Capitolo 1

Integrazione DiffServ-MPLS

1.1 La qualità del servizio nelle reti IP

Le reti IP sono nate seguendo un approccio di tipo *best effort*, prive cioè della capacità di fornire all'utenza garanzie di qualità del servizio (QoS). Il modello basato su questo paradigma implementativo si è dimostrato valido fintanto che non si sono affacciate alla scena nuove classi di applicazioni con requisiti stringenti per quanto riguarda banda disponibile, ritardo e *jitter*. Servizi come *Voice over IP* (VoIP) o *streaming audio-video* hanno infatti reso insufficienti i meccanismi di gestione delle risorse utilizzati per Internet, che erano stati originariamente concepiti per tutt'altro tipo di traffico (FTP, TELNET, SMTP e simili) e che non accordano la possibilità di controllare attivamente importanti parametri come tempi di latenza, perdite di pacchetti, percentuale delle risorse di un link utilizzate da un flusso dati.

In questo scenario si è resa necessaria la ricerca di nuovi tipi di soluzioni

che possano fornire una risposta adeguata alle crescenti esigenze prestazionali richieste alle reti IP, anche nell'ottica di un possibile ritorno economico per i *provider* di servizi dalla stipulazione di diversi SLA (Service Level Agreement) con l'utenza.

Le condizioni da soddisfare per la QoS possono così compendiarsi:

- È necessaria l'allocazione garantita di una certa misura di risorse trasmissive in tutto il percorso effettuato dai dati dell'applicazione di interesse. Questa condizione di tipo *end-to-end* non deve venire meno neanche in situazioni di congestione o malfunzionamento di nodi della rete.
- Il flusso dati dell'applicazione deve sperimentare nei nodi della rete un trattamento specifico, scelto tra diverse possibili classi di servizio, che rispetti determinati vincoli pertinentemente alla perdita di pacchetti ed ai ritardi nelle code.

I due requisiti sono indipendenti e devono entrambi essere soddisfatti.

La prima proposta nell'ambito della qualità del servizio è stata l'architettura IntServ, basata sul protocollo RSVP (ReSerVation Protocol) per l'allocazione delle risorse. IntServ si basa sull'instaurazione di un ambiente *connection oriented*, in cui si riservano risorse per ogni singolo flusso dati con un'opportuna segnalazione e col mantenimento di informazioni di stato sul flusso in ogni nodo attraversato. Nonostante l'architettura IntServ sia in grado di soddisfare le condizioni viste, essa ha incontrato poco successo fino ad essere completamente messa da parte. Questo perché il trattamento singolo di ogni flusso dati comporta una considerevole complessità, che diviene insormontabile se si pensa a reti sempre più grandi con una quantità crescente di traffico di questo tipo.

Per superare queste problematiche di scalabilità si è passati ad un altro tipo di approccio che ha portato allo sviluppo dell'architettura DiffServ da parte dell'IETF (Internet Engineering Task Force) [1-1]. Essa rappresenta un primo reale passo avanti per l'implementazione della QoS, ma è anche da considerarsi incompleta in quanto non offre le garanzie quantitative *end-to-end* richieste dal primo dei due requisiti.
MPLS (Multiprotocol Label Switching) è una tecnologia emersa negli ultimi anni da un gruppo di lavoro sempre in ambito IETF. Spesso viene erroneamente citata come architettura per la qualità del servizio, mentre in realtà essa non fornisce di per sé alcun meccanismo per la gestione della QoS. Ciò che MPLS offre è la possibilità di lavorare in un ambiente di path *connection oriented*, nei quali è possibile riservare risorse ed utilizzare in modo efficiente la rete mediante applicazioni di *traffic engineering*. MPLS è quindi una tecnologia che può fornire il supporto per una completa gestione *end-to-end* della QoS.

Associare in un'unica architettura DiffServ e MPLS consente di soddisfare tutti i requisiti per la qualità del servizio nelle reti IP. Il lavoro presentato in questa tesi si colloca proprio all'interno di un progetto di integrazione di questo tipo, del quale si parlerà nel paragrafo 1.2.

1.1.1 DiffServ

Nell'architettura DiffServ [1-2] è stata abbandonata l'idea di trattare singolarmente ogni flusso dati. A tale scopo si definiscono delle classi di servizio (Classes of Service o CoS) che accolgono degli aggregati di traffico. L'appartenenza di un flusso dati ad un aggregato di traffico

determina quale sarà il trattamento riservato ai pacchetti di quel flusso nei nodi della rete. DiffServ consente infatti la gestione delle risorse interne ai nodi con un meccanismo di tipo *Per-Hop*. Non vi è alcun concetto di percorso del flusso dati come in IntServ: l'approccio è *connectionless*. Tutti i pacchetti di un flusso DiffServ portano con sé l'informazione sull'appartenenza ad una certa classe di servizio, che i nodi DiffServ-compatibili sono in grado di interpretare comportandosi poi di conseguenza nel trattamento del pacchetto.
Per come è concepita, l'architettura DiffServ soddisfa esclusivamente il secondo dei requisiti visti per la QoS. Non c'è alcun modo di garantire l'allocazione della capacità trasmissiva nei link attraversati.

