ALMA MATER STUDIORUM – UNIVERSITÀ DI BOLOGNA
CAMPUS DI CESENA

Corso di Laurea in Ingegneria e Scienze Informatiche

Studio dei protocolli MultiPath nell'Internet allo stato dell'arte

Tesi di laurea in
Reti di Telecomunicazioni

Relatore
(**Prof.** | **Dott.**) **Franco Callegati**

Candidato
Nicola Milandri

Sessione di Laurea 20/07/2023
Anno Accademico 2022-2023

Abstract

Nel mondo interconnesso di oggi, una comunicazione affidabile ed efficiente è fondamentale per il trasferimento dei dati e per il miglioramento dell'esperienza dell'utente.
L'affidamento convenzionale alla comunicazione a percorso singolo, tuttavia, spesso deve affrontare problematiche come la congestione dei percorsi, il degradamento del segnake e la vulnerabilità alle interruzioni.
Per superare queste limitazioni e ottenere nuove possibilità, i ricercatori hanno rivolto la loro attenzione alla comunicazione MultiPath, un paradigma che sfrutta la potenza di più percorsi sumultanei per migliorare l'affidabilità e le prestazioni della rete.
In questo documento andremo ad analizzare il protocollo attualmente più utilizzato, il MultiPathTCP (MPTCP), assieme ad altri protocolli più recenti ed andremo a metterli a confronto per verificare la prestanza delle nuove soluzioni rispetto a quella più diffusa.

Alla mia famiglia, ai miei amici

Contents

List of Figures

Chapter 1

Introduzione

La domanda sempre crescente di contenuti multimediali di alta qualità, di elaborazione dei dati in tempo reale e di connettività ininterrotta ha reso necessaria la continua evoluzione delle reti di telecomunicazione.
I sistemi di comununicazione tradizionali si basano principalmente sulla trasmissione a percorso singolo, in cui i pacchetti di dati viaggiano lungo un percorso predeterminato dalla sorgente alla destinazione.
Man mano che le reti diventano più complesse e il volume dei dati aumenta in modo esponenziale, fare affidamento esclusivamente sulla comunicazione a percorso singolo si sta rilevando inadeguato.
La comunicazione MultiPath offre una soluzione convincente per affrontare le problematiche incontrate nella trasmissione singlepath. Consentendo l'utilizzo simultaneo di più percorsi, si mira a migliorare l'affidabilità, il throughput e la resilienza delle connessioni di rete.
Il concetto fondamentale su cui si basa la comunicazione multipath risiede nella distribuzione dei dati su più percorsi invece di affidarsi ad un unico percorso bottleneck, i pacchetti di dati vengono divisi e inviati simultaneamente su percorsi diversi. Dividendo i dati, la comunicazione MultiPath non solo mitiga la congestione, ma riduce anche la latenza e migliora il throughput complessivo.
In questo documento andremo ad analizzare come il protcollo più diffuso ad oggi, ovvero MPTCP, ed altri protocolli più recenti, come MPQUIC, MPMTP-AR e CMT-SCTP siano implementati per portare questi vantaggi alla comunicazione della rete, andando anche a confrontare tra di loro le loro prestazioni per vedere quale protocollo, attraverso la sua struttura riesca ad adempiere al meglio a questo compito.

Chapter 2

MultiPath e Multihoming

Questo capitolo presenta alcune soluzioni sviluppate da IETF in ambito multipath e multihoming

2.1 Soluzioni Multipath

Nel corso del tempo si è dibattuto a lungo su quale approccio fosse più adatto ad utilizzare più percorsi, così che possa sfruttare in modo migliore le risorse disponibili nella rete, in modo da fornire una connettività costante attraverso l'utilizzo di interfacce multiple sui dispositivi mobili. Diversi ricercatori supportano diverse soluzioni, alcuni credono che la scelta migliore, per ottenere dei benefici, sia quella di effettuare l'implementazione a livello di rete, altri hanno utilizzato applicazioni specifiche per implementare le loro soluzioni, le quali diffonodono i blocchi di file in peer diversi, con l'unico scopo di aumentare il througput.
Nelle reti locali è possibile esplorare la diversità dei percorsi a livello di collegamento. Un'implementazione a livello di trasporto invece porterebbe vantaggi grazie alla sua trasparenza a livello di rete, fornendo una relativa indipendenza a livello dell'applicazione, evitando la necessità di modificare i collegamenti ben consolidati e il livello di rete. D'altra parte, dato che MPTCP è un'estensione di TCP, le applicazioni TCP a percorso singolo possono essere utilizzate senza alcuna modifica a livello di applicazione. Successivamente verranno mostrati alcuni approcci alternativi.

2.1.1 Livello di collegamento

Una soluzione per fornire multipath a livello di collegamento consiste nell'utilizzo del protocollo di Link Aggregation Control Protocol (LACP) che utilizza un approccio di bundling a livello di collegamento. Questo approccio avviene tramite

l'aggregazione delle porte dello switch in modo da poter sfruttare più connessioni di rete in parallelo per ottenere un througput maggiore e creare ridondanza in caso di guasto del collegamento.
Una soluzione che sfrutta uno schema multipath a questo livello, è lo Shortest Path Bridging (SPB) che permette l'utilizzo di percorsi multipli con egual costo in ambienti di rete Ethernet mash, questa soluzione supporta una molto più ampia topologia di livello due che può essere utilizzate nei nodi del data center, ma che non può utilizzare interfacce multiple. Un'altra implementazione suggerita a livello di collegamento, che permette di ottenere un miglior throughput nelle Wireless Mesh Networks (WMN) necessita l'implementazione di un multicanale a livello di collegamento, in combinazione con il routing multi-path che si occupa di instradare il traffico in modo efficiente. Il livello di collegamento è il responsabile della pianificazione del canale e dei pacchetti, il primo è utilizzato per controllare da quale canale le informazioni verranno ricevute, mentre il secondo decide quando saranno inviati i pacchetti.
Il multipath-routing ha il compito di selezionare i due percorsi più rapidi verso il gateway, una soluzione a questo livello ha un ottimo potenziale per ottenere buone prestazioni e un più elevato throughput end-to-end in questo tipo di reti, ricorrendo alla scomposizione del traffico in diversi canali, programmandone le diverse tempistiche e utilizzando percorsi differenti. Le soluzioni multipath di livello 2 sono principalemnte limitate alle reti locali.

2.1.2 Livello di Rete

L'implementazione di un protocollo MultiPathTCP a livello di rete sembra scontata.
In questo caso si avrebbe, a livello di trasporto, una singola connessione e i pacchetti verrebbero sparsi su flussi differenti, il load balancing viene eseguito a livello di connessione e non a livello di pacchetto e si ha una sola connessione a livello di trasporto, il throughput sarà così dettato dal collegamento più congestionato o da quello più lento, poichè i controlli della congestione dei percorsi sono aggregati fra loro tramite il protocollo di trasporto. Questa soluzione può diminuire le ritrasmissioni superflue sul livello di trasporto grazie al riordinamento dei pacchetti, portando a una conseguente riduzione del throughput della connessione, queste ritrasmissioni possono verificarsi perchè la maggior parte dei dispositivi di rete sparsi nell'Internet non supportano e non riconoscono il traffico generato.
Per una soluzione multipath a livello di rete si dovrebbero aggiornare tutti i dispositivi di rete attualmente in uso su Internet.

2.1.3 Livello di trasporto

L'implementazione di un protocollo multipath a livello di trasporto può raccogliere le informazioni come capacità, latenza e stato di congestione di ogni percorso utilizzato. Grazie a ciò si può contrastare la congestione della rete, spostando il traffico in modo da evitare l'utilizzo di percorsi congestionati, un'implementazione multipath su questo livello consente di avere trasparenza sia nei livelli superiori che in quelli inferiori, fornendo la possibilità di utilizzare più flussi somiglianti a normali connessioni TCP , offre anche funzioni di gestione del percorso, pianificazioni dei pacchetti, controllo della congestione e persino dell'interfaccia del flusso secondario senza intaccare gli altri livelli. Quindi si può dire che è solamente necessario che gli endpoint supportino il protocollo, senza doversi preoccuapre di dover aggiornare alcun router o componente di livello 3 tra gli endpoint.

2.1.4 Livello di Applicazione

Un esempio di approccio a livello di applicazione sono i protocolli Peer-to-Peer(P2P), soluzione multipath che sfrutta la chunk granularity cercando di aumentare il troughput. Questi cercano di raggiungere il proprio obbiettivo scaricando i diversi blocchi di un file attraverso peer diversi, selezionando i server più veloci per il download, ad esempio, BitTorrent esegue un pooling delle risorse sul livello di lavoro per i trasferimenti molti a uno. Funziona come un controllo di congestione disaccoppiato controllando la gestione su ogni percorso in maniera indipendente, utilizzando percorsi con diversi endpoint è inoltre possibile sviluppare un applicazione che fornisca il multihoming ai dispositivi e ai server che utilizzano protocolli P2P. Il problema di questa soluzione è, che sfruttando la diversità dei percorsi per la disponibilità di più server, si potrebbero portare svantaggi ad altri utenti, fornendo una concorrenza sleale per le risorse disponibili, ma comunque non affrontando la tematica del multipath end to end.

2.1.5 Approcci Multipath proposti da IETF

Nel corso del tempo, il gruppo IETF ha creato un team che si occupa di capire come poter utilizzare percorsi multipli a livello di trasporto. Il più recente e promettente protocollo implementato è il MultiPathTCP(MPTCP).

Approcci multipercorso

Una delle prime soluzioni proposte,a livello di trasporto, che permettesse a molteplici flussi di attraversare diversi percorsi, è lo Stream Control Transmission Protocol (SCTP), risulta affidabile per il trasferimento di messaggi utente tra due endpoint

SCTP. Durante l'associazione di avvio mette a disposizione un elenco di più indirizzi IP in combinazione con una porta SCTP, così facendo ogni endpoint può formare un indirizzo di trasporto compatibile. Questo protocollo è stato sviluppato partendo dai limiti che il TCP ha con le applicazioni odierne per l'invio dei dati in modo affidabile attraverso lo User Datagram Protocol (UDP), inoltre avrebbe dovuto nativamente supportare il multihoming a livello di trasporto. La sua diffusione su Internet però, riscontra delle problematiche legate principalmente al fatto che siano presenti dispositivi incompatibili e che quindi non siano in grado di riconoscerlo, poichè non offre un ordinamento rigido nei diversi flussi non permettendo di utilizzare la larghezza di banda su più percorsi. Una soluzione importante per l'evoluzione di questo protocollo è il Multiple paths TCP (mTCP), protocollo end-to-end che permette l'aggregazione della larghezze di banda disponibile tra percorsi parelleli e ridondanti, molto comuni tra una coppia di host, in modo da ottenere performance più elevate e più robustezza ai fallimenti dei percorsi. Questo approccio rimuove i pacchetti di un flusso da più percorsi e fornisce un throughput end-to-end maggiore, supporta inoltre un meccanismo condiviso di rilevamento della congestione per evitare percorsi con una congestione condivisa. Un'altro protocollo è il pTCP, un protocollo end-to-end sul livello di trasporto che permette l'aggregazione di banda su agenti multihoming: riesce a separare il recupero delle perdite dal controllo di congestione e dallo stripping dei dati tramite le connessioni disponibili. L'approccio che più si avvicina al MPTCP è il Concurrent Multi-Path Stream Control Transmission Protocol (CMP-SCTP), estensione di SCTP traente vantaggi dal multihoming al fine di aumentare il throughput tra due endpoint multihomed. Sono aggiunti: tolleranza agli errori, grazie all'invio simultaneo dei dati su tutti i percorsi, l'utilizzo di un singolo spazio di sequenza tramite un file di associazione a più percorsi, presenta un unico controllo della congestione e uno di ritrasmissione su ciascun percorso. I percorsi di ritorno sono entrambi percorsi disponibili per la trasmissione di informazioni di riconoscimento, portando ad avere una distribuzione efficace del carico sul livello di trasporto.

Approcci multihoming

MPTCP può trarre vantaggio dall'utilizzo simultaneo di diverse interfacce di un dispositivo, migliorando l'affidabilità della connettività. Una soluzione mirata a migliorare la mobilità è il Mobility Support in IPv6 (MIPv6), protocollo che permette ai nodi di essere raggiunti anche durante il loro spostamento, a patto che siano all'interno di una rete Internet IPv6, ciò è dato dal fatto che il nodo mobilie presenta tre indirizzi IPv6, il primo è l'indirizzo di casa del nodo, il secondo è il suo indirizzo link-local mentre il terzo è il care-of-address del nodo mobile. I tre indirizzi permettono agli agenti, domestico e interno, di formare un tunnel mentre il nodo è assente, consentendo all'agente domestico di poter trovare il nodo

indipendentemente dalla sua posizione. Una soluzione precedente a questa per la mobilità era il Mobility Support IPv4 (MIPv4), base di sviluppo per MIPv6, ma con la differenza che presupponeva che ogni nodo della rete avesse un indirizzo univoco, cosa diventata obsoleta con l'aumento dei personal computer e l'emersione di sempre più oggetti di uso quotidiano connessi a Internet (cellulari, tablet, orologi, televisori etc.). Un'altra soluzione per la mobilità è il protocollo Shim6, protocollo implementato sul terzo strato, il che significia che questa soluzione IPv6 permette al multipath di essere implementato a livello di pacchetto invece che routing a livello di connessione. Il load balancing, effettuato a livello di connessione, porta il throughput ad essere influenzato dal collegamento più lento o da quello più congestionato.

Approcci multipath alternativi

Qui di seguito vengono presentati alcuni progetti in fase di sviluppo, ma che forniscono un punto di vista diverso sull'argomento. Beijum ha sviluppato una prospettiva applicativa di un MPTCP implementata esclusivamente lato host mittente, non richiedendo modifiche al ricevente. Questo permette al protocollo di utilizzare la capacità disponibile sui diversi percorsi multipli o di trovare dinamicamente quello più efficiente, in modo da ottenere un throughput maggiore.

