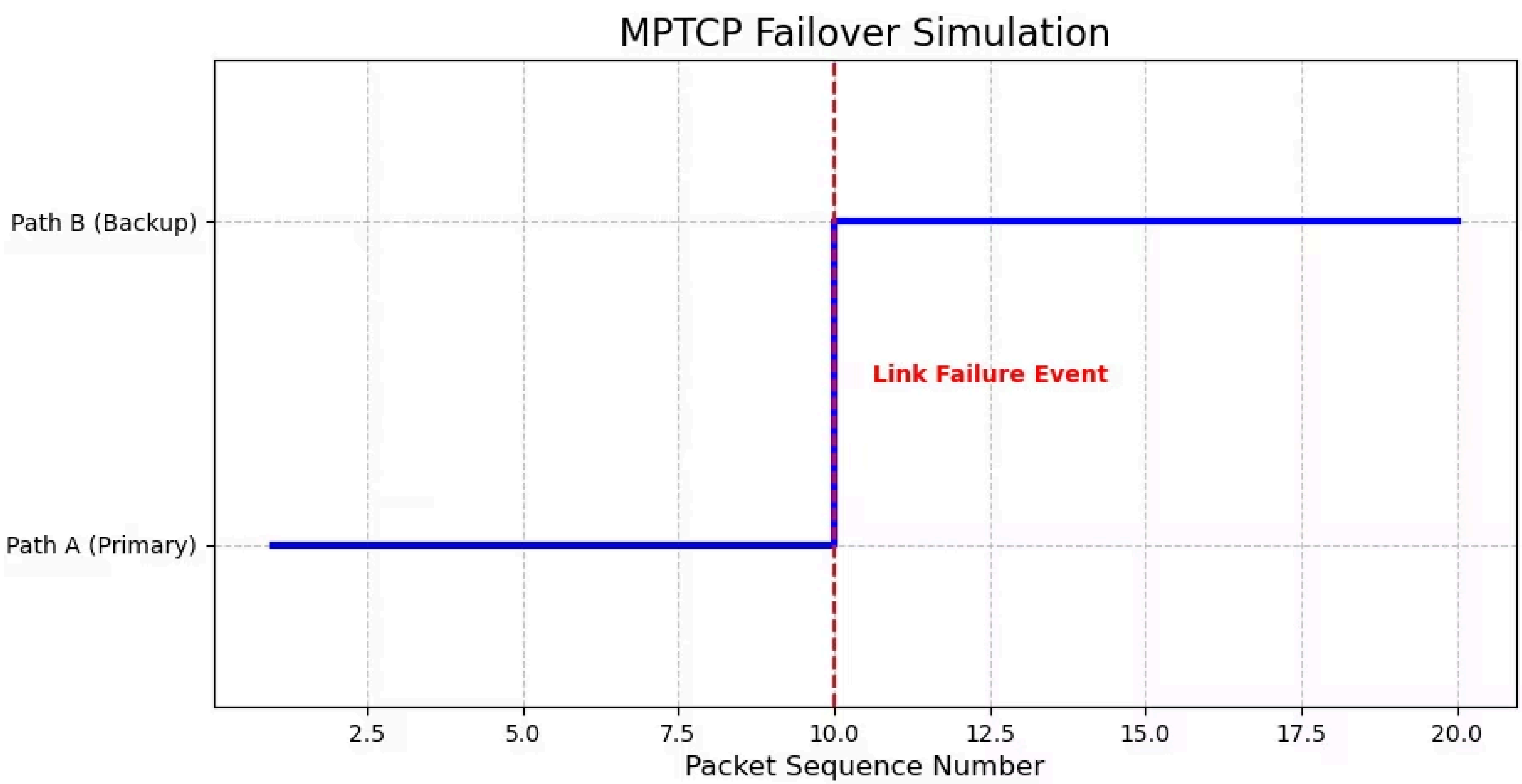
Instant Switch

Our logic immediately catches the exception and reroutes the subsequent data packets to the Backup Path (Path B).

Zero Data Loss

The application continues to stream data without crashing or experiencing noticeable service interruption, demonstrating seamless failover.

Demonstration Results: Seamless Recovery



01

Active Path Indication

The blue line on the graph indicates the currently active data path, with Level 1 representing the Primary (WiFi) and Level 2 representing the Backup (LTE).

02

Simulated Failure Event

Precisely at **Packet 10**, marked by a vertical red line, we simulated a failure of the Primary path.

03

Immediate Path Switching

The graph demonstrates an **immediate and sharp jump** to Level 2 (the backup path) right after the primary path's failure, showing no interruption in data flow.

04

The MPTCP Advantage

In a standard TCP connection, this graph would abruptly end at the red line. MPTCP's intelligent scheduler allows the connection to **seamlessly continue** over the secondary path, ensuring continuous service.

SECTION 4

Who Uses MPTCP Today?

Apple iOS Integration

Siri, Apple Music, and Maps leverage MPTCP to maintain responsiveness even when **WiFi signals are weak**, ensuring commands and streams go through without interruption.

5G Networks (ATSSS)

Access Traffic Steering, Switching, and Splitting (ATSSS) within 5G architectures uses MPTCP to **bond WiFi and cellular connections** for enhanced performance and reliability.

Hybrid Access Broadband

Internet Service Providers (ISPs) deploy MPTCP to combine fixed lines like DSL with **4G/5G mobile connections**, delivering boosted home internet speeds and greater resilience.

CONCLUSION

Summary

Breaks Single-Path Constraints

MPTCP overcomes the inherent limitations of traditional TCP by leveraging multiple network paths simultaneously.

Delivers Core Advantages

Enjoy significantly higher throughput, seamless device mobility, and superior connection reliability.

Validates Failover Logic

Our simulation successfully showcased MPTCP's "**Seamless Failover**" mechanism for uninterrupted data flow.

References & Further Reading

Core Protocol Specifications (IETF RFCs)

- Ford, A., Raiciu, C., Handley, M., Bonaventure, O., & Paasch, C. (2020). *TCP Extensions for Multipath Operation with Multiple Addresses*. IETF RFC 8684.
- Scharf, M., & Ford, A. (2011). *Architectural Guidelines for Multipath TCP Development*. IETF RFC 6182.
- Raiciu, C., Handley, M., & Wischik, D. (2011). *Coupled Congestion Control for Multipath Transport Protocols*. IETF RFC 6356.

Academic Research (Algorithms & Latency)

- Khalili, R., Gast, N., Popovic, M., & Le Boudec, J.-Y. (2012). *MPTCP is not pareto-optimal: performance issues and a possible solution*. Proceedings of ACM CoNEXT '12.
- Ferlin, S., Dreibholz, T., & Alay, O. (2014). *Multi-path transport over heterogeneous wireless networks: Does it really pay off?*. 2014 IEEE Global Communications Conference (GLOBECOM).

Industry Deployments (Apple iOS & 5G)

- Apple Inc. (2023). *Multipath TCP for iOS*. Apple Developer Documentation.
- 3GPP. (2019). *System Architecture for the 5G System (5GS)*. 3rd Generation Partnership Project, Technical Specification TS 23.501, Release 16.