Una CoS in ambito DiffServ specifica essenzialmente due parametri: il *Behavior Aggregate* (BA) e l'*Ordered Aggregate* (OA). Il primo indica i requisiti per lo *scheduling* e per le regole con cui vengono scartati i pacchetti; il secondo compie una classificazione basata esclusivamente sullo scheduling. In pratica l'OA è una classificazione meno approfondita ed ogni OA può comprendere più BA.
Il trattamento riservato ad un certo BA all'interno del nodo DiffServ è detto *Per Hop Behavior*, o PHB; un OA è servito da una *PHB Scheduling Class* o PSC. Una PSC identifica le regole di scheduling ed include diversi criteri di *discarding* per discriminare il comportamento nei confronti di diversi BA che fanno parte dell'OA ad essa relativo.

Le informazioni sulla CoS sono state incluse nell'header IP dei pacchetti in modo non invasivo. Si è difatti deciso di inserirle nel campo Type of

Service (ToS) di IPv4[1], nato proprio per indicare diversi metodi di gestione dei pacchetti, ma praticamente inutilizzato. In figura 1.1 si può vedere la nuova definizione degli otto bit del campo.

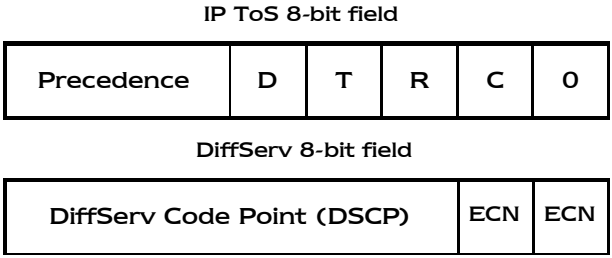

Figura 1.1 – Nuova definizione del campo IPv4 ToS

Nel vecchio campo ToS, le lettere indicano D = delay, T = throughput, R = reliability, C = cost. Nell'implementazione DiffServ i primi sei bit rappresentano il cosiddetto *DiffServ Code Point* o DSCP, che serve per specificare un BA (e quindi il relativo PHB che il pacchetto sperimenterà nei nodi DiffServ). Sono quindi possibili allo stato attuale 64 diversi PHB. Gli ultimi due bit servono per la *Explicit Congestion Notification* (ECN), un meccanismo sperimentale di *congestion avoidance.*

I PHB standardizzati

Dei sessantaquattro possibili valori per il campo DSCP, trentadue sono stati riservati per la creazione di PHB standard (quelli con ultimo bit 0), sedici sono stati riservati ad uso sperimentale (con ultimi due bit 01); i restanti sedici (con ultimi due bit 11) sono per adesso sperimentali ma saranno in

[1] Nel caso di IPv6 rimpiazzano invece il campo *Traffic Class*

futuro destinati alla standardizzazione qualora i primi trentadue non risultassero sufficienti.

Per adesso soltanto quattordici DSCP sono stati effettivamente assegnati a dei PHB standard. Tra questi figura la classe BE (Best Effort), presente per ovvie ragioni di compatibilità con l'architettura IP tradizionale. Ad essa è riservato il DSCP "000000".

Per quanto riguarda i DSCP che gestiscono realmente la qualità del servizio, sono presenti un PHB chiamato EF (Expedited Forwarding), ed una classe che raggruppa dodici diversi PHB detta AF (Assured Forwarding).

Il PHB Expedited Forwarding, detto anche *Premium Service*, è il migliore trattamento che possono sperimentare i pacchetti in ambito DiffServ. I dati appartenenti a questo BA vengono inoltrati con latenze e perdite minime all'interno dei nodi DiffServ. Per poter garantire un servizio di questo tipo è necessario che un nodo limiti il rate di arrivo dei dati appartenenti a questa classe al di sotto del rate al quale è in grado di inoltrarli. Se questa condizione non fosse soddisfatta si formerebbero delle code troppo lunghe e ci sarebbero attese e perdite non tollerabili. L'EF può essere realizzato assegnando ad una coda con massima priorità i pacchetti per concedere la dovuta precedenza, ma necessita anche di strumenti per il condizionamento del traffico per far rispettare il vincolo sul rate di arrivo. Il DSCP riservato per EF è "101110".

La classe di PHB Assured Forwarding raggruppa dodici tipi distinti di comportamenti. A sua volta, AF è suddivisa in quattro sottoclassi, ciascuna pertinente ad una diversa priorità di coda, quindi ad un differenziato utilizzo delle risorse disponibili nel nodo DiffServ. All'interno di ogni sottoclasse sono possibili tre diversi livelli di trattamento nello scartare i pacchetti (*drop precedence*). Si ottiene così il totale di dodici AF PHB, che vengono specificati con la notazione AF_{xy}, dove x varia da uno a quattro indicando la

sottoclasse, y varia da uno a tre indicando le regole di discarding. Valori più bassi degli indici significano qualità più alta. Per implementare il gruppo di AF PHB sono necessarie discipline di scheduling che consentano di gestire più code con diversa priorità (come ad esempio WRR, *Weighted Round Robin*) e algoritmi parametrizzabili per l'eliminazione dei pacchetti per poter impostare i diversi livelli di *drop precedence* (come ad esempio GRED, *Generalized Random Early Discard*). La classe AF offre nel complesso un servizio che non è ai livelli di quello della EF, ma è senz'altro migliore del *best effort*.

Blocchi funzionali dell'architettura DiffServ

Per garantire la messa in opera delle funzionalità QoS viste in precedenza i nodi DiffServ devono eseguire diverse operazioni sui pacchetti in arrivo, che possono essere suddivise in:

- funzioni di classificazione del traffico;
- funzioni di condizionamento del traffico.

La classificazione del traffico è il procedimento che consente di istradare i pacchetti in arrivo verso l'aggregato che riceverà il servizio relativo al corretto PHB. Il blocco funzionale che compie quest'operazione è chiamato *classifier*.