Chapter 3

MultiPathTCP

MultipathTCP(MPTCP) aggiunge la capacità di utilizzare più percorsi a una regolare sessione TCP. Questo viene fatto perchè porta ad un aumento del throughput, delle risorse complessve utilizzate e della resilienza al fallimento della rete, MPTCP offre lo stesso, affidabile e ordinato trasporto byte-stream di TCP ed è progettato per essere retrocompatibile con entrambe le applicazioni e con il livello di rete, di seguito verrà presentato il funzionamento del protocollo MPTCP.
Qualunque operazione MultipathTCP viene segnalata tramite campi d'intestazione opzionali TCP, IANA ha deciso di assegnare,per MPTCP, un singolo numero di opzione ("Kind") , portando i singoli messaggi ad essere determinati da un certo "sottotipo". Come per ogni altra opzione TCP, venogno utilizzati i byte per specificare la lunghezza del campo, e fra questi byte sono inclusi i 2 di Kind e Lenght, andando così a definire i messaggi come presentato nell'immagine sottostante.

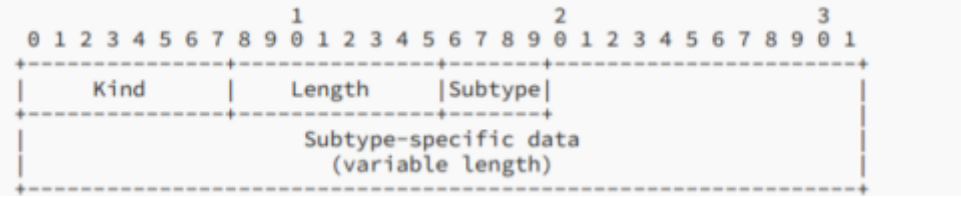

Figure 3.1: Opzioni MPTCP

All'avvio del flusso secondario sono associate le opzioni MPTCP, usate nei pacchetti dove è stato impostato il flag SYN, oltre a ciò esiste anche un'altra opzione che permette di segnalare i metadati, in modo da garantire ai vari segmenti di poter essere ricombinati, al fine di venire consegnati all'applicazione. Le restanti opzioni, tuttavia, non devono essere su un pacchetto specifico, come quella che segnala indirizzi aggiuntivi, un'implementazione potrebbe preferire l'invio delle opzioni non

appena pronte, ma la combinazione delle opzioni desiderate non sempre risulta possibile su un singolo pacchetto, quindi un'implementazione può decidere di inviare ACK dupilicati contenenti altre informazioni di segnalazione, scelta che porta ad un cambiamento semantico di un ACK duplicato, solitamente inviato solo come segnale di perdita di un segmento. Pertanto, una soluzione MPTCP che riceve un ACK duplicato contenente un'opzione MPTCP non deve considerarlo come un segnale di congestione, inoltre, un'implementazione MPTCP non dovrebbe mai inviare più di due ACK duplicati di seguito per l'invio di opzioni, o per permettere alle middlebox di non interpretarli come segnali di congestione. I controlli di validità TCP (garantire il numero di sequenza e il numero di riconoscimento) devono essere intrapresi ancor prima di elaborare un qualsiasi segnale MPTCP.

3.0.1 Organizzazzione MPTCP a livello di trasporto

Di seguito verrà presentata la struttura di MPTCP sul livello di trasporto. Il livello di trasporto viene diviso in due sottostrati dal protocollo, come si può notare nell'immagine 3.2, il substrato superiore si occupa della raccolta delle informazioni utili a tenere sotto controllo la connessione e opera end-to-end, mentre quello inferiore si occupa dei subflow, in modo da renderli visibili come singoli flussi TCP permettendo alle componenti TCP di poter lavorare segmento per segmento. Per gestire i molteplici subflow TCP sottostanti, l'estensione MPTCP deve implementare delle funzioni di gestione dei path, di schedulazione dei pacchetti, del controllo della congestione e di interfaccia del subflow.

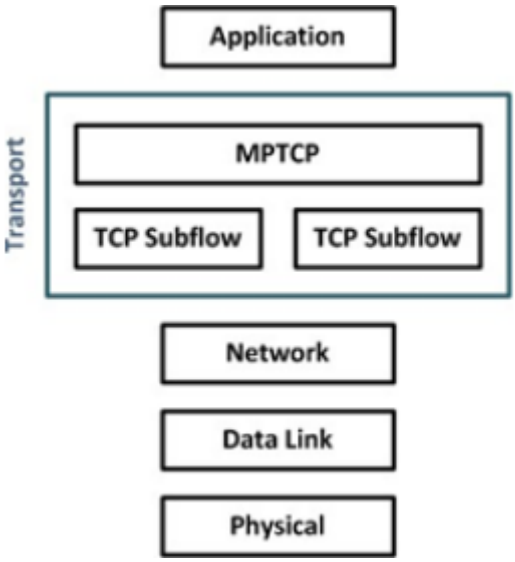

Figure 3.2: Livello di trasporto MPTCP

La gestione dei path deve rilevare e utilizzare i percorsi disponibili fra gli host, deve anche prendersi a carico la segnalazione degli indirizzi alternativi degli host e deve impostare nuovi subflow collegati ad una connessione MPTCP già presente.
Con la schedulazione dei pacchetti,invece, si va a suddividere il flusso di byte in entrata dall'applicazione in segmenti, in modo che possano essere trasmessi attraverso un unico subflow, anche qui viene eseguito il riordino dei pacchetti a livello di connessione e viene fatto ogni qualvolta che i subflow TCP ricevono i vari pacchetti.
Il design MPTCP, per permettere il corretto ordinamento dei segmenti, sfrutta una mappatura di sequenza dei dati, attraverso la quale vengono associati i segmenti ad una numerazione sequenziale nel livello di connessione, le informazioni ottenute dalla componente che si occupa di gestire i path influisce sullo scheduler, permettendo cosi di avere a disposizione le informazioni corrette dei subflow.
Il meccanismo di controllo della congestione invece si occupa del coordinamento dei controlli di congestione tramite i subflow, è il responsabile della scelta dei segmenti da inviare, attraverso quali subflow devono essere inviati e a quale velocità, inoltre, prende parte anche allo scheduling dei pacchetti. L'interfaccia dei subflow, invece, è responsabile per la gestione della trasmissione sul percorso prestabilito dei segmenti che sono stati ricevuti dalla componente dello scheduler dei pacchetti, successivamente alla ricezione di un pacchetto, il subflow trasmette i dati al packet scheduling per permetterne il riassemblaggio a livello di connessione.
Dato che MPTCP sfrutta TCP per la compatibilità di rete, riesce ad ottenere una consegna affidabile e ordinata, il TCP aggiunge ai segmenti la propria sequenza numerica in modo da poter rilevare e ritrasmettere i pacchetti persi nel livello del subflow.
L'Internet Assigned Numbers Authority ha creato un sotto-registro utilizzato da MPTCP nel campo delle opzioni TCP.
Nell'immagine 3.3 sottostante vengono presentate i diversi sottotipi di opzioni del protocollo MPTCP

Valore	Simbolo	Designazione
0x0	MP_CAPABLE	Multipath Capable
0x1	MP_JOIN	Join Connection
0x2	DSS	Data Sequence Signal
0x3	ADD_ADDR	Add address
0x4	REMOVE_ADDR	Remove Address
0x5	MP_PRIO	Change Subflow Priority
0x6	MP_FAIL	Fallback
0x7	MP_FASTCLOSE	Fast Close
0x8-0xe	Unassigned	Unassigned
0xf	Reserved for Private Use	Reserved for Private Use

Figure 3.3: Sottotipi opzioni MPTCP

3.0.2 Gestione della connessione

A questo punto descriviamo come il protocollo MPTCP gestisce la connessione, dal suo avvio alla sua conclusione.
Per l'avvio della connessione il socket aprirà un normale socket TCP, che porterà all'avvio del subflow TCP iniziale.
MPTCP necessita di trasparenza a livello di rete e di applicazione, fino a qui la connessione risulta la stessa di una normale connessione TCP, successivamente, se entrambi gli endpoint supportano il MultiPath TCP, la sessione può cominciare su entrambi gli host. Per stabilire un nuovo subflow da Host A a Host B, il primo riceve dal DNS le informazioni su cosa può raggiungere il secondo all'inidirizzo 1, in seguito viene utilizzato un Three-way handshake, simile al normale TCP, in modo da mantenere la trasparenza con la rete.
L'opzione MP_CAPABLE, presentata nell'immagine 3.4, inserita nell'handshake, ha il compito di permettere agli host di informarsi vicendevolmente e di essere abilitati per il MPTCP. Il primo handshake, atto a iniziare la sessione MPTCP dell'host A, ha inizio con l'invio di un SYN con opzione MP_CAPABLE, l'host B risponde con un SYN+ACK, sempre con l'aggiunta dell'opzione MP_CAPABLE, annunciando il token B al peer, questo token è un identificatore locale della connessione su un host, adesso l'host A risponde con un ACK, sempre contenente l'opzione MP_CAPABLE, che annuncia il suo token A, infine, viene inviato un ACK finale per garantire la ricezione dei dati di sicurezza MPTCP finali.

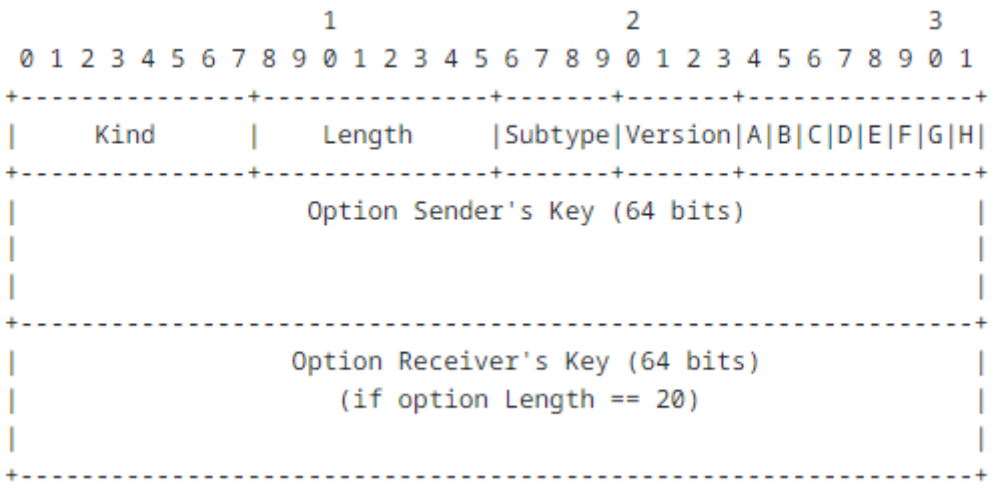

Figure 3.4: MP_CAPABLE

La sessione MPTCP è ora completa.

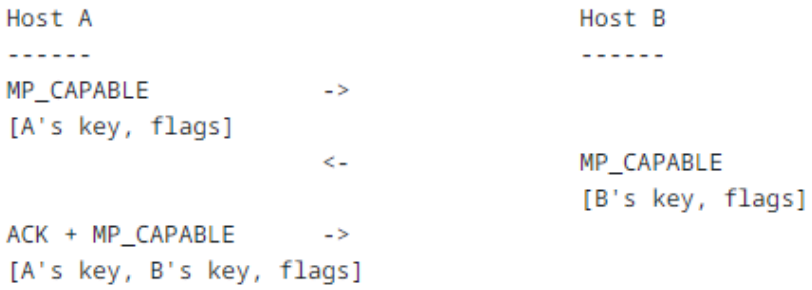

Figure 3.5: Avvio di una comunicazione

Di seguito verrà esposto il metodo di aggiunta di nuovi subflow sulla sessione MPTCP(immagine 3.6).
In qualsiasi momento, entrambi gli host possono portare a stabilire nuovi subflow, se, per esempio, l'host A vuole creare un subflow tra il suo indirizzo 1 e l'indirizzo 2 dell'host B, si ha un nuovo three-way handshake, con un'opzione diversa, in quanto i nuovi subflow creati dovranno essere collegati alla sessione MPTCP già esistente. L'opzione che verrà assegnata per la creazione dei nuovi subflow sarà l'opzione MP_JOIN, l'host A può stabilire un nuovo subflow sfruttando gli indirizzi <A.2,B.1>, Host A indirizzo 2 con Host B indirizzo 1, l'host A allega il token B alla MP_JOIN, l'host B risponde con SYN/ACK, assieme all'opzione MP_JOIN ma senza token, in quanto l'host A ha già uno stato per quel subflow. Successivamente

l'host A invia un ACK con MP_JOIN nel contesto dell'host B, infine viene inviato l'ACK finale, così il subflow è pronto per essere utilizzato.

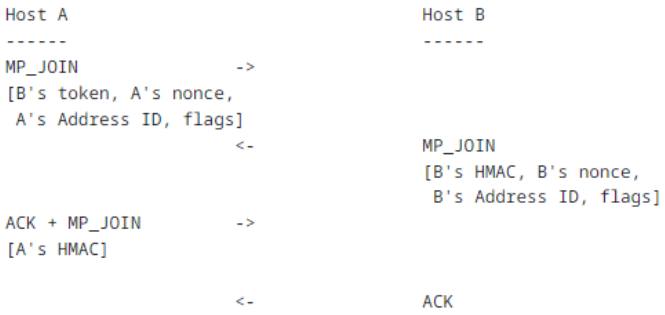

Figure 3.6: Associazioni di un nuovo subflow

Un'opzione MP_JOIN è presente sia in SYN che in SYN/ACK, ma anche in ACK, anche se in formati differenti, come possiamo vedere nell'immagine(3.6), la prima opzione legata al pacchetto SYN, il mittente invia un token, un numero casuale e un indirizzo ID

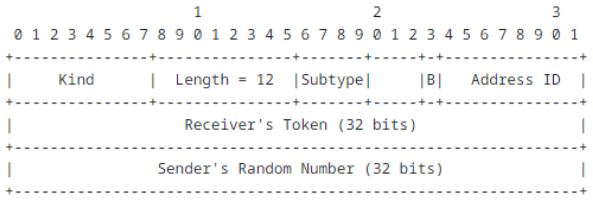

Figure 3.7: MP_JOIN

Nel caso in cui un nuovo indirizzo diventasse disponibile su uno degli host, quel nuovo indirizzo deve essere annunciato all'altro host prima di poter essere utilizzato per connettere un nuovo subflow alla sessione MPTCP.
In primo luogo l'host deve avvisare l'altro tramite l'invio di un opzione ADD_ADDRESS del MPTCP, contenente il nuovo indirizzo, prima di avviare un nuovo three-way handshake, nel caso in cui un subflow smetta di rispondere esiste l'opzione REMOVE_ADDRESS per rimuovere quell'indirizzo dalla connessione.