Il condizionamento del traffico serve ad assicurare il corretto funzionamento dei meccanismi per la QoS. Ad esempio, in quest'ottica rientra il controllare che la quantità di dati in arrivo per ogni BA non superi la capacità del nodo di smaltire il traffico di quell'aggregato. Questo è necessario perché altrimenti si osserverebbe un deterioramento delle prestazioni oltre i livelli accettabili rispetto ai dettami dei vari PHB.

Un blocco necessario immediatamente a valle del classifier è il cosiddetto *marker*, che ha due funzioni. In un nodo al confine di un dominio DiffServ, i pacchetti in ingresso devono essere marcati con l'appropriato DSCP in base alla loro provenienza; a questo provvede il marker. La seconda funzione è quella di marcare quei pacchetti che non rispettano un certo profilo di traffico previsto o desiderato, in modo da poter prendere i dovuti provvedimenti per la limitazione del traffico in ingresso al nodo. Il blocco che prende questi provvedimenti, che possono consistere nell'accodare il traffico di un aggregato in una struttura di tipo *token bucket* o nello scartare alcuni pacchetti, viene chiamato *shaper*. È altresì richiesto un blocco di misura detto *meter*, al quale è demandato l'incarico di valutare se un certo aggregato di dati è in linea con il profilo di traffico previsto oppure no. Sia il marker che lo shaper fanno uso dei dati provenienti dal meter.
Naturalmente nel nodo DiffServ devono essere anche implementate le code con diversa priorità per servire i diversi OA, e all'interno di esse gli algoritmi per discriminare i pacchetti secondo diverse *drop precedence*.

1.1.2 MPLS e *traffic engineering*

L'architettura MPLS (Multiprotocol Label Switching) [1-3] rappresenta per le reti IP[2] una sostanziale innovazione: l'introduzione in un ambito da sempre *connectionless* della possibilità di instradare flussi di traffico con un approccio orientato alla connessione. Tra i vantaggi di MPLS il più importante è senz'altro la possibilità di avvalersi di procedure di *traffic engineering.*

[2] MPLS è stato progettato per essere indipendente dai protocolli di livello 2 (collegamento) e 3 (rete), anche se in realtà trova il suo principale utilizzo nelle reti IP.

Un altro beneficio che merita menzione è il forte contributo all'integrazione IP-ATM, semplificata dall'introduzione della nuova tecnologia MPLS.

Il routing IP tradizionale (*destination based*) presenta il considerevole limite di tener conto esclusivamente della destinazione dei pacchetti, il che comporta uno sfruttamento inefficiente delle risorse della rete. Non è infatti possibile reagire a situazioni di sbilanciamento del carico della rete o impostare criteri efficienti di ottimalità nella scelta dei percorsi.
Questi, che sono i requisiti del *traffic engineering*, sono invece resi possibili dalle nuove funzionalità garantite dalla tecnica di *forwarding* di tipo *label switching*. Per label switching si intende un tipo di forwarding basato su un'etichetta (label) di lunghezza fissa, portata da ogni pacchetto.

Architettura MPLS

Un dominio MPLS è costituito da una serie di nodi contigui che supportano la tecnologia Multiprotocol Label Switching, detti LSR (Label Switching Router). In questo scenario sono di particolare importanza i nodi al confine del dominio, che nella terminologia MPLS vengono chiamati LER (Label Edge Router). Essi rappresentano l'interfaccia del dominio MPLS con il resto della rete, quindi devono implementare sia l'algoritmo di forwarding label switching sia quello tradizionale IP. Un altro loro compito è quello di assegnare i label al traffico in ingresso.
Un percorso effettuato dai pacchetti instradati tramite label switching all'interno del dominio MPLS prende il nome di LSP (Label Switched Path).

Da un punto di vista funzionale, l'architettura MPLS può essere divisa (come del resto anche l'architettura tradizionale IP) in componente di forwarding e componente di controllo.

La componente di forwarding consiste nella procedura mediante la quale in un nodo si estraggono dai pacchetti le informazioni per identificare l'appropriato *next hop* nella *forwarding table*. Un LER esegue il mappaggio del traffico in classi di equivalenza per il forwarding (FEC), ossia in gruppi di pacchetti che devono essere tutti inoltrati nella stessa maniera. Di seguito associa l'etichetta ai pacchetti in base alla loro FEC di appartenenza. All'interno del dominio MPLS, ogni LSR ha una tabella di forwarding nella quale ad un determinato label corrisponde un *next hop*. L'indirizzo di destinazione finale dei pacchetti non è più preso in considerazione nelle decisioni riguardo al loro inoltro. Un parametro importante per la componente di forwarding è la granularità con cui i flussi in ingresso al dominio MPLS vengono associati nelle FEC. In linea teorica sarebbe possibile associare una diversa etichetta ad ogni singolo flusso, ma questo comporterebbe i medesimi problemi dell'architettura IntServ. Il pregio di MPLS da questo punto di vista è la possibilità di selezionare arbitrariamente i criteri per l'aggregazione di più flussi in classi di equivalenza per il forwarding.

Dal momento che solo alcuni protocolli di livello 3 (ATM, Frame Relay) supportano la presenza di un label, si è deciso di creare in ambito MPLS un header di 32 bit da inserire tra quello di livello 2 e quello di livello 3, detto *shim header*. Tutti i protocolli di livello 3 che non prevedono la presenza di un label (tra i quali rientra IP) possono fare uso dello shim header per trasportare le informazioni per il label switching. La struttura dello shim header è mostrata in figura 1.2 a pagina seguente.