Scambio di dati

Per permettere ai subflow di consegnare in maniera affidabile e ordinata i dati, MPTCP utilizza un Data Sequence Number (DSN), che enumera tutti i dati inviati tramite la connessione MPTCP e ogni subflow possiede il proprio spazio

numerico a 32 bit come in un normale TCP. MPTCP specifica la mappatura dallo spazio di sequenza del subflow fino allo spazio di sequenza dei dati tramite l'opzione Data Sequence Signal (DSS), il DSS contiene il riconoscimento e si occupa delle informazioni che permettono la mappatura della sequenza di dati, inoltre, in caso di errore può ritrasmettere i dati su diversi subflow. I dati si trovano sparsi sui subflow sotto forma di segmenti e devono essere riconosciuti una volta ricevuti, per prima cosa i dati vengono riconosciuti su tutti i subflow, in quanto si comportano come normali flussi TCP, quindi i dati vengono riconosciuti anche dalla numerazione della sequenza di dati tramite l'utilizzo di un riconoscimento dati cumulativo inviato su uno dei subflow.

Termine della connessione

In una normale connessione TCP, il FIN vine annunciato una volta che un'applicazione effettua la chiamata close() su un socket, indicando di aver terminato i dati da inviare, in MPTCP un FIN influisce solo sul subflow su cui viene inviato.
Per chiudere definitivamente la connessione in MPTCP esiste un meccanismo equivalente al normale TCP FIN, chimato DATA FIN, quando questo comando viene riconosciuto a livello di connessione tramite un DATA ACK, quest'ultimo viene inviato solo successivamente alla trasmissione e alla ricezione, con successo, di tutti i dati sulla connessione MPTCP. Quando è inviato il primo DATA FIN, viene innescata la restituzione del DATA ACK e del DATA FIN dell'altro host, quest'ultimo deve essere inviato solo su un subflow e dal momento in cui viene riconosciuto, tutti i restanti subflow dovrebbero essere chiusi tramite lo scambio di FIN standard, questi scambi consentono alle middlebox di ripulire il loro stato. La connessione MPTCP viene considerata chiusa una volta che i DATA FIN inviati da entrambi gli host sono stati riconosciuti dai DATA ACK. Esiste anche un ulteriore modo per terminare la connessione, e si chiama "Fast Close", analogo alla chiusura di una normale connessione TCP a percorso singolo con un segnale RST, MP_FASTCLOSE ha il compito di indicare al peer che la connessione verrà chiusa bruscamente e che non sarà più accettato alcun tipo di dato. Questo segnale può essere inviato su un ACK (per garantirne l'affidabilità) o su un RST.

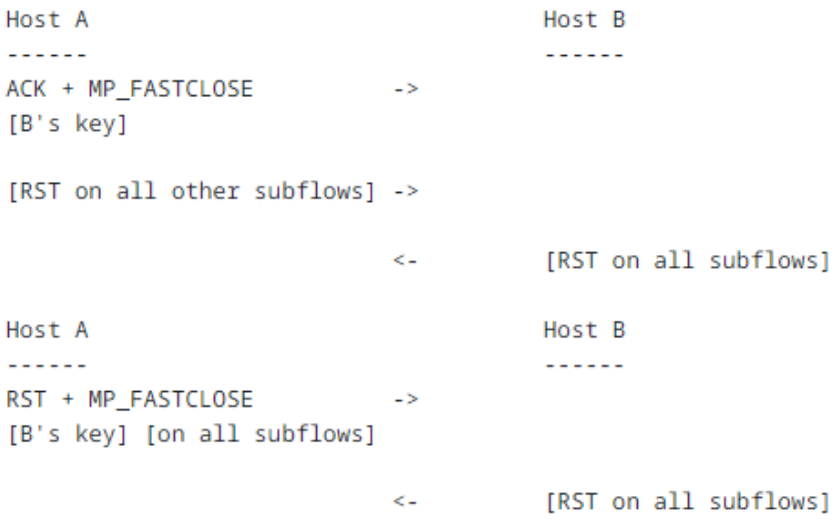

Figure 3.8: Fast Close

3.0.3 Subflow policy

All'interno di un'implementazione locale MPTCP, un host è libero di utilizzare come policy locale quella che più lo aggrada su come condividere il traffico da inviare sui diversi percorsi disponibili.
In un caso d'uso tipico, dove l'obbiettivo principale è quello di massimizzare il throughput, saranno utilizzati contemporaneamente tutti i percorsi che risultano disponibili per il trasferimento dei dati grazie all'utilizzo del controllo di congestione accoppiato.
Un'altro possibile approccio è quello del "tutto o niente", scenario in cui si ha un percorso secondario pronto a subentrare nel caso in cui il primo abbia un guasto. Un'altra alternativa, invece, potrebbe includere la completa saturazione di un percorso prima di passare ad uno aggiuntivo ("overflow").
La scelta di utilizzare un'implementazione piuttosto che un'altra si basa principalmente sul costo monetario dei collegamenti, ma potrebbe anche incentrarsi su alcune proprietà come il delay o il jitter dei collegamenti, dove la stabilità (di delay o di larghezza di banda) è più importante della portata, tali requisiti sono discussi più in dettaglio nel RFC6897. L'essere in grado di effettuare scelte efficaci nel mittente richiede piena conoscenza del "costo" del percorso. Sarebbe auspicabile che un ricevitore fosse in grado di segnalare le proprie preferenze per quanto riguarda i percorsi, in quanto sarà spesso la parte in cui si presenta il multihoming, e potrebbe risentire della poca larghezza di banda in entrata. L'opzione MP_JOIN contiene il bit 'B' che permette ad un host di comunicare al suo peer

che il percorso deve essere adibito a percorso di backup e utilizzato solo in caso di guasto di altri subflow funzionanti (un subflow dove il ricevitore presenta B=1 non dovrebbe eessere impiegato a meno che non si presenti l'assenza di subflow con B=0). Nel caso in cui l'insieme dei percorsi disponibili cambi, un host potrebbe voler segnalare al peer il cambio di priorità dei subflow (es. il precedente subflow di backup dovrebbe poi avere la priorità su tutti gli altri subflow rimanenti). Per questo l'opzione MP_PRIO può essere utilizzata per attivare il flag 'B' del subflow sul quale è inviato.

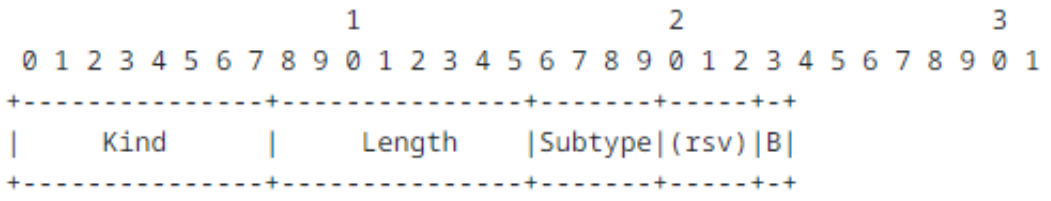

Figure 3.9: Cambio priorità di un subflow MP_PRIO

Il flag di backup è una richiesta proveniente da un ricevente ad un solo mittente di dati, che dovrebbe accettare queste richieste. Un host non può presumere che lo farà il destinatario, tuttavia, le policy locali hanno la capacità di sovrascrivere le richieste dell'opzione MP_PRIOR, il segnale è monodirezionale, quindi il ricevitore di questa opzione potrebbe decidere di continuare ad utilizzare il subflow per l'invio dei dati anche se ha segnalato B=1 all'altro host. Questa opzione può essere applicata anche ad altri subflow diversi da quello attivo, basta impostare il campo facoltativo dell'indirizzo ID, questa applica l'impostazione data di B a tutti i subflow nella connessione che utilizzano l'indirizzo identificato dall'ID specificato. La presenza di questo campo è determinata dall'opzione lenght, se Lenght==4 allora è presente, se invece, Lenght==3 allora si applica solo al subflow corrente, in questo caso l'host può segnalare al suo peer che un indirizzo è temporaneamente indisponibile e il peer dovrebbe quindi cadere per eseguire il backup dello stato su tutti i subflow che stanno utilizzando quell'indirizzo ID.

3.0.4 Path Management

La rete non espone la diversità del percorso tra le coppie degli indirizzi IP.
Per ottenere tale diversità di percorso MPTCP utilizza più indirizzi in uno o in entrambi gli host, per trovare percorsi diversi attraverso la rete e, ci si aspetterebbe che questi percorsi siano sufficientemente disgiunti per consentire al multipath di ottenere una migliore produttività e robustezza. L'utilizzo di più indirizzi IP è un meccanismo che non richiede funzionalità aggiuntive nella rete, saranno quindi

molteplici coppie di indirizzi diverse ad essere utilizzate, nella maggior parte dei casi, come selettori di percorso. Ogni percorso sarà indentificato da cinque-tuple standard (es. indirizzo di origine, indirizzo di destinazione, porta di originie, porta di destinazione, protocollo), per consentire all'estensione di MPTCP di sfruttare porte e indirizzi come selettori di percorso, questo permetterà agli host di utilizzare porte basate sul load balancing.
Gli ISP (Internet Service Provider) spesso intraprendono l'ingegneria del traffico per ottimizzare l'utilizzo delle risorse all'interno delle propie reti e si dovrebbe prestare attenzione al fatto che MPTCP utilizza percorsi sostanzialmente simili che non devono interferire. Per una maggiore possibilità di creare ulteriori subflow entrambi gli host dovrebbero essere in grado di aggiungere nuovi subflow a una connessione MPTCP. Quest'ultima deve essere in grado di gestire i percorsi che appaiono e scompaiono durante la durata di una connessione.
La gestione del percorso è una funzione separata dal pacchetto di schedulazione, dall'interfaccia del subflow e dalle funzioni di controllo della congestione.

3.0.5 Controllo di congestione

La possibilità di utilizzare percorsi multipli porta delle preoccupazioni sul controllo della congestione dei dati inviati su questi percorsi. Una grande quantità di risorse è stata adibita alla ricerca sugli algoritmi di controllo della congestione e sugli schemi di routing che permettono una divisione ottimale dei pacchetti tra i diversi percorsi disponibili, di seguito verranno presentati alcuni algoritmi di controllo di congestione.

Algoritmo Fully Coupled

Questo algoritmo considera il caso in cui tutti i flussi hanno un Round Trip Time (RTT) simile. Il problema di questo algoritmo è che porta "flappines" che si presenta una volta che il traffico di una connessione multi-path tende a concentrarsi prima su un percorso poi su un altro.

Algoritmo Linked Increases

E' un algoritmo che ha come scopo quello di ridurre la flappines e cambiare il pooling delle risorse. Per compensare le diverse connessioni RTT, è stato studiato un algoritmo particolare: RTT ComPensator Algorithm che, sacrificando un po' di bilanciamento della congestione, mira a migliorare la stabilità

Algoritmo Dynamic Window Coupling

L'algoritmo DWC tende a differenziare i flussi a seconda se condividono o meno un collegamento a collo di bottiglia e, nel caso in cui lo facciano, li accoppia. L'algoritmo è in grado di individuare i collegamenti a collo di bottiglia e di migliorarne il throughput al costo di essere meno user-friendly per utenti con connessioni TCP singole.

Chapter 4

Protocolli MultiPath alternativi

Nel seguente capitolo andremo a presentare dei protocolli alternativi rispetto al più utilizzato MPTCP.

4.1 MultiPath QUIC

MultiPath QUIC (MPQUIC) è un'estensione del protocollo di trasporto QUIC che consente alle connessioni QUIC di aggregare più percorsi di rete in modo simile a come MPTCP abilita il multipath per TCP.
QUIC è già in grado di utilizzare più percorsi per una singola connessione tramite il meccanismo di migrazione della connessione. QUIC non è però in grado di utilizzare efficacemente più di un percorso contemporaneamente su una singola connessione. MPQUIC mira ad andare oltre il meccanismo sopracitato per fornire funzinalità CMT.
MPQUIC supporta nativamente meccanismi di controllo della congestione ispirati a MPTCP, inoltre è meno suscettibile alla manomissione della middlebox rispetto a MPTCP, che porta ad ottenere meno compromessi e un design più semplice.
Dato che MPQUIC supporta il multiplexing dei flussi, è più robusto contro il blocco head-of-line (ogni browser/client può attivare un numero limitato di connessioni a un server, e prima di poter attivare una nuova richiesta deve attendere che si liberi una connessione) e può utilizzare tecniche di pianificazione dei pacchetti basate sul flusso.

4.1.1 Path Initiation

In accordo con il protocollo QUIC, ogni endpoint utilizza un frame new_connection_id per emettere ID di connessione utilizzabili per raggiungerlo.

Prima che un endpoint aggiunga un nuovo percorso avviandone la convalida, deve prima verificare se almeno un ID di connessione inutilizzato è disponibile per ciascun lato. Se il parametro di trasporto active_connection_id_limit viene negoziato come N, significa che il server ha fornito N ID di connessione e il client sta già utilizzando attivamente N percorsi, nel caso in cui il client voglia iniziare un nuovo percorso, deve prima ritirare uno dei percorsi stabiliti. Quando l'opzione multipath viene negoziata, i client che desiderano utilizzare un percorso aggiuntivo devono prima avviare la procedura di convalida dell'indirizzo con i frame PATH_CHALLENGE e PATH_RESPONSE.

Dopo aver ricevuto i pacchetti dal client su un nuovo percoroso, se il server decide di utilizzarlo, il server deve eseguire la sua convalida a meno che non abbia precedentemente convalidato tale indirizzo. Se la convalida ottiene esito positivo, il client può inviare pacchetti 1-RTT non sondanti sui nuovi percorsi.
In contrasto con le specifiche di QUIC, il server non deve presumere che la ricezione dei pacchetti non sondanti su un percoroso indichi un tentativo di migrazione verso di esso, invece, i server dovrebbero considerare come disponibili per la trasmissione nuovi percorsi sui quali sono stati ricevuti pacchetti non sondanti.

Gestione dei percorsi

Un percorso di MPQUIC è rappresentato da una quadrupla composta da

- **Indirizzi IP sorgente**
- **Indirizzi IP di destinazione**
- **Porta sorgente**
- **Porta di destinazione**

Analogamente a QUIC, una connessione MPQUIC viene stabilita su un flusso dedicato su un percorso iniziale, dove viene eseguito l'handshake TLS, percorso che però successivamente viene evitato di utilizzare per il trasferimento dei dati. Invece, MPQUIC può stabilire un "duplicato" del percorso che utilizza gli stessi indirizzi IP ma numeri di porta diversi.
Durante l'handshake, entrambi gli endpoint annunciano quanti percorsi possono essere utilizzati per la connessione, il valore pubblicizzato più basso determinerà quanti percorsi aggiuntivi possono essere utilizzati per la connessione.
MPQUIC può sfruttare l'handshake 0-RTT di QUIC per stabilire connessioni a bassa latenza su nuovi percorsi, è in grado di consentire ad entrambi gli estremi di aprire nuove strade ,tuttavia, nell'attuale implementazione MPQUIC i percorsi avviati dal server non vengono utilizzati.

Ogni percorso è identificato utilizzando un ID percorso univoco e i pacchetti che vengono trasmessi dopo che è stata stabilita una connessione devono includere l'ID del percorso nell'intestazione. La quadrupla associata ad un percorso può cambiare nel tempo. A differenza di QUIC, che migra l'intera connessione su un altro percorso, MPQUIC può utilizzare questo meccanismo per migrare un percorso su un altro, questa capacità è chiamata percorso di migrazione. Per migrare un percorso, un endpoint può semplicemente inviare un pacchetto che utilizza lo stesso ID di un percorso su un altro percorso, così la quadrupla gestita dal ricevitore verrà quindi aggiornata in base al pacchetto inviato più recente. I percorsi dei frame contengono informazioni relative ai percorsi attivi del mittente, tali informazioni contengono l'ID del percorso, l'indirizzo locale e l'RTT percepito dai mittenti del percorso.
Il frame dei percorsi può essere utilizzato per rilevare potenziali problemi di connettività e fornire a entrambi gli endpoint una visione globale di ciascun percorso. Inoltre, un endpoint potrebbe usare il frame ADD ADDRESS per segnalare la disponibilità di un nuovo indirizzo mentre REMOVE ADDRESS per segnalare che l'indirizzo precedentemente disponibile, non lo è più.
Un'idea centrale in QUIC è che i frame dovrebbero essere indipendenti dai pacchetti che trasportano. Per rimanere conforme a tale idea, MPQUIC considera anche i frame indipendenti dal percorso su cui sono inviati, in questo modo il protocollo può ritrasmettere i frame in percorsi diversi e ogni percorso mantiene il proprio set di numeri di sequenza dei pacchetti. Il frame ACK è stato modificato per consentire riconoscimenti in base al percorso, includendo il corrispondente PATH ID, le modifiche apportate all'ACK implicano che i riconoscimenti per i pacchetti di un percorso possano essere inviati su un percorso diverso, nonstante questa opportunità fornisca flessibilità di programmazione, può avere un impatto sull'RTT percepito di un percorso.
Il riconoscimento dei frame è tipicamente utilizzato per la stima dell'RTT, tuttavia, dato che la ricezione di ACK non è più garantita, i riconoscimenti non sono più adatti per stimare l'RTT.

4.1.2 Chiusura percorso

Ogni endpoint gestisce il set di percorsi disponibili per la trasmissione. In qualsiasi momento della connessione, gli endpoint possono decidere di abbandonare uno di questi percorsi, seguendo ad esempio dei cambiamenti nella connettività locale o cambiamenti nelle preferenze locali. Dopo che un endpoint abbandona un percorso, il peer non riceverà più pacchetti non sondanti su quel percorso.
Un endpoint che vuole chiudere un percorso dovrebbe usare la richiesta esplicita a terminarlo inviando il frame PATH_ABANDON, da notare che l'abbandono di un percorso porta al ritiro dell'ID di connessione, il ritiro dell'ID però non annuncia

necessariamente l'abbandono del percorso, infatti l'unico modo che ha effettivamente un host per rilevarne la chiusura sono dei segnali impliciti come i tempi di inattività o le perdite di pacchetti. Esistono anche altri meccanismi di chiusura espliciti di QUIC che si applicano a tutta la connessione, in particolare, la ricezione di un CONNECTION_CLOSE o di un Stateless Reset chiude la connessione.

PATH_ABANDON frame

Entrambi gli endpoint, il client e il server, possono avviare una chiusura di un percorso tramite l'invio di un frame PATH_ABANDON, che richiede al peer di interrompere l'invio di pacchetti con il corrispondente ID di destinazione della connessione.
Il mittente e il destinatario di un frame PATH_ABANDON non devono rilasciare le loro risorse immediatamente, ma dopo l'invio dell'ultimo pacchetto dovrebbero aspettare un tempo pari ad almeno tre volte l'attuale intervallo di timeout della sonda, che equivale al Probe Timeout (PTO), prima di inviare il frame RETIRE_CONNECTION_ID per il CID corrispondente. Solitamente, è previsto che il frame PATH_ABANDON sia utilizzato dal client per indicare al server che le condizioni del percorso sono cambiate, in modo tale che il percorso sia o non sia più utilizzabile.
Il frame RETIRE_CONNECTION_ID viene utilizzato anche per indicare al peer ricevente che il mittente non invierà più alcun pacchetto associato all'ID di connessione utilizzato su quel percorso.
Il destinatario di un frame PATH_ABANDON può inviare a sua volta un frame PATH_ABANDON con il quale indica la propria riluttanza a ricevere alcun tipo di pacchetto su questo percorso.
Questi frame possono essere inviati su qualsiasi percorso, non solo su quello che è destinato ad essere chiuso, ciò implica che un percorso può essere abbandonato anche se la sua connettività è già stata interrotta.
Frame ritrasmissibili, che sono già stati inviati sul percorso abbandonato e sono considerati perduti, saranno ritrasmessi su un percorso diverso. Se viene ricevuto un frame PATH_ABANDON per l'unico percorso attivo di una connessione QUIC, il peer ricevente dovrebbe inviare un frame CONNECTION_CLOSE ed entrare nello stato di chiusura. Se il client ha ricevuto un PATH_ABANDON per l'ultimo percorso aperto, potrebbe invece provare ad aprirsi un nuovo percorso se disponibile, e avviare la chiusura della connessione solo se la convalida del percorso non riesce o se viene ricevuto un CONNECTION_CLOSE dal server. Allo stesso modo il server può attendere per un tempo breve e limitato come un PTO se un pacchetto di rilevamento del percorso viene ricevuto su un nuovo percorso prima di inviare il CONNECTION_CLOSE.

Nell'immagine 4.1 viene mostrato ciò che è stato descritto a parole nelle sezioni precedenti

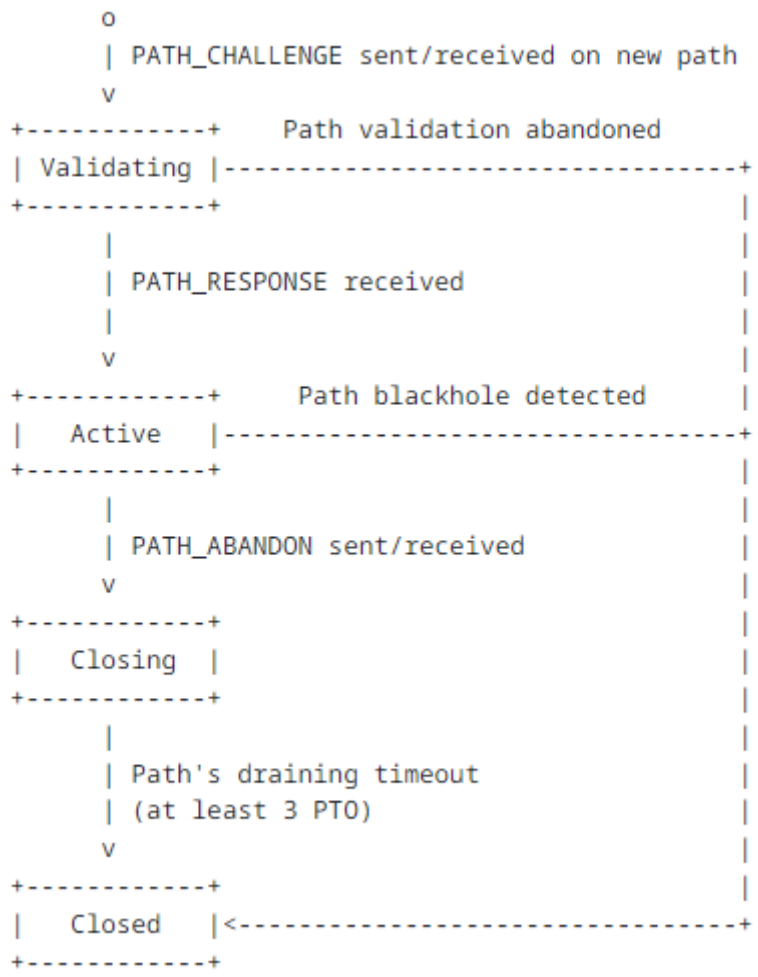

Figure 4.1: MPQUIC Path Management

4.2 CMT-SCTP

Il Concurrent Multipath Transfer over Stream Control Transmission Protocol (CMT-SCTP) è un protocollo progettato per migliorare le prestazioni e l'affidabilità della trasmissione dei dati su reti IP.
Introduce il concetto di trasferimento multipath simultaneo al protocollo SCTP, mirando ad aumentare il throughput, ridurre la latenza e migliorare la resilienza complessiva dei trasferimenti di dati.

4.2.1 SCTP

SCTP è un protocollo originariamente sviluppato per trasportare messaggi di segnalazione telefonica su reti IP, ma un continuo lavoro da parte dei ricercatori ha portato ad un evoluzione di SCTP, portandolo ad essere un protocollo di trasporto generico che include opzioni di consegna avanzate. SCTP, similmente a TCP fornisce una connessione full duplex affidabile, chiamata associazione. Un pacchetto SCTP, o più in generale, una Protocol Data Unit(PDU) è costituita da uno o più blocchi concatenati chiamati chunck, che possono essere di controllo o di dati. Ai fini dell'affidabilità e del controllo della congestione, ad ogni chunk di dati in un'associazione è assegnato un TSN (transmission Sequence Number) unico. Poichè i chunk sono atomici, i TSN sono associati a chunks of data, a differenza di TCP che associa un sequence number a ciascun ottetto del bytestream, ogni PDU trasporta ed è associato ad un singolo TSN. SCTP utilizza uno schema di riconoscimento selettivo simile al SACK TCP, gli ack SCTP portano le informazioni degli ack cumulativi e selettivi (anche detti gap ack)e vengono denominati come SACK. In questo "lavoro" verrà a volte utilizzato il termine "SACK" al posto di ack per denotare quando un ack porta le infomrazioni relative sia ad un ack cumulativo che selettivo.

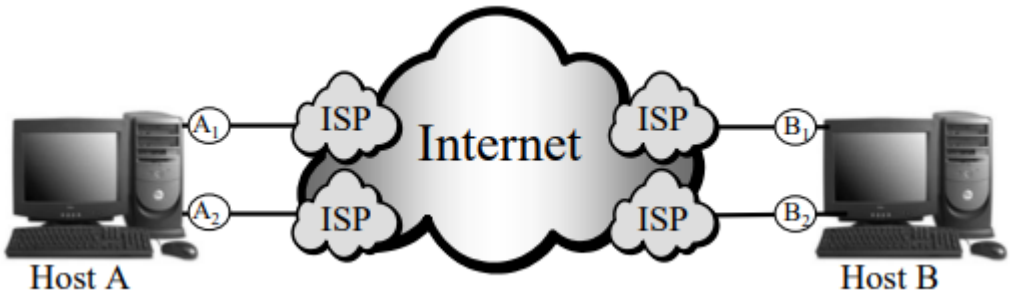

Figure 4.2: Esempio di topologia multihomed

A differenza di TCP, questo protocollo supporta il multihoming in modo da poter fornire ridondanza sul livello del percorso, aumentando così la capacità di sopravvivenza dell'associazione nel caso di un fallimento di un percorso di rete. Un endpoint SCTP può essere associato a più indirizzi IP durante l'inizializzazzione dell'associazione.
Facendo riferimento all'immagine sopra riportata (4.2) facciamo un rapido confronto fra TCP e SCTP per spiegare ulteriormente la funzione multihoming del protocollo preso in esame.
Fra gli host A e B sono possibili quattro tipi diversi di connessioni TCP distinte: (A1,B1), (A1,B2), (A2,B1), (A2,B2). SCTP, invece, non è obbligato a scegliere un singolo indirizzo IP su ciascun host, una singola associazione potrebbe consistere di due insieme di indirizzi IP, che nel esempio riportato potrebbero essere: (A1,A2, B1,B2). Un endpoint SCTP può collegarsi ad un sottoinsieme dei suoi indirizzi IP disponibili, questo legame permette al mittente SCTP di inviare dati ad un ricevitore multihomed attraverso diversi indirizzi di destinazione. Attualmente, SCTP utilizza il multihoming solo per scopi di ridondanza.
I nuovi dati devono essere inviati ad un'unica destinazione primaria, mentre le ritrasmissioni possono essere inviate a qualsiasi destinazione alternativa, da notare che in ciascuna di esse viene utilizzato un singolo numero di porta di un ednpoint indipendentemente dal numero degli indirizzi IP. Analogamente a TCP, SCTP utilizza tre variabili di controllo: la finestra annunciata dal destinatario (rwnd), la finestra di congestione del mittente (cwnd) e una slow start threshold del mittente (sstrhesh). Tuttavia, a differenza del cwnd del TCP, che riflette quali e quanti dati possono essere inviati, il cwnd di SCTP determina solo la quantità di dati che possono essere inviati. Dato che un'associazione SCTP consente endpoint di origine e destinazione mulihomed, una sorgente, in base alla destinazione, mantiene diversi parametri quali: cwnd, ssthresh e la stima del RTT. Un mittente SCTP mantiene anche un timer di ritrasmissone separato per destinazione, ma la rwnd di un'associazione è condivisa. Questa "opera" parte dal presupposto che le code dei bottleneck (in altre parole, i punti di congestione) sui percorsi end-to-end utilizzati in CMT siano indipendenti, infatti la sovrapposizione dei percorsi è accettabile fintanto che i bottleneck dei percorsi sono indipendenti. Due esempi significativi di questa ipotesi sono le reti di telefonia e le reti dei campi di battaglia.