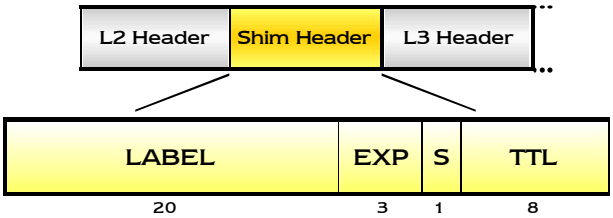

Figura 1.2 – Shim header

Il campo di 20 bit Label è l'etichetta vera e propria. Il campo Exp, di tre bit, è stato riservato ad uso sperimentale ed è spesso utilizzato per il supporto della qualità del servizio in ambito MPLS. Il bit S serve per la tecnica di *label stacking*, che consiste nell'inserire nello shim header più label da gestire come una pila LIFO (Last In First Out), ed è utile per ottimizzare i processi di forwarding. Il campo di 8 bit TTL (Time To Live) ha funzione analoga a quella del corrispettivo campo IP, ed è usato per la prevenzione e la mitigazione degli effetti dei loop di routing.

La componente di controllo nell'architettura MPLS consiste nelle procedure utilizzate per scambiare informazioni di routing tra gli LSR e per convertire tali informazioni in una tabella di forwarding. Come la componente di controllo dell'architettura tradizionale, anche quella di MPLS include dei protocolli di routing (ad esempio OSPF). Per la creazione delle tabelle di forwarding sono però necessarie in un LSR ulteriori funzionalità oltre ai protocolli di routing. Un LSR deve essere in grado di creare associazioni (*bindings*) tra label e FEC, informare gli altri LSR dei bindings ed aggiornare le tabelle di forwarding in base ai bindings creati localmente ed a quelli di cui viene informato dagli altri LSR. I protocolli di routing consentono di effettuare il passaggio da FEC a *next hop*; le procedure di creazione e distribuzione dei bindings portano alla conversione da FEC a label.

Complessivamente si ottiene la relazione tra label e *next hop* necessaria per la creazione delle tabelle di forwarding, come mostrato in figura 1.3.

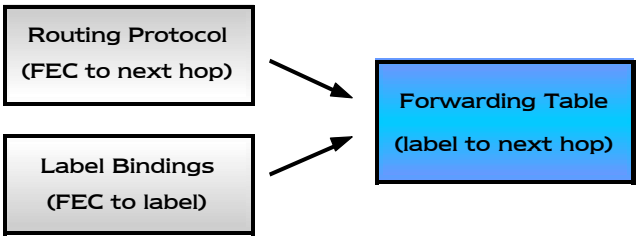

Figura 1.3 – Componente di controllo dell'architettura MPLS

Nell'architettura MPLS si utilizza per il traffico *unicast* una procedura di label binding detta *downstream*, in quanto il binding FEC - label è creato a valle rispetto alla direzione del flusso dei pacchetti di quella FEC. Sono possibili due modalità: downstream *unsolicited* e downstream *on demand*. Nella prima le informazioni sui bindings vengono inviate a tutti gli LSR coinvolti, nella seconda vengono inviate soltanto a quelli che ne fanno espressa richiesta. L'assegnamento e la rimozione dei label posono avvenire sia in risposta al traffico effettivamente presente (*data driven*) sia in risposta all'arrivo di informazioni della componente di controllo (*control driven*); tra le due opzioni è solitamente preferita la seconda. La distribuzione dei label bindings viene affidata preferibilmente a protocolli tradizionali (RSVP) ma può anche essere demandata ad un apposito protocollo detto LDP (Label Distribution Protocol).

MPLS consente di scegliere i criteri di unicità per i label. È infatti possibile creare label che siano unici a livello di interfaccia di un LSR, a livello di un LSR oppure a livello dell'intero dominio.

L'instaurazione degli LSP può avvenire secondo due modalità: *ordered* ed

indipendent. Nel primo caso l'assegnamento dei bindings tra FEC e label procede in una maniera ordinata dal nodo di ingresso al nodo di uscita (o viceversa) di un LSP. Nel secondo caso ogni LSR compie in modo indipendente le decisioni sui binding e su come informare gli altri LSR, e l'instaurazione di un LSP è conseguenza della convergenza del routing.

Traffic engineering

L'architettura MPLS è nata per aumentare le prestazioni dei nodi della rete, introducendo per il forwarding un campo di dimensione fissa che consentisse un *processing* più rapido dei pacchetti. Tuttavia, questo tipo di necessità è progressivamente venuto meno in conseguenza delle continue innovazioni tecnologiche che hanno portato a router sempre più performanti. Il vero vantaggio di MPLS, che non rientrava tra i suoi scopi originari, è quello di consentire ai *service provider* di implementare il TE (traffic engineering) nelle reti, aumentandone l'efficienza e le prestazioni. Il TE permette di bilanciare il traffico in una rete in modo da non avere link congestionati né scarsamente utilizzati, cosa che porta ad un pieno sfruttamento della rete, con il conseguente guadagno per i provider che possono servire più utenza a parità di risorse. Inoltre il TE consente l'allocazione delle risorse trasmissive per un aggregato di flussi, rendendo possibile lo sviluppo di meccanismi di QoS *end-to-end.*

Lo sviluppo del TE ha importanti implicazioni per le procedure di routing. Nell'architettura IP tradizionale la selezione del percorso per l'instradamento del traffico viene effettuata minimizzando una determinata metrica, rappresentata ad esempio dal numero di nodi attraversati o dalla somma di termini di costo assegnati staticamente ai link. La conseguenza di ciò è uno

sbilanciamento del traffico, instradato per la maggior parte su percorsi preferenziali che in tal modo sono soggetti a congestione. In questo contesto poco flessibile mancano le premesse per poter sviluppare il TE.