- La comunicazione di segnalazione nelle reti di telefonia viene migrata sugli IP e utilizza SCTP per il trasporto. Dati i severi requisiti di disponibilità su queste reti, i dispositivi di segnalazione sono multihomed e sono interconnessi tramite più percorsi end-to-end indipendenti per motivi di tolleranza ai guasti. I percorsi end-to-end non condividono alcuna risorsa di rete e riescono così ad evitare ogni singolo point of failure.

- Il sistema di combattimento futuro, proposto dall'esercito americano, per le

reti sui campi di battaglia si equipaggerà di host mobili con interfacce multiple, che si connettono spesso a reti wireless indipendenti, ad esempio una radio terrestre a corto raggio e una comunicazione a lungo raggio con satelliti a bassa quota o geostazionari. Queste diverse comunicazioni forniranno più percorsi indipendenti tra i nodi alle tecnologie.

4.2.2 CMT

Per illustrare gli effetti del riordino introdotto in SCTP da CMT, si andrà ad utilizzare il setup proposto nell'immagine (4.4).

Due host dualhomed, il mittente A con indirizzi locali A1 e A2 e il mittente B con indirizzi locali B1 e B2, sono connessi attraverso due percorsi indipendenti: Percorso 1 (A1-B1), e il Percorso 2 (A2-B2) aventi lunghezze di banda end-to-end disponibili rispettivamente di 0.2 Mbps e 1 Mbps, con roundtrip propagation delay di entrambi i percorsi di 90 ms. Il mittente CMT A invia nuovi dati alle destinazioni B1 e B2 in maniera concorrente, in quando la larghezza di banda diventa disponibili in entrambi i percorsi, permesso dalle cwnds. Quando lo spazio cwnd è disponibile simultaneamente per due o più destinazioni, i dati sono inviati alle destinazioni in ordine arbitrario - policy di trasmissione ragionevole nel caso in cui il mittente CMT non abbia una conoscenza a priori delle caratteristiche dei percorsi.

Il setup proposto viene descritto tramite l'immagine (4.4) che mostra l'evoluzione della cwnd nel tempo, viene mostrata l'evoluzione della cwnd del mittente CMT nel tempo (1) per la destinazione B1, l'evoluzione della cwnd per B2 (2), il calcolo dell'evoluzione della cwnd aggregata (3) (somma di (1) e (2)) ma anche il risultato che ci si aspettava per la cwnd aggregata (4). Quest'ultima rappresenta l'obbiettivo iniziale per la performance di CMT, ovvero la somma delle cwnd di due connessioni SCTP utilizzando B1 e B2 come destinazioni primarie.
L'immagine mostra come, quando CMT viene utilizzato con SCTP senza nessuna modifica, il riordino ostacoli singnificativamente la crescita delle cwnd sia di B1 che di B2.

Da notare le diverse riduzioni delle cwnd sia per B1 che per B2, in particolare quella di B2 risulta a volte dimezzata.
Per riuscire a godere a pieno dei guadagni delle trasmissioni parallele, vanno prima gestite alcune problematiche: 1) fast retrasmission non necessarie nel mittente 2)riduzioni della crescita di cwnd dovute alle poche cwnd update nel mittente 3) l'incremento dell'ack traffic dovuto ai pochi delayed ack

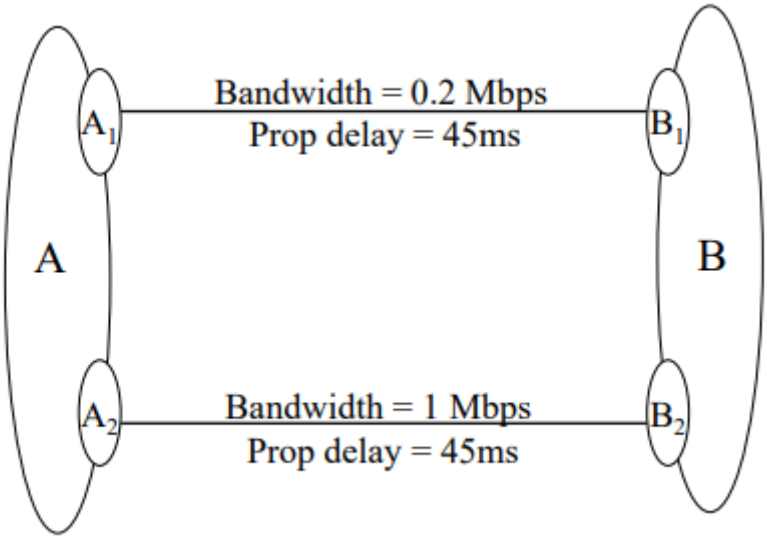

Figure 4.3: Simulazione per illustrare gli effetti del riordino

Prevenzione delle fast retransmission non necessarie

Quando si osserva il riordino, il ricevitore invia dei gap reports (gap acks) al mittente che li utilizza per individuare la perdita data da una procedura di fast retransmission simile alla Fast Retransmit TCP. Con CMT, possono essere causate delle fast retransmission dal riordino, che porta a conseguenze negative: 1) Dato che ogni ritrasmissione, si assume, che sia data da una perdita di congestione, il mittente riduce la sua cwnd per la destinazione sulla quale la ritrasmissione dei dati è stata ottimale 2) Il problema della sovracrescita porta una cwnd del mittente a crescere in maniera aggressiva per la destinazione sulla quale avviene la ritrasmissione, per colpa degli ack ricevuti durante la trasmissione originale.
Nell'immagine precedente, ogni riduzione di cwnd osservata per B1 e B2 è data da fast retransmission non necessarie, queste sono dovute a riordini "voluti" dal mittente e non dovute dall'effetto della rete.
L'interpretazione convenzionale dei chunk SACK in SCTP (opzioni SACK TCP) vede i gap reports come possibili perdite, la probabilità che un TSN sia perso è maggiore più sono i gap report ricevuti per quel TSN. A causa del riordino indotto dal mittente, un mittente CMT necessita di informazioni aggiuntive per individuare la perdita, i soli gap report non sempre ne implicano una, ma un mittente la può dedurre tramite l'utilizzo dei gap report e la conoscenza delle destinazioni di ogni TSN

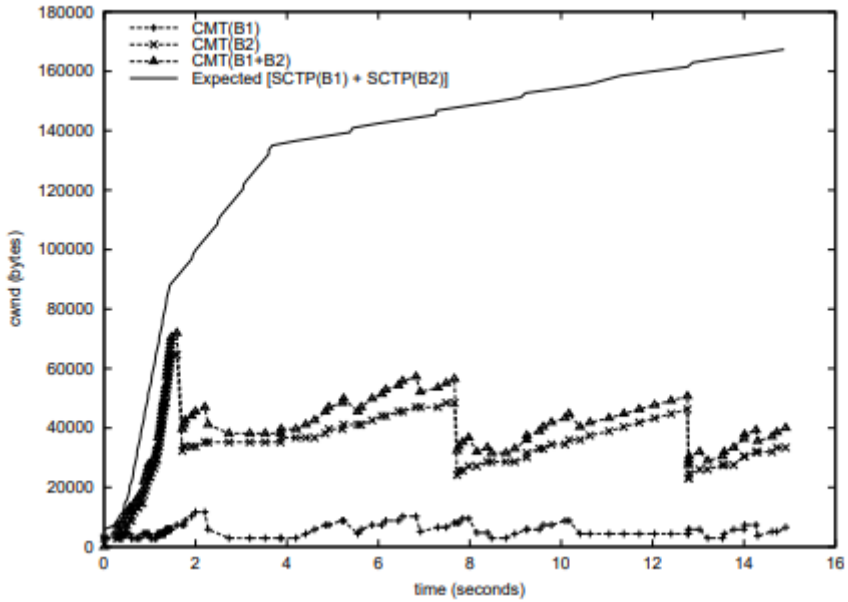

Figure 4.4: Evoluzione dell cwnd in CMT-SCTP

prevenzione della riduzione nelle cwnd updates

L'algoritmo di evoluzione delle cwnd per SCTP permette la crescita delle cwnd solo quando viene ricevuto dal mittente un nuovo cum ack. Quando vengono generati dei SACK con dei cum ack non modificati e successivamente arrivano al mittente quest'ultimo non modifica la cwnd. Dato che un ricevitore CMT osserva il riordino, vengono inviati molti SACK contenenti nuovi gap report ma non nuovi cum ack, quando questi gap vendono successivamente riconosciuti da un nuovo cum ack, avviene la crescita della cwnd, ma solo per i dati riconosciuti dalla SACK più recente, mentre i dati riconosciuti precedentemente attraverso i gap report non contriubuiscono alla crescita della cwnd. Questo comportamento previene improvvise crescite della cwnd risultando in invii di burst of data, anche se i dati possono aver raggiunto il destinatario "in-order per destination", senza modifiche l'attuale gestione delle cwnd, non si rifletterà nell'update delle cwnd.
Questa inefficienza può essere attribuita agli attuali principi di design dei cum ack in SACK, che tracciano gli ultimi TSN ricevuti in ordine nel destinatario, che vengono applicati all'intera associazione, non per destinazione. L'attuale SCTP utilizza un solo indirizzo di destinazione per trasmettere nuovi dati per permettere ai principi del design di funzionare al meglio, mentre CMT utiliza destinazioni multiple simultaneamente, portando la crescita cwnd in CMT a dover richiedere il tracking delle informazioni degli ultimi TSN ricevuti in ordine e non codificati direttamente in un SACK.

Incremento del curbing nel traffico ACK

Inviando un ack ogni due ricezioni di dati PDU, in SCTP viene ridotto il traffico ack nell'Internet, SCTP specifica che il ricevitore dovrebbe utilizzare un algoritmo delayed ack, dove gli ack vengono rallentati solo mentre il destinatario riceve i dati in ordine e i PDU riordinati dovrebbero essere riconosciuti immediatamente. Con il frequente riordino di CMT, questa regola porta il destinatario SCTP a ritardare frequentemente gli ack portando l'aumento del traffico ack che è un effetto negativo in CMT.
Per prevenire questo incremento, si porta il ricevitore CMT a non riconoscere immediatamente un PDU out-of-order, ma a ritardare gli ack, quindi il destinatario CMT ritarderà sempre gli ack, non curandosi del fatto che i dati siano ricevuti in ordine o meno, eliminando quasi completamente l'incremento del traffico ack.
Un elemento da sottolineare nel design SCTP è che i dati generalmente arrivano in ordine, quelli che sono ricevuti out-of-order indicano possibili perdite, per questo un destinatario dovrebbe immediatamente riconoscere i dati ricevuti da un gap in modo da accelerate il recupero della perdita grazie all'algoritmo di fast retransmission. In SCTP, quattro ack con un missing report per un TSN, indicano che il destinatario ha ricevuto almeno quattro PDU inviati successivamente al TSN mancante, e la ricezione di quattro missing report per un TSN fa scattare l'algoritmo fast retransmission, in altre parole il mittente ha un reordering threshold di quattro PDU. Dato che un ricevitore CMT non può distinguere fra perdita e riordino indotto dal mittente CMT, la modifica sopra riportata porterebbe ad un rallentamento degli ack anche di fronte ad una perdita, portando la fast retransmission ad essere attivata dopo otto PDU ricevuti successivi alla perdita.
L'incremento del reordering threshold nel mittente può essere contrastato riducendo l'attuale numero di ack richiesti per attivare la fast retransmission, incrementando il numero di missing reports registrati per ack. Quindi, se un mittente può aumentare il numero di missing reports per ack, sarà necessario un minor numero di ack per attivare la fast retransmission, un mittente in questo caso può fornire maggiori informazioni in ogni ack per essere assistito nel riferire il numero di missing report per ack per un TSN perso. Per questo in ogni ack, un mittente riporta il numero di PDU ricevuti da quando è stato inviato il precedente ack, un mittente poi deduce il numero di missing reports per TSN in base ai TSN che sono stati riconosciuti nel SACK, il numero di PDU riportati dal destinatario e la conoscenza delle destinazioni di ricezione di ogni TSN.

CMT retransmission policy

I percorsi multipli presentano un mittente SCTP con diverse opzioni su dove inviare la ritrasmissione, ma la scelta non è ben informata poichè SCTP limita l'invio

di nuovi dati che possono fungere da sonde per le informazioni (es lunghezza di banda, tasso di perdita, RTT) ad una sola destinazione primaria. Di conseguenza un mittente SCTP dispone di informazioni minime sui percorsi verso un destinatario diverso dal percorso primario, d'altra parte, un mittente CMT mantiene informazioni accurate su tutti i percorsi, poichè i nuovi dati vengono inviati a tutte le destinazioni contemporaneamente.
Queste informazioni consentono ad un mittente CMT di decidere su quale percorso sia meglio inviare la ritrasmissione, ovvero, il mittente può sceglere la destinazione di ritrasmissione per ogni loss recovery in modo indipendente, sfruttando le informazioni più recenti a sua disposizione.
Di seguito sono presentate cinque retransmission policies per CMT, in quattro di queste una ritrasmissione può essere inviata ad una destinazione diversa da quella utilizzata per la trasmissione originale. Precedenti ricerche sui criteri di ritrasmissione SCTP mostrano che l'invio di ritrasmissioni ad una destinazione alternativa degradano le prestazionie, a causa della mancanza di sufficiente traffico sui percorsi alternativi, mentre, come detto in precedenza con CMT, i dati vengono inviati contemporaneamente su tutti i percorsi. Le retransmission policies per CMT sono:

- RTX-SAME: una volta che un nuovo chunck di dati è stato pianificato e inviato ad una destinazione, tutte le ritrasmissione di quel blocco vengono inviate alla stessa destinazione (fino a quando la destinazione non è indicata come inattiva per un guasto).

- RTX-ASAP: Una ritrasmissione di un chunck di dati viene inviata a qualsiasi destinazione per la quale il mittente ha spazio cwnd disponibile, se più destinazioni hanno questa disponibilità ne viene scelta una casualmente.

- RTX-CWND: Una ritrasmissione viene inviata alla destinazione per cui il mittente ha cwnd maggiore, un pareggio porta ad una scelta randomica.

- RTX-LOSSRATE: Una ritrasmissione viene inviata alla destinazione con il minor loss rate, se più destinazioni presentano il medesimo loss rate ne viene scelta una casualmente.

- RTX-SSTHRESH: Una ritrasmissione viene inviata alla destinazione per cui il mittente ha l'ssthresh maggiore, nel caso di parità si sceglie casualmente.

Tra le policy presentate RTX-SAME è la più semplice, RTX-ASAP è una policy "hot-potato" (ritrasmissione appena possibile non curante dell loss rate), RTX-CWND e RTX-SSTHRSH in pratica tracciano e provano a spostare la ritrasmissione nel percorso con il loss rate minore, RTX-LOSSRATE utilizza informazioni syl loss rate fonite da un "oracle", informazioni che RTX-CWND e RTX-SSTHRESH stimano.

4.3 MPMTP-AR

MPMTP-AR funziona su UDP e funge da intermediario tra i programmi dell'applicazione e le operazioni del livello di trasporto e fornisce comunicazione process-to-process, ovvero un tipo di comunicazione che utilizza più percorsi contemporaneamente per fornire i dati. Questo protocollo fornisce, non solo un metodo di consegna affidabile, ma offre anche un sistema di comunicazione full duplex, una comunicazione bidirezionale e simultanea, come avviene in una conversazione faccia a faccia fra due persone, così due utenti, che devono stabilire o terminare una connnessione, possono mantenere il proprio buffer di ricezione e di invio e utilizzare il meccanismo di riconoscimento per garantire un arrivo corretto dei dati.
MPMTP-AR utilizza estensioni di intestazione per supportare funzioni aggiuntive non definite da MPTP, così vengono definiti due tipi di pacchetto, vale a dire il pacchetto di dati e il pacchetto di controllo.
Il primo è costituito da un'intestazione di 16 byte, in questi 16 byte sono inclusi il type of service (TOS), checksum, PID, TSN e FSN. Il TOS è simile a quello che si trova nel pacchetto di intestazione di Internet e rappresenta la sua priorità, se il server di inoltro incontra una congestione, scarterà per primi i pacchetti di dati con la priorità più bassa.
Il pacchetto di controllo invece è formato da un'intestazione di 8 byte. Il campo di intestazione del controllo del pacchetto include il tipo del control packet type (CT), la lunghezza e il PID. Il CT indica diversi tipi di pacchetti di controllo MPMTP-AR definendone tre classi:

- Gestione delle sessioni del pacchetto
- feedback
- enanched selective acknowledgment (ESACK)

4.3.1 Sessione MPMTP-AR

Questo protocollo è un tipo di protocollo connection-oriented, stabilisce una connessione virtuale che è composta da più percorsi, tra il mittente e il destinatario, e tutti i pacchetti vengono inviati tramite questa connessione.
Prima di iniziare la trasmissione dei dati, l'utente utilizza il tipico handshaking a tre vie per stabilire la connessione, l'utente negozia il valore iniziale di TSN, della dimensione della finestra di ricezione e il numero di percorsi da utilizzare, e a differenza di TCP non consente una sessione half-close, quindi la connessione da bidirezionale non può diventare unidirezionale.
Se un utente chiude la sessione, l'altro utente con cui è stata stbilita la connes-

sione deve smettere di ricevere dati dall'applicazione del livello superiore, ma deve limitarsi ad inviare i dati rimanenti e chiudere la sessione.

4.3.2 Operazioni Base

In questa sezione verranno presentate le operazioni base del protocollo:

- Finestra in MPMTP-AR: l'utente mantiene una finestra di invio condivisa a livello di sessione e una finestra di ricezione condivisa sullo stesso livello per tutti i percorsi. La dimensione della finestra di invio è determinata dal controllo del flusso e dal controllo di congestione, la finestra di invio invece è più simile a quella in TCP, ma con una differenza nel numero del timer, infatti TCP mantiene un solo timer, mentre MPMTP-AR mantiene un timer dedicato per ogni invio.

- Feedback: il destinatario invia il receiver report (RR) per informare il mittente delle trasmisisoni dello stato di ogni percorso. RR trasporta i byte dei byte ricevuti correttamente e il massimo TSN/FSN dell'ultimo pacchetto ricevuto correttamente, per garantire un arrivo corretto, vengono trasmessi gli RR lungo il percorso con le migliori prestazioni. In un ambiente di rete stabile, il feedback viene inviato a una velocità relativamente costante, mentre in un ambiente di rete dinamico è necessario un feedback tempestivo affinchè il mittente si adatti rapidamente agli errori di trasmissione.

- Controllo del flusso: MPMTP-AR separa il controllo del flusso dal controllo della congestione. Assumiamo che la connessione virtuale tra il mittente e il destinatario sia priva di congestione quando parliamo del controllo del flusso. I messaggi ESACK vengono utilizzati per informare il mittente sulla dimensione della finestra di ricezione, l'aggregazione delle finestre dei vari percorsi disponibili è uno dei fattori che determinano la dimensione della finestra di invio a livello di congestione.

- Controllo degli errori: il meccanismo di controllo degli errori ritrasmette i pacchetti danneggiati o persi e scarta i pacchetti duplicati, in questo processo il protocollo è aiutato da tre semplici strumenti, il checksum, l'ESACK e il timer.
  Il checksum, simile a quello in TCP, viene utilizzato per verificare la presenza di pacchetti danneggiati, se un pacchetto ha un checksum non valido viene scartato o considerato come perso, e verrà poi ritrasmesso dal mittente. Quando quest'ultimo invia il pacchetto, lo memorizza in un buffer di invio fino a quando il pacchetto non viene riconosciuto.

In questo protocollo la ritrasmissione è più complessa rispetto a quella in TCP, da un lato si hanno più percorsi con diverse caratteristiche e il mittente imposta un timer per ogni percorso, dall'altra un pacchetto è ritrasmesso quando il timer di ritrasmissione scade o quando il mittente riceve più ACK duplicati dal percorso sul quale il pacchetto è stato inviato. Il numero di ACK duplicati del percorso con un round trip time (RTT) inferiore è inferiore a quello del percorso con un RTT maggiore.

- Controllo di congestione: il meccanismo di controllo della congestione aiuta il mittente a evitare la congestione. Il mittente mantiene una finestra di ricezione per la sessione e una di congestione condivisa tra tutti i percorsi, la dimensione effettiva dei permessi di invio è il valore minimo di queste due finestre. Normalmente il controllo di congestione comprende 3 fasi:

  - slow start
  - rilevamento della congestione
  - evitamento della congestione

  Quando si verifica una congestione, il mittente riduce di molto il moltiplicativo della finestra di congestione. Nello scenario di trasporto multipath, la congestione di un percorso non è uguale alla congestione di tutti i percorsi, per questo il meccanismo di controllo della congestione deve valutarne il livello per permettere al mittente di diminuire la finestra di congestione e ridistribuire il carico originariamente assegnato al percorso.

- Distribuzione dei dati: per migliorare le prestazioni di trasmissione, il mittente distribuisce i dati a percorsi multipli in modo ottimale, tramite l'utilizzo di uno schema di distribuzione, come ad esempio l'algoritmo round-robin che assegna ugualmente i dati a più percorsi. Sebbene l'algoritmo sia facile da implementare non riesce a migliorare significativamente l'efficienza della trasmissione neanche provando a ridurre la portata. La distribuzione dei dati dovrebbe considerare la qualità del percorso. MPMTP-AR mantiene un buffer di invio a livello di percorso per ognuno di essi, come mostrato in figura 4.5, il mittente distribuisce il traffico ad un buffer a livello di percorso in base alla qualità di quest'ultimo. Ogni percorso trasmette un pacchetto indipendentemente da quale allevii il problema dell'head of blocking.

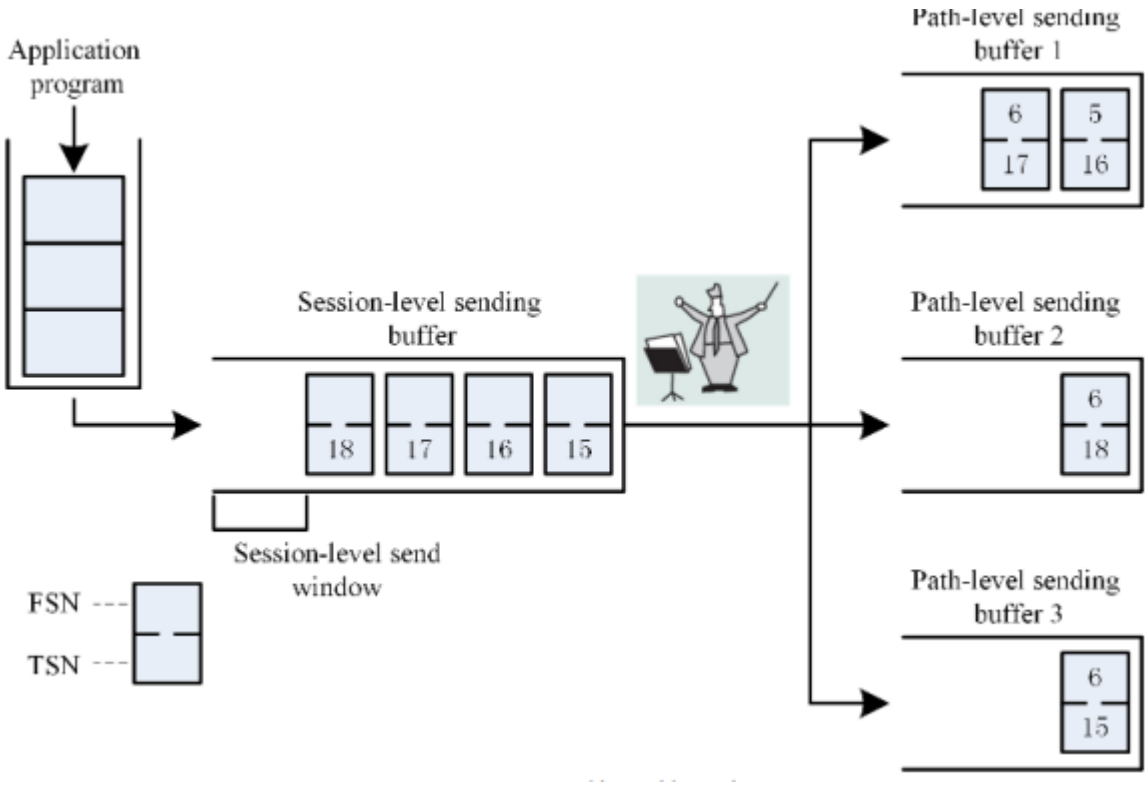

Figure 4.5: Data distribution

- Riassemblaggio dati: lato ricevitore, MPMTP-AR riassembla i pacchetti ricevuti, come possiamo notare nella figura 4.6, i pacchetti ricevuti subiscono due tipi di jitter (è la misura dell'irregolatà e dell'inconsistenza del tempismo del packet delivery, è la deviazione della latenza media), il primo è all'interno del percorso mentre l'altro si trova su più percrosi. Il ricevitore utilizza un buffer di ricezione a due livelli per assorbire i jitter e riordinare i pacchetti ricevuti. Il ricevitore utilizza un buffer sul percorso per raccogliere lo stato di ricezione e riordina i pacchetti ricevuti da FSN, il buffer "pubblica" il riassemblaggio nella sessione, questo è un passaggio necessario per ogni buffer di ricezione sul percorso, e successivamente esegue il push dei pacchetti al buffer di ricezione sulla sessione. Come si può notare nell'immagine sottostante, per quanto riguarda la sessione, il buffer di ricezione è in attesa del pacchetto con TSN di 16, il buffer del percorso uno invia immediatamente il pacchetto atteso al buffer di ricezione della sessione dopo averlo ricevuto. Il buffer di ricezione nella sessione ordina i pacchetti ricevuti da TSN.

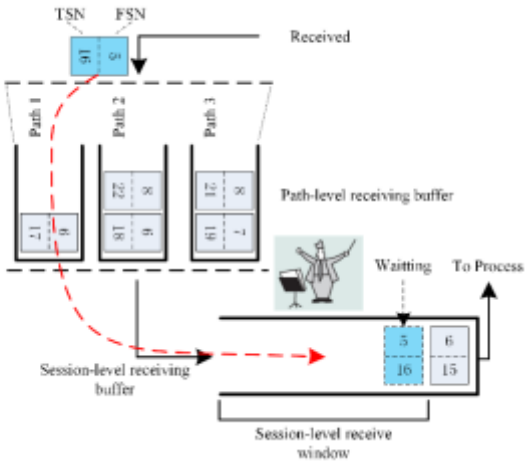

Figure 4.6: Data reassembling

Chapter 5

Comparazione

Nel seguente capitolo andremo a confrontare i protocolli presentati nel capitolo 4 con il protocollo MPTCP, attualmente il protocollo più diffuso.

5.1 MPQUIC e MPTCP

Esistono diversi approcci per valutare le prestazioni di un protocollo di trasporto, in questo documento ci basiamo sulle misurazioni eseguite sulla piattaforma di simulazione Mininet con complete implementazioni MPQUIC e MPTCP, utilizzando quest'ultimo come linea guida. Per fornire una valutazione equa delle implementazioni confrontate, utilizziamo un approccio di progettazione sperimentale simile a quello utilizzato per MPTCP che copre un'ampia gamma di parametri.

5.1.1 Large file download

Il primo scenario è il download di file di grandi dimensioni su un singolo flusso, qui, un host multihomes vuole ridurre al minimo il tempo necessario per il download e massimizzare così l'aggregazione della banda disponibile.
Considerando una rete multipath con due host multihomed su percorsi disgiunti con caratteristiche diverse, come visto nell'immagine 5.1, in questo caso, le prestazioni di protocolli multipath sono in funzione della larghezza di banda dei collegamenti, del tempo di andata e ritorno, della presenza di un bufferbloat (ritardo di accodamento) e le random packet losses. Il design sperimentale proposto seleziona i valori di questi parametri, utilizzando l'algoritmo WSP negli intervalli elencati nella tabella (immagine 5.2), raggruppiamo così le simulazioni in quattro classi: ambienti con un basso ritardo di larghezza di banda e senza perdite casuali (low-BDP-no-loss), ambienti con un basso ritardo della larghezza di banda e con perdite casuali (low-BDP-losses), ambienti con un alto ritardo di larghezza di banda e senza

perdite casuali (high-BDP-no-losses), ambienti con un alto ritardo di larghezza di banda e con perdite casuali(high-BDP-losses).