MPLS supera i limiti dell'architettura tradizionale supportando un nuovo tipo di routing, detto *Constraint-based Routing* (CR). L'idea alla base di questa innovazione sta nel fatto che per avere un routing efficiente bisogna instradare il traffico lungo percorsi a costo minimo, rispettando contemporaneamente determinati vincoli (*constraints*) sullo sfruttamento delle risorse di rete. Un esempio di vincolo è trovare un percorso con una certa minima capacità disponibile in ogni link. Altri esempi potrebbero essere l'evitare o l'includere un determinato link. Soltanto un nodo di ingresso può determinare un percorso secondo il CR, poiché i nodi interni non sono a conoscenza dei vincoli da rispettare. Siamo quindi in presenza di un meccanismo di *explicit* routing (o *source* routing).

Per poter effettuare il CR sono necessarie funzionalità aggiuntive nella rete, tra le quali la presenza di protocolli di routing opportunamente estesi (come ad esempio OSPF-TE o ISIS-TE) per portare le informazioni relative al TE. Servono poi algoritmi per il calcolo dei percorsi che rispettino i vincoli, ed un protocollo per l'allocazione delle risorse esteso per il TE; il più famoso è senz'altro RSVP-TE (ReSerVation Protocol with Tunneling Extension), che è molto utilizzato per la distribuzione dei label nelle reti che implementano il traffic engineering e la QoS. Le informazioni sulle richieste di allocazione saranno fornite a RSVP-TE in seguito alle opportune procedure di *admission control*, necessarie per stabilire se vi siano le risorse disponibili per accettare una determinata richiesta.

RSVP-TE opera in ambito MPLS segnalazioni tra due LER, trattando non il singolo flusso (come RSVP tradizionale nell'architettura IntServ) ma degli aggregati di flussi chiamati Traffic Trunk (TT). Questo riduce l'*overhead*

introdotto dal TE per le informazioni di controllo. Più TT possono confluire in un singolo LSP.

Riepilogando, l'architettura MPLS con i protocolli di routing opportunamente estesi fornisce il supporto al TE, grazie alla classificazione del traffico in TT ed all'utilizzo del CR. In questo modo si ottiene un ambiente *connection-oriented* in cui si ha uno sfruttamento efficiente della rete e la possibilità di riservare risorse *end-to-end* per un percorso all'interno del dominio. Grazie alle tecniche di *Fast Recovery* possibili con il CR la rete può reagire prontamente ed in modo valido ad avarie in un nodo o in un link. Tutto questo soddisfa il primo dei due requisiti per la QoS visti nel paragrafo iniziale di questo capitolo.

1.1.3 QoS con l'utilizzo combinato di MPLS-TE e DiffServ

Nel precedente paragrafo si è visto come MPLS, con le opportune estensioni per il traffic engineering (ed indicato in questo caso come MPLS-TE), possa rispettare il primo dei due requisiti per la QoS, garantendo risorse trasmissive ad aggregati di traffico ma non discriminando i pacchetti nel trattamento ad essi riservato nei nodi. L'architettura DiffServ è esattamente complementare in questo senso. La classificazione dei pacchetti in BA e il loro trattamento secondo diversi PHB soddisfano esclusivamente il secondo requisito. L'integrazione delle due architetture porta quindi allo sviluppo di una meccanismo per gestire in modo esaustivo la QoS, come descritto in [1-4] e [1-5]. L'architettura MPLS con supporto per DiffServ risulta molto più semplice e scalabile rispetto ad IntServ con RSVP. Infatti IntServ richiede segnalazione e memorizzazione dello stato in ogni nodo per ogni singolo

flusso. Gli LSP contengono aggregati di flussi, e la segnalazione viene ridotta. Inoltre, gli LSR non hanno bisogno di mantenere in memoria le informazioni per ogni flusso, ma per aggregati di traffico.

Affinché MPLS possa supportare DiffServ, è necessario che gli LSR riescano a distinguere i vari pacchetti in base al loro DSCP per inoltrarli secondo il PHB corrispondente. Il DSCP si trova, come detto in precedenza, nel campo ToS dell'header IP. Sorge allora un problema, in quanto gli LSR si basano esclusivamente sulla label dello shim header MPLS per il forwarding dei pacchetti, e non esaminano l'header IP. Le soluzioni che sono state proposte sono due. La prima prevede l'utilizzo del campo Exp dello shim header come sostituto del campo ToS per portare l'informazione sul DSCP all'interno del dominio MPLS. L'inconveniente di questo approccio consiste nella dimensione del campo Exp, di soli tre bit contro i sei del campo ToS: in questo modo si possono supportare al massimo otto diversi DSCP (e quindi otto diversi PHB). Nel caso di una rete nella quale siano effettivamente implementati o richiesti al massimo otto PHB, le funzioni DiffServ possono essere esplicate semplicemente leggendo negli LSR il valore del campo Exp ed assegnando di conseguenza i pacchetti al corretto BA. Questo metodo ha il pregio di essere semplice e non richiedere alcuna segnalazione di controllo aggiuntiva. Un LSP per il quale il campo Exp è utilizzato in questo modo prende il nome di E-LSP.
La seconda soluzione proposta è utile qualora si debbano trattare più di otto PHB. In tal caso il campo Exp non è più sufficiente, e lo shim header necessita di essere in qualche modo rivisitato per consentire il trasporto dell'informazione sul DSCP. Si è deciso di incrementare il significato del campo label, che deve indicare sia l'appartenenza ad una certa FEC che l'appartenenza ad un OA servito da una PSC DiffServ; il campo Exp viene