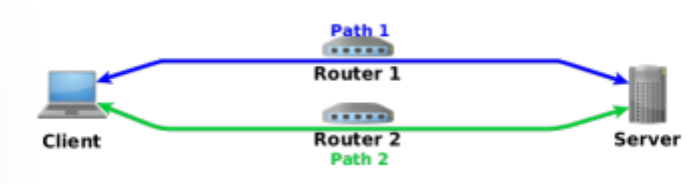

Figure 5.1: Esempio di rete con due host e con percorsi disgiunti

	Low-BDP		High-BDP	
Factor	Min.	Max.	Min.	Max.
Capacity [Mbps]	0.1	100	0.1	100
Round-Trip-Time [ms]	0	50	0	400
Queuing Delay [ms]	0	100	0	2000
Random Loss [%]	0	2.5	0	2.5

Figure 5.2: Parametri del design sperimentale

Per ogni classe si vanno a considerare 253 scenari e viene variato il percorso utilizzato per avviare la connessione, portando a 506 simulazioni per ogni protocollo, in modo da analizzare la corsa mediana.

- Low-BDP-no-losses: In questo caso MPQUIC performa meglio rispetto a MPTCP. Il primo dato considerato è il rapporto tra il ritardo che si ha nel ricevere un file su TCP diviso per il tempo richiesto con QUIC, se questo rapporto è uguale a 1, entrambi i protocolli sono equivalenti, mentre se il rapporto è maggiore di 1, TCP è più lento di QUIC.
  L'immagine 5.3 fornisce la funzione di distribuzione cumulativa (CDF) di questo rapporto, in tutti gli scenari considerati. Quando viene utilizzato un singolo percorso, non ci sono grandi differenze fra TCP e QUIC, ciò è dato dal fatto che in questo scenario, il controllo della congestione è il principale fattore di influenza ed entrambi utilizzano CUBIC. Per quanto riguarda il multipath, QUIC supera TCP nell'89 % dei casi, infatti MPTCP affronta il blocco head-of-line più spesso rispetto a MPQUIC, con dei percorsi eterogenei, MPTCP tende ad inviare burst of packets sul percorso più lento. Questo potrebbe essere correlato alle ambiguità legate alla stima del RTT del kernel Linux che porta lo scheduler a preferire il percorso più lento quando il percorso più rapido è carico, portandolo ad un blocco HoL. Per

queste condizioni MPTCP utilizza il meccanismo Opportunistic Retransmission and Penalisation (ORP) che porta i pacchetti ad essere ritrasmessi attraverso il percorso più veloce, limitando il goodput. Sotto le stesse condizioni, MPQUIC è meno "aggressivo" verso il percorso lento, grazie alla sua accurata stima della latenza del percorso, eviatndo così le limitazioni del buffer di ricezione grazie alla trasmissione di un frame WINDOW_UPDATE su tutti i percorsi disponibili. Senza perdite casuali la ritrasmissione è rara.

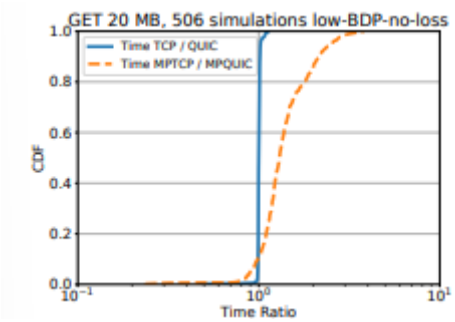

Figure 5.3: TCP e QUIC hanno performance simili, mentre MPQUIC permorma meglio di MPTCP

- Low-BDP-losses: MPQUIC anche qui risulta più performante rispetto a MPTCP. Le perdite casuali, come quelle che si verificano nelle connessioni wireless, infieriscano sulle prestazioni di protocollo di trasporto affidabile. Come mostrato nella tabella (immagine 5.2) si considerano le perdite casuali fino al 2,5 % che rientrano nei tassi di ritrasmissione sperimentati dall'implmentazione di Google. L'immagine 5.4 mostra che in questi scenari MPQUIC ha quasi sempre prestazioni migliori rispetto a MPTCP, questo è dovuto principalmente al frame ACK che può riconoscere fino a 256 packet number ranges, range molto più ampio rispetto ai 2-3 blocchi che possono essere ricnosciuti sall'opzione SACK TCP, a seconda di quanto spazio consumino le altre opzioni. Pertanto le prime ritrasmissioni sono più efficaci in QUIC che soffre anche meno il blocco HoL.

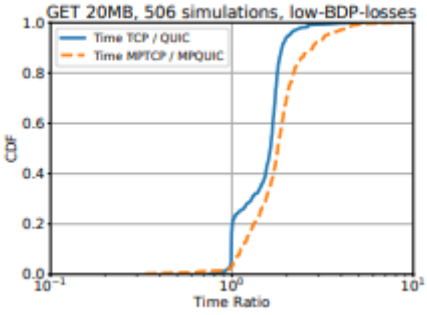

Figure 5.4: MPQUIC reagisce più velocemente rispetto a MPTCP

- High-BDP-no-losses: MPQUIC in questa situazione si comporta bene, in questi scenari, i benefici dell'aggregazione sperimentale di MPTCP diminuiscono, come si può notare dall'immagine 5.5. In queste reti, le connessioni subiscono principalmente i bufferbloat e i blocchi HoL dati dalle limitazioni della finestra di ricezione, l'utilizzo di più percorsi con caratteristiche diverse non previene questi problemi, anzi, rischia di accentuarli se i percorsi presentano latenze molto diverse, a causa dell'ulteriore ritardo generato nello stabilire il percorso, MPTCP favorisce il percorso iniziale poichè ha più tempo per aumentare la finestra di congestione portandolo ad essere più fragile nei confronti del buffferbloat. Dato che tutti i percorsi possono iniziare ad inviare i dati una volta stabilira la connessione, le finestre di congestine si evolvono in modo più equo portando le problematiche sopra citate ad essere meno influenti in MPQUIC, infatti, indipendentemente dal percorso iniziale, il multipath è più vantaggioso degli scenari a percorso singolo con QUIC nel 58% dei casi, mentre per MPTCP e TCP scende fino al 20%.

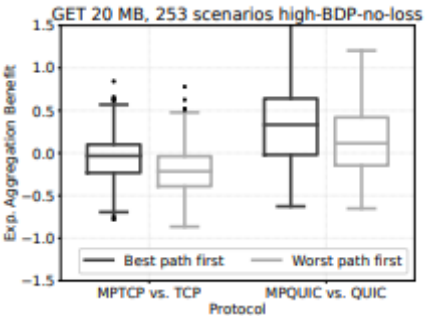

Figure 5.5: MPQUIC risulta vantaggioso anche per gli scenari ad alto BDP

- High-BDP-losses: MPQUIC affronta meglio le perdite casuali rispetto a

MPTCP. La segnalazione di perdita migliora, più precisamente la stima della latenza e l'equità nell'evoluzione della finestra di congestione dei percorsi con MPQUIC

5.1.2 Short file Downloads

Lo scenario successivo considera il donwload di un file da 256kB su un singolo flusso. A causa dei limiti di spazio si discute solo dello scenario low-BDP-no-losses. MPQUIC anche in questo caso performa meglio di MPTCP. Per trasferimenti brevi, la realizzazione della connessione è una frazione non trascurabile del totale del tempo di trasferimento, con QUIC l'handshake consuma un singolo RTT, con TCP invece vengono consumati 3 RTT insieme, questo ritardo potrebbe essere ridotto tramite l'utilizzo dell'opzione Fast Open.

5.2 CMT-SCTP e MPTCP

Come presentato in [3] la configurazione di Internet utilizzata per i test permette ai protocolli multipath di operare come previsto.
Le prime misurazioni hanno mostrato una differenza significativa tra MPTCP e CMT-SCTP, e il fattore principale di ciò è identificato nelle strategie di Path Management.

Mentre MPTCP costruisce una rete mesh utilizzando tutte le coppie di indirizzi disponibili, CMT-SCTP identifica un percorso tramite una coppia source e destination address, crea un percorso aggiuntivo per source address e , più importante, lo instrada all'hop successivo del source address (selezione del source address).
Questa non fa differenza negli scenari semplici con solo due percorsi disgiunti o addirittura in senari completamente meshed up (qui tutti i punti di accesso forniscono la stessa capacità). La scelta della strategia ha un impatto significativo in più topologie Internet complesse e asimmetriche, per SCTP, inoltre la selezione della coppia source address/destination address utilizzata per l'handshake iniziale determina il percorso primario (e di conseguenza anche le opzioni per i percorsi aggiuntivi rimasti). Se vengono scelte combinazioni sfavorevoli, si potrebbe avere un impatto significativo sul throughput ottenibile. La verifica e la quantificazione dell'impatto delle strategie della gestione del percorso, porta ad eseguire una serie di misurazioni in cui si utilizzano ognuna delle quattro coppie di indirizzi per l'avvio delle connessioni MPTCP e CMT-SCTP.

La figura 5.6 mostra come il throughput della connessione MPTCP superi notevolmente il throughput massimo rilevato per le connessioni a percorso singolo, portando alla conferma dei vantaggi che offre il traferimento multipath, risulta anche superiore a CMT-SCTP in tutti i casi singlepath. Questo vantaggio è dato dall'utilizzo dei percorsi aggiuntivi da parte di MPTCP che porta effettivi vantaggi in questo tipo di scenari, il throughput MPTCP non dipende in modo significativo dalla coppia di inidirizzi utilizzata per configurare la connessione iniziale, in quanto le variazioni sono sostanzialmente coperte dagli intervalli di confidenza.

Per quanto riguarda CMT-SCTP invece, la scelta della coppia di indirizzi iniziale incide significativamente sul throughput ottenibile, il throughput singlepath risulterebbe migliore se l'interfaccia DSL fosse utillizata in Essen, mentre per CMT-SCTP il throughput è risultato migliore quando la connessione è stata stabilita con l'interfaccai DFN dall'inizio. Questo dipende da una moltitudine di fattori, correlati al protocollo e alla configurazione di Internet.
Per continuare ad analizzare il problema della gestione del percorso, è stato esteso ulteriormente il banco di prova, in modo da mostrare che questo è un effetto

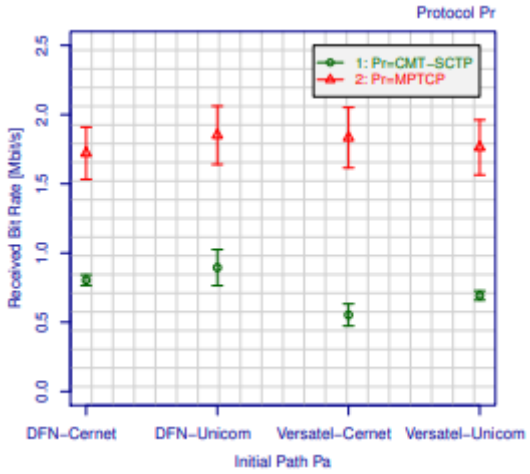

Figure 5.6: Valori througput su test intercontinentali

generale in una rete mesh con accessi asimmetrici. In contrasto con lo scenario iniziale, tutti e quattro i percorsi possibili (P_{S1-R1}, P_{S2-R2}, P_{S2-R1} e P_{S1-R2}) possono essere utilizzati. Sono stati configurati anche le restrizioni della larghezza di banda su ogni collegamento in base alla situazione del testbed, i collegamenti S1 e R1 sono stati limitati a 800 Kbit/s e il collegamento R2 a 3 Mbit/s. La larghezza di banda di S2 è variata da 200 Kbit/s a 3 Mbit/s. E' stato impostato un ritardo di 200 ms sui collegamenti S1 e S2 in modo che corrispondano al delay end-to-end misurato.
I risultati sono presenti nell'immagine 5.7.

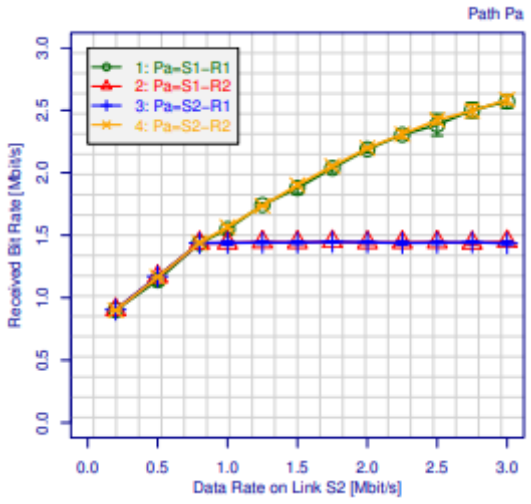

Figure 5.7: Differenze nel througput tramite l'utilizzo di diversi percorsi primari

Come previsto, se il percorso iniziale è impostato tramite il collegamento a bassa velocità al mittente e tramite un collegamento ad alta velocità al destinatario - o viceversa: il throughput non beneficia del collegamento R2 più veloce facendo risultare la scelta iniziale delle coppie di indirizzi cruciale. Dato che la configurazione di accesso della parte remota è genericamente non nota, una scelta intelligente richiederebbe misurazioni più complesse prima dell'impostazione della connessione (non sempre buona). Un rimedio ovvio sarebbe quello di adottare la gestione del percorso di tipo mesh di MPTCP. Tuttavia la decisione per la strategia CMT-SCTP era consapevole, l'obbiettivo era quello di mantenere la scalabitlià del protocollo per, ad esempio, scenari con più di due indirizzi, comuni per le configurazioni dual stack IPv4/IPv6, un altro obbiettivo era quello di evitare di mantenere troppe informazioni sul livello di rete nel livello di trasporto, in quanto è soggetto a cambiamenti dinamici e di conseguenza richiede uno sforzo gestionale. Dalle strategie presentate possiamo dire che MPTCP performa meglio nel creare una rete mesh rispetto a CMT-SCTP anche se in reti più complesse, dove ad esempio possono esserci più di due indirizzi per endpoint, si potrebbero avere problemi di scalabilità e risultare meno efficente di CMT-SCTP

5.3 MTMTP-AR e MPTCP

Sia MPMTP-AR che MPTCP sono protocolli progettati per fornire un servizio di consegna dati affidabile su molteplici percorsi.
Questa sezione confronta MPMTP-AR con MPTCP in termini di througput, tasso di perdita dei pacchetti, dimensione del buffer di ricezione e ritardo di riassemblaggio.
In una rete reale, un client utilizza MPMTP-AR e MPTCP per scaricare un file da un server. In base all'unità massima di trasferimento, il file da consegnare viene diviso in pacchetti di egual lunghezza.
Nel caso di MPTCP, il client utilizza il controllo di congestione Reno per gestire la finestra di congestione e un buffer di ricezione condiviso per riodrinare i pacchetti.
In MPMTP-AR invece, il client utilizza il Circuit-Switched Data (CSD), metodo di trasmissione di dati su una rete che utilizza canali a larghezza di banda fissa, per regolare la finestra di congestione e sfrutta il buffer di ricezione a due livelli per riordinare i pacchetti. Il bottleneck link di ogni percorso ha una larghezza di banda di 1 Mbps e il tasso di perdita ha una distribuzione uniforme che varia dallo 0,75% all'1,25%.
Le immagini da 5.8 a 5.11 mostrano la traccia raccolta da MPMTP-AR e MPTCP.
Per quanto riguarda l'immagine 5.8, il numero di percorsi disponibili varia da 1 a 5. Con il crescere dei percorsi disponibili si ha un incremento del througput e una diminuzione dei miglioramenti, entrambi i protocolli però non riescono a massimizzare l'utilizzo dei percorsi multipli, questo è dato dal fatto che per gestire un numero di percorsi sempre più alto i protocolli necessitano di costi sempre più elevati.
Un valore ragionevole di α contribuisce inoltre ad una valutazione più accurata della qualità del percorso. I risultati sperimentali dimostrano come MPMTP-AR ottenga il througput più elevato quando si ha $\alpha = 0{,}8$, indipendentemente dal numero di percorsi che si stanno utilizzando.

Nella figura 5.9, si può notare come con l'aumentare del numero di percorsi disponibili non incida sul tasso di perdita di MPMTP-AR rimanga stabile a differenza di quello di MPTCP aumenta. Questa differenza è data dalla rilevanza dei percorsi disponibili, percorsi multipli forniti dalla sovrapposizione di rete risultano meno rilevanti rispetto a quelli dati da indirizzi IP multipli accoppiati.
In MPMTP-AR, il tasso di perdita è minore quando $\beta = 0{,}07$, β è il valore che regola la frazione della larghezza di banda per il feedback per un passaggio moderato, il quale riflette tempestivamente l'andamento della variazione della path condition evitando di essere influenzato da fattori stocastici, come la latenza, la qualtà del collegamento o fallimenti della rete.
Durante la trasmissione, ci sono due percorsi tra il mittente e il destinatario.

L'immagine 5.10 mostra l'occupancy probability of receiving buffer. L'occupazione media passa dai 19,7 pacchetti del protocollo MPTCP ai 16,6 pacchetti del MPMTP-AR, mentre il buffer di ricezione a due livelli aiuta il ricevitore a ridurre la dimensione del buffer occupato dai blocchi discontinui, come ad esempio la ricezione di due pacchetti del destinatario. Un pacchetto Ha un TSN di 1 e un FSN di 1 dal percorso 1, mentre l'altro pacchetto ha TSN 3 e FSN 2 dal percorso 2. Nel caso in cui sia presente un solo buffer di ricezione, i due pacchetti occuperebbero lo spazio di 3 pacchetti, il meccanismo del buffer a due livelli necessita solamente di due pacchetti di spazio, potendo cosi memorizzare questi due pacchetti nel buffer di ricevimento al corretto livello di percorso.
L'imagine 5.11 invece mostra il ritardo, ovvero la durata dall'ingresso del pacchetto nel buffer di ricezione al momento in cui tale pacchetto potrà essere consegnato all'applicazione.
Si può notare come, il ritardo medio nel caso di MPTCP sia di 98ms, mentre in MPMTP-AR sia di 85ms, il buffer di ricezione a due livelli accelera il rilevamento e la ritrasmissione dei pacchetti persi.
Entrambi i protocolli presentano proprietà multimodali nella distribuzione delle probabilità, questo è principalmente dato dall'intervallo di tempo della procedura di riassemblaggio. Ad esempio, nel caso di MPMTP-AR, la maggior parte delle durate rientrano in una distribuzione uniforme che varia tra i 10 e i 160 ms. L'intervallo di tempo della procedura di riassemblaggio è di circa 5 ms.

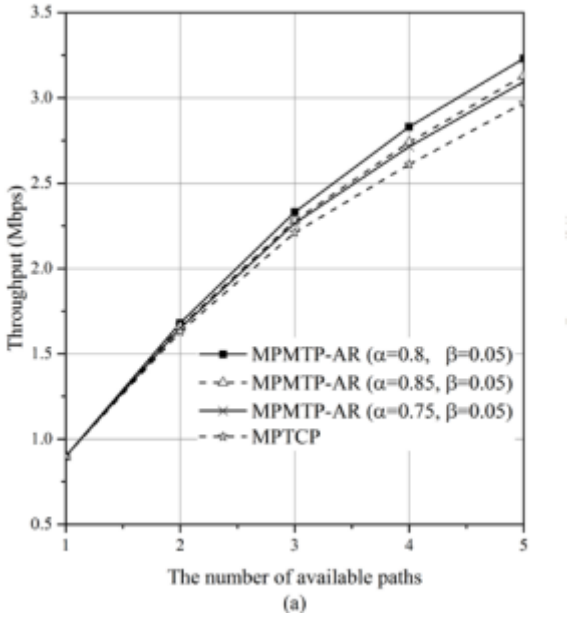

Figure 5.8: Comparazione del Throughput con diversi valori α e β

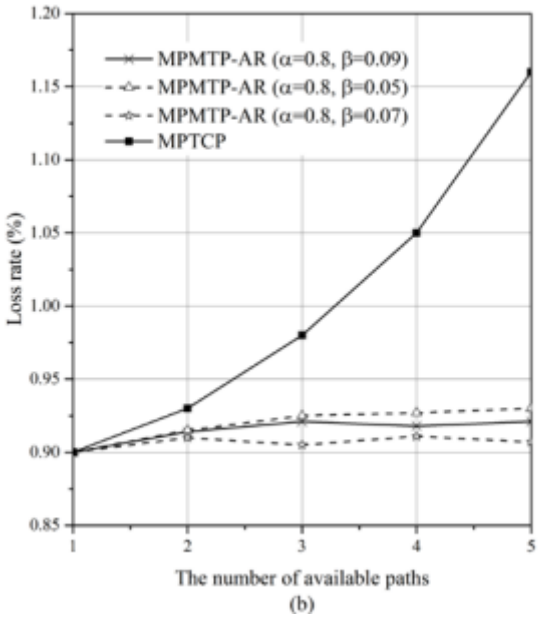

Figure 5.9: Comparazione del Loss Rate con diversi valori α e β

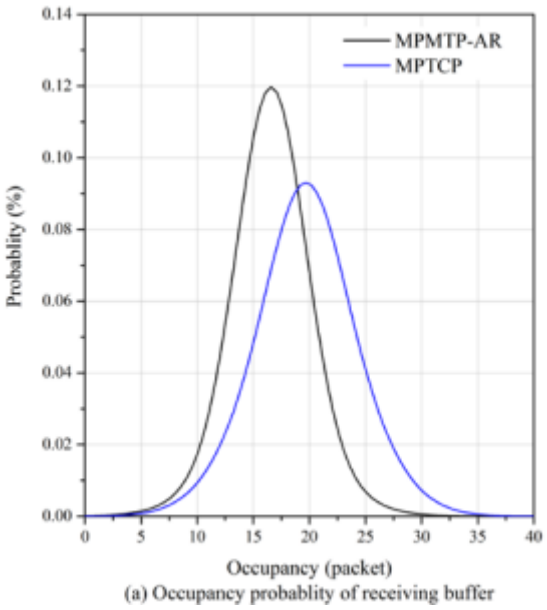

Figure 5.10: Comparazione dei receiving buffer

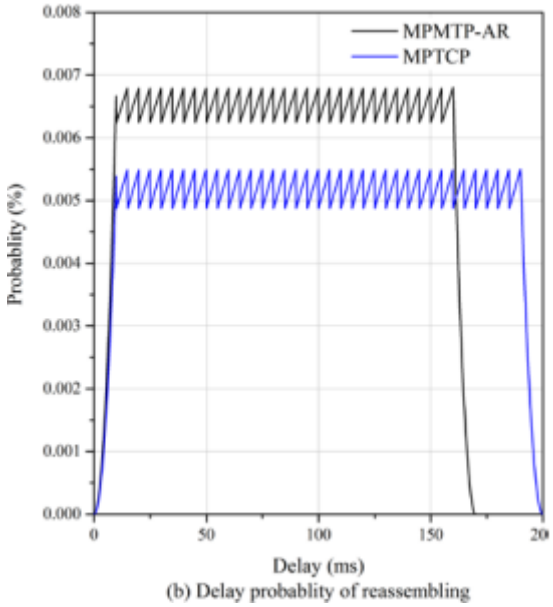

Figure 5.11: Comparazione del reassembling delay

Chapter 6

Conclusioni

In conclusione, dopo aver esaminato e comparato diversi protocolli multipath è evidente che ognuno di loro abbia i suoi punti di forza e le sue debolezze e che le scelta del protocollo dipende dalle specifiche richieste dalla rete .
MPQUIC In questo articolo abbiamo potuto analizzare un'estensione di QUIC che consente a questo nuovo protocollo di utilizzare più percorsi contemporaneamente, abbiamo potuto anche valutare delle prestazioni su migliaia di scenari Mininet che coprono una vasta gamma di paramentri dimostrandoci come MPQUIC possa fornire maggiori vantaggi rispetto a MPTCP, risvolto notato anche dalla migliore gestione delle perdite di pacchetti.
MPMTP-AR Abbiamo potuto anche analizzare e valutare un'implementazione MPMTP-AR che fornisce un servizio di consegna efficace, utilizza un three-way handshake per stabilire una connessione con il livello dell'applicazione, i pacchetti di dati sono trasmessi dal mittente al ricevitore tramite un applicazione UDP-based, il meccanismo di ritrasmissione per recuperare i pacchetti persi, prende in considerazione il peso del carico del percorso congestionato per sistemare la congestione. Abbiamo anche potuto confrontare i risultati di MPMTP-AR con MPTCP notando che il primo performa meglio dell'ultimo nella maggior parte dei casi.
CMT-SCTP Abbiamo anche presentato un comparazione fra CMT-SCTP e MPTCP e nonostante le buone prestazioni in scenari semplici con due percorsi disgiunti, si riscontrano comunque dei problemi. Uno, ad esempio, è quello del mantenimento delle gap list per il Selective Acknowledgement, che può avere un impatto sulle performance in determinate situazioni reali, inoltre la strategia di MPTCP di creare una rete full mesh di percorsi performa meglio rispetto alla scelta di CMT-SCTP, queste scelte hanno però risultato opposto se si valuta la scalabilità del protocollo, cosa che risulta meglio affrontabile per CMT-SCTP.

Chapter 7

Ringraziamenti

Per concludere questo elaborato vorrei menzionare le persone che hanno reso possibile il raggiungimento di questo traguardo. Voglio ringraziare in primis i miei familiari, perchè questi anni per me sono stati molto duri, non sono riuscito a vivere bene l'esperienza universitaria portando malumore all'interno delle mura domestiche, ma nonstante questo sono sempre stati i primi ad incoraggiarmi e tranquillizzarmi fino all'ultimo giorno senza mai mollare, vorrei ringraziare la "compagnia del punto e virgola" che sono le persone che più mi hanno aiutato nello studio e nel superamento degli esami, non avrei potuto chiedere compagni di Università migliori, vi devo una grande parte della mia Laurea, senza di voi di certo non sarei qui oggi, la strada sarebbe stata ancora lunga, poi vorrei anche ringraziare mio cugino, tutti i miei amici e le persone a me vicine, ognuno di voi mi ha aiutato a costruire la strada che mi ha portato alla fine di questo percorso.
Per concludere vorrei fare un piccolissimo ringraziamento a me stesso, Nicola, finalmente sei arrivato in fondo, ne sono successe tante, ma vorrei che ti guardassi intorno ora, guarda tutte le persone che ti hanno aiutato a raggiungere questo traguardo, loro sono stati la tua più grande forza, ma ce ne hai messo anche del tuo, goditi il momento, hai finito.

Bibliography

[1] Sébastien Barré, Christoph Paasch, and Olivier Bonaventure. Multipath tcp: from theory to practice. In *NETWORKING 2011: 10th International IFIP TC 6 Networking Conference, Valencia, Spain, May 9-13, 2011, Proceedings, Part I 10*, pages 444–457. Springer, 2011.

[2] Dinamene de Lima Correia Almeida Barreia. Multipath tcp protocols. *Mém. de mast. Técnico Lisboa*, 2014.

[3] Martin Becke, Hakim Adhari, Erwin P Rathgeb, Fu Fa, Xiong Yang, and Xing Zhou. Comparison of multipath tcp and cmt-sctp based on intercontinental measurements. In *2013 IEEE Global Communications Conference (GLOBECOM)*, pages 1360–1366. IEEE, 2013.

[4] Quentin De Coninck and Olivier Bonaventure. Multipath quic: Design and evaluation. In *Proceedings of the 13th international conference on emerging networking experiments and technologies*, pages 160–166, 2017.

[5] Alan Ford, Costin Raiciu, Mark J. Handley, and Olivier Bonaventure. TCP Extensions for Multipath Operation with Multiple Addresses. RFC 6824, January 2013.

[6] Jiayue He and Jennifer Rexford. Toward internet-wide multipath routing. *IEEE network*, 22(2):16–21, 2008.

[7] Janardhan R Iyengar. *End-to-end concurrent multipath transfer using transport layer multihoming*. University of Delaware, 2006.

[8] Shaowei Liu, Weimin Lei, Wei Zhang, and Xiaoshi Song. Mpmtp-ar: Multipath message transport protocol based on application-level relay. *KSII Transactions on Internet & Information Systems*, 11(3), 2017.

[9] Yanmei Liu, Yunfei Ma, Quentin De Coninck, Olivier Bonaventure, Christian Huitema, and Mirja Kühlewind. Multipath Extension for QUIC. Internet-Draft draft-ietf-quic-multipath-04, Internet Engineering Task Force, March 2023. Work in Progress.