in questo caso utilizzato per esprimere il valore della *drop precedence*, in modo da poter selezionare un determinato BA all'interno dell'OA e trattare i pacchetti secondo l'opportuno PHB all'interno della PSC. Parlando ad esempio della categoria di PHB AF, nella label si troverebbe l'informazione sullo scheduling e quindi riguardo alla coda sulla quale instradare il pacchetto; nel campo Exp si troverebbe il valore (tra i tre possibili per ogni classe di scheduling di AF) della drop precedence. Un LSP costruito con queste regole viene detto L-LSP; è necessario far confluire i pacchetti di medesimo OA in un L-LSP comune, poiché sono destinati tutti alla stessa coda. Questo modo di procedere consente di avere quanti PHB si vogliano, a spese di una complicazione nella componente di controllo MPLS. Infatti è necessario estendere i protocolli per la distribuzione dei binding tra label e FEC, che adesso devono includere anche il binding tra label e PHB. La distribuzione deve essere effettuata al momento della creazione di un L-LSP.

Le due alternative per gli LSP con supporto di DiffServ non sono mutuamente esclusive e possono coesistere non solo a livello di dominio MPLS ma anche a livello di un singolo link.

Diffserv-aware MPLS Traffic Engineering (DS-TE)

Appurato che l'architettura MPLS-TE con supporto per DiffServ è il criterio ad oggi più appropriato per implementare la qualità del servizio nelle reti IP, gli sforzi attuali del gruppo di lavoro IETF Traffic Engineering Work Group (per i quali si rimanda al riferimento [1-6]) vanno nella direzione di rendere sempre più affini ed efficientemente coesi il TE in ambito MPLS e l'architettura DiffServ.

Lo scopo primario del DS-TE è quello di garantire le risorse di rete

separatamente per ogni tipo di traffico. Il TE viene effettuato sulla CoS (Class of Service), definita come un set di OA, mediante procedure di *admission control* e *resource reservation* calibrate sulle classi di traffico. Si introducono in questo ambito i concetti di *Class Type* (CT), un raggruppamento di TT basato sulla loro CoS, e di *Bandwith Constraint* (BC), un limite sulla percentuale di risorse trasmissive di un link che un CT può utilizzare. Le relazioni tra CT e BC sono definite nei cosiddetti *Bandwidth Constraint Models*, che sanciscono le regole per la suddivisione delle risorse trasmissive tra i vari CT.

1.2 Il trial sperimentale DiffServ-MPLS

Il lavoro presentato in questa tesi si inserisce all'interno del progetto *Tango* (Traffic models and Algorithms for Next Generation IP networks Optimization) [1-7]. Nel contesto di questo progetto, il *TLC NetGroup* [1-8] dell'Università di Pisa sta progettando ed implementando un'architettura di rete IP multiservizio avanzata, che presenti un efficiente controllo del traffico ed il supporto per la QoS. Le attività di ricerca in tal senso hanno condotto alla definizione dell'architettura MAID (Multiple Access Inter-Domain), della quale si parla in questo paragrafo. Si rimanda il lettore interessato a maggiori dettagli su questa architettura ai riferimenti bibliografici [1-9], [1-10].

1.2.1 L'architettura MAID ed i suoi componenti

L'architettura Multiple Access Inter Domain è una possibile soluzione per

l'integrazione di MPLS e DiffServ , ed ha come scopo il fornire ai *Network Operators* un meccanismo valido e semplice per la gestione di reti con QoS. Essa è basata su un manager centralizzato per il calcolo dei path, espediente efficace per la funzionalità e la flessibilità in uno scenario di domini multipli.
L'architettura MAID è stata implementata, al fine di analizzarne le prestazioni, in un *testbed* sperimentale costituito da PC Linux. È stato necessario lo sviluppo di protocolli estesi (RSVP-TE, COPS-MAID) per supportare le funzionalità richieste. I PC del testbed sono equipaggiati con la versione 2.4.20 del kernel Linux con le estensioni per MPLS e per il traffic conditioning, con il modulo Zebra per il supporto di OSPF-TE e con una versione modificata del *daemon* RSVP-TE.

Gli elementi chiave dell'architettura MAID sono il Bandwidth Broker (BB) ed i Multiple Access Label Edge Router DiffServ (MA-LER DS). Il primo è l'elemento centralizzato preposto al management delle risorse di rete ed al calcolo dei path; i secondi rappresentano la frontiera del dominio MAID e pertanto svolgono il ruolo cruciale di MPLS Label Edge Router (LER) e DiffServ Border Router (BR), fornendo l'interfaccia con le reti di accesso. La comunicazione tra il BB ed i MA-LER DS avviene a mezzo del protocollo esteso COPS-MAID (Common Open Policy Service - MAID).
La procedura di instaurazione di un LSP può avvenire come conseguenza di una richiesta esplicita dell'amministratore di rete (*provisioning*) oppure di una richiesta implicita proveniente dalle reti di accesso al dominio MAID (*outsourcing*). Il MA-LER DS informa il BB, a mezzo del protocollo COPS, delle richieste che riceve. A sua volta il BB verifica che ci siano risorse disponibili per soddisfare la richiesta, quindi risponde, sempre con COPS, al MA-LER DS informandolo dell'eventuale Explicit Route calcolato. Se la

richiesta può essere accolta, parte dal MA-LER l'instaurazione dell'LSP tramite l'utilizzo del protocollo RSVP-TE.

Il protocollo COPS-MAID funziona con modello client-server ed è basato su due elementi costitutivi: un server per il policing, chiamato Policy Decision Point (PDP), situato all'interno del BB, ed uno o più client, chiamati Policy Enforcement Point (PEP), dislocati nei MA-LER DS. Rispetto al protocollo COPS tradizionale è stato esteso per fornire una semantica comune indipendente dal protocollo di accesso al dominio MAID utilizzato per le richieste di instaurazione LSP (RSVP, RSVP-TE, H.323, SIP, MPEG-4). Elementi presenti nei MA-LER DS e detti Inter Working Units (IWU) provvederanno a traslare le richieste di questi protocolli di accesso nella semantica comune utilizzata da COPS-MAID. In questo modo un solo PDP centralizzato risulta sufficiente.

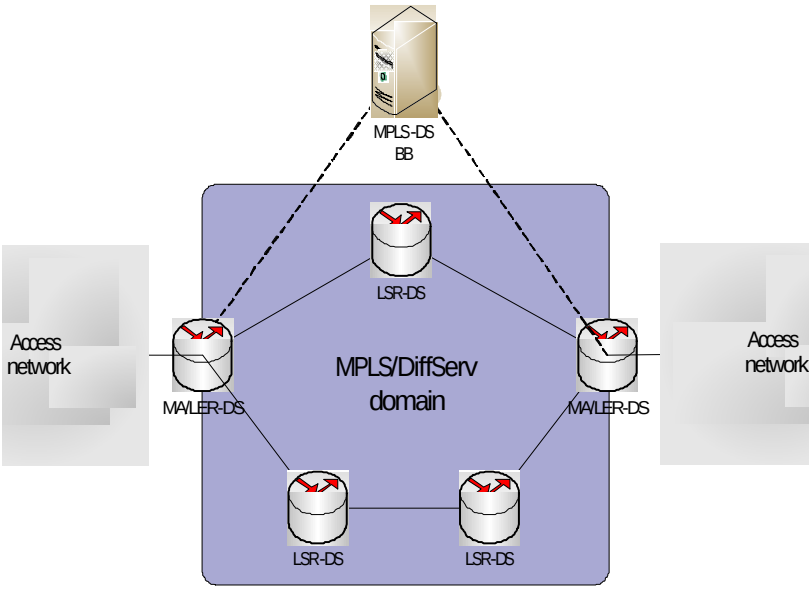

Figura 1.4 – Architettura MAID

Componenti dell'architettura MAID

Oltre ai due componenti fondamentali già visti, il BB ed il MA-LER DS, ne esiste un terzo, detto Label Switching Router DiffServ (LSR DS), che rappresenta un nodo interno al dominio MAID, e quindi non comunica con le reti di accesso al dominio. La struttura dell'architettura MAID è mostrata in figura 1.4 a pagina precedente.

MA-LER DS: È, come detto in precedenza, un elemento cardine di questa architettura. Racchiude al suo interno le funzionalità di un Border Router DiffServ (*classification, metering, marking, policing, dropping, scheduling, shaping*) e quelle di un Label Edge Router MPLS (tabelle di forwarding MPLS, assegnamento label). I suoi compiti a livello di componente di controllo sono:

- Gestire le richieste di instaurazione degli LSP.
- Gestire le richieste di servizio provenienti, con diversi protocolli di segnalazione, dalla rete di accesso al dominio MAID.

Nel primo caso, le azioni compiute dal MA-LER sono l'autenticazione, l'admission control, la comunicazione della richiesta al BB tramite COPS-MAID, l'instaurazione degli LSP tramite RSVP-TE in seguito alla replica del BB contenente l'explicit route calcolato. Nel secondo caso, dopo autenticazione ed admission control, il MA-LER provvederà a far confluire il traffico sull'appropriato LSP associandolo al corretto PHB.
Si parlerà con dettaglio della componente di forwarding dei MA-LER DS nel paragrafo 1.2.2, in quanto trattasi di argomento strettamente correlato con l'oggetto di questa tesi.

LSR DS: Si tratta di un nodo interno al dominio MAID, ed implementa un

sottoinsieme delle funzioni di un MA-LER DS. Svolge infatti il ruolo di Core Router DiffServ e di Label Switching Router MPLS.

MPLS DS BB: Il Bandwith Broker MPLS DiffServ è l'unità centrale di gestione del dominio MAID. Esso deve essere costantemente a conoscenza della topologia della rete e delle risorse disponibili. Queste informazioni sono ricavate a mezzo SNMP oppure OSPF-TE. Le funzioni che il BB svolge sono le seguenti:

- Calcolo del path per gli LSP (con eventuale path di backup) tenendo di conto dei parametri di QoS richiesti. Gli algoritmi utilizzati sono di tipo Constraint-based Routing.
- Comunicazione dei path calcolati ai MA-LER DS.

1.2.2 Piano dati nel MA-LER DS

Nel precedente paragrafo si è parlato delle funzionalità del MA-LER DS a livello di componente di controllo. Si esamina adesso la componente di forwarding, che consiste nelle operazioni svolte sul pacchetto per la gestione dell'inoltro. Al suo interno convivono la parte riguardante DiffServ e quella relativa a MPLS. Alla prima compete la classificazione dei pacchetti in base a diversi parametri ed il loro scheduling secondo le discipline dettate dai PHB, alla seconda la determinazione della FEC (e quindi del label) e l'aggiunta (o rimozione) dello shim header. In figura 1.5 a pagina seguente è mostrato lo schema riassuntivo della componente di forwarding.
Per quanto riguarda la componente DiffServ, il primo blocco è un classificatore *Multi-Field*, che distingue i pacchetti in base ad una combinazione dei campi degli header di livello 3 o superiore (come ad esempio indirizzo o

porta sorgente/destinatario). Segue il blocco meter/policer, che serve a contenere il traffico entro certi limiti (in termini di *rate* e *burst size*). Si incontra poi il marker, che ha la funzione di marcare il campo DSCP dell'header IP (che sarà poi utilizzato dalla componente MPLS per marcare a sua volta il campo Exp dello shim header). Il pacchetto sarà inoltrato in base al BA di cui fa parte. Gli algoritmi utilizzati sono HTB (Hierarchical Token Bucket) per lo scheduling e la distribuzione delle risorse tra le varie classi (PSC) e GRED (Generalized Random Early Detection) per il dropping, con tre diversi valori di *drop precedence.*

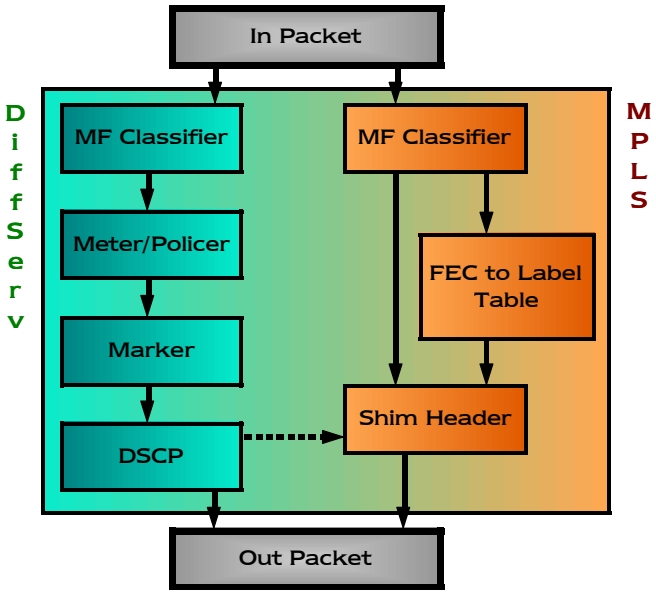

Figura 1.5 – Componente di forwarding del MA-LER DS

Anche nella componente MPLS incontriamo per primo un classificatore Multi-Field. Esaminando il pacchetto ne stabilisce la FEC di appartenenza,

e consultando le tabelle costruite dalla componente di controllo gli assegna un opportuno label, completando poi lo shim header in base al DSCP impostato dalla componente DiffServ.

Linux networking nel MA-LER DS

Nel trial sperimentale sono dei PC Linux a svogere il ruolo dei vari componenti dell'architettura MAID. Si fornirà adesso una descrizione sintetica e non approfondita delle funzionalità del networking di Linux utilizzate per ricoprire le funzioni DiffServ-MPLS in un MA-LER DS.

Per poter inoltrare i pacchetti in base alla FEC (e quindi in modo diverso da quello tradizionale IP destination-based) è necessario che il kernel di Linux effettui il routing basandosi su un parametro diverso dall'indirizzo IP del destinatario. Questo tipo di procedura si chiama *advanced routing*, e si basa su un parametro interno al kernel chiamato *fwmark*. Si possono creare delle tabelle di forwarding MPLS configurando opportunamente il *netfilter* (un'altra componente del kernel che consente il filtraggio e la modifica dei pacchetti) in modo da assegnare ad un pacchetto un dato fwmark in base ad una serie di campi del pacchetto stesso (IP destinatario, porta TCP…). In parole povere, il fwmark svolge il ruolo di identificatore della FEC al quale viene associato il pacchetto. L'aggiunta dello shim header ai messaggi in uscita viene effettuata da un'interfaccia di rete virtuale mediante la quale Linux implementa le funzionalità MPLS. L'interfaccia di rete virtuale sarà configurata come destinazione di una data FEC (identificata dal fwmark) e rappresenterà il punto di partenza di un LSP. Nell'architettura MAID gli identificativi per gli LSP sono unici a livello di tutto il dominio.
Per implementare le funzionalità DiffServ nel MA-LER DS si ricorre ad

un'altra componente di Linux chiamata *traffic control*. Essa offre funzioni di policing, scheduling e classificazione. Anche in questo caso esiste un parametro interno al kernel, detto *tcindex*, che consente di caratterizzare il pacchetto associandolo ad un BA e consentendo inoltre di marcare il DSCP nell'header IP del pacchetto stesso. In pratica il tcindex identifica il PHB al quale è destinato un pacchetto.

Riassumendo, un pacchetto in ingresso al MA-LER DS passa prima dal policer che imposta il valore di tcindex (a meno che il pacchetto non sia da scartare perché il traffico non rispetta le dovute condizioni) in base ai vincoli contrattuali riguardanti lo specifico flusso dati. Segue poi il netfilter che decide il fwmark esaminando vari campi del pacchetto. Il pacchetto sarà instradato verso l'opportuna interfaccia virtuale MPLS in base al fwmark e alle tabelle di advanced routing, ed infine inoltrato mediante un classifier che marca il DSCP ed implementa i meccanismi per la messa in atto dei vari PHB (code e *drop precedence